# CONTROL OF BURGERS' EQUATION WITH MIXED BOUNDARY CONDITIONS

by

**Kenneth L. Massa**

**Thesis submitted to the faculty of the**

**Virginia Polytechnic Institute and State University**

**in partial fulfillment of the requirements for the degree of**

**MASTER OF SCIENCE**

**Mathematics**

**APPROVED:**

---

**John A. Burns, Chairman**

---

**Robert C. Rogers**

---

**Terry L. Herdman**

**March, 1998**

**Blacksburg, Virginia**

# CONTROL OF BURGERS' EQUATION WITH MIXED BOUNDARY CONDITIONS

**by**

**Kenneth L. Massa**

**Committee Chairman: John A. Burns**

**Mathematics**

**(ABSTRACT)**

We consider the problems of simulation and control for Burgers' equation with mixed boundary conditions. We first conduct numerical experiments to test the convergence and stability of two standard finite element schemes for various Robin boundary conditions and a variety of Reynolds numbers. These schemes are used to compute LQR feedback controllers for Burgers' equation with boundary control. Numerical studies of these feedback control laws are used to evaluate the performance and practicality of this approach to boundary control of non-linear systems.

# ACKNOWLEDGEMENTS

I would like to give my most sincere thanks to my friend, mentor, and thesis comittee chairman Dr. John Burns for his support, motivation, and guidance which made this research possible. Also, I want to thank him for his career insight, valuable courses, and general advice he has provided for me over the years I have known him. I thank Dr. Terry Herdman and Dr. Robert Rogers for serving on my thesis committee and for their guidance throughout my years at Virginia Tech.

I wish to acknowledge the Department of Mathematics and the Interdisciplinary Center for Applied Mathematics for the financial support provided for me as a graduate student. I also acknowledge the financial support I received from the Air Force Office of Scientific Research under Grant F49620-96-1-0329.

I would like to give many thanks to my parents, Robert and Virginia, and the rest of my family for the constant encouragement, unconditional love, and unyielding devotion to my welfare and education. This paper is dedicated to the memory of my mother, Carol, who instilled in me the importance of life, knowledge, and cherishing every experience, whether good or bad.

I thank all my friends on the Virginia Tech swim team. I would also like to thank my closest buddies Collins, Shawn, Russ, Seth, Skippy, Gordo, Mike, Doogie, Chip, Nancy, Angel, Sarah, JJ, Adam, Steve, Dave, Doug, and Jon for all the great times we had over the years at Tech. I give special thanks to Tom, Shana, Jason and Lyle. Without your

friendship and help, I would not have been able to earn my degree.

A special thank you goes to Cara, for the love and friendship she has given me throughout the past eight years. I would like to thank Paul and Barb for all the love, support, and advice they provided for so many years to help me grow emotionally. Finally, I wish to extend my thanks to Dr. Leone who made me believe in my abilities when I doubted myself.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Introduction

Burgers' equation is a useful model for physical phenomena involving non-linear wave propagation subject to dissipation. Although formulated by Burgers to exhibit the essential features of turbulence in hydrodynamic flows, it is often used in models of such physical problems as shock flow, traffic flow, acoustic transmission in fogs, air flow over an airfoil, heat flow through a material, etc. The associated dissipation may arise from viscosity, heat conduction, chemical reaction, mass diffusion, thermal radiation, or other source. In short, all problems modeled by Burgers' equation can be related to an evolutionary process in which convection and diffusion are in conflict, as was done in [11].

Burgers' equation

$$u_t(t,x) + u(t,x)u_x(t,x) = \epsilon u_{xx}(t,x) + f(t,x) \tag{1.1}$$

is a quasi-linear parabolic partial differential equation which describes the evolution of the function $u$ with respect to time. The convective term is the non-linear term, $u(t,x)u_x(t,x)$, and the diffusive term is $\epsilon u_{xx}(t,x)$. The forcing function $f(t,x)$ is set to zero to make the equation homogeneous. Fletcher [11] illustrates the difficulty of computing fluid dynamic problems which arises from the inability to efficiently balance the non-linear convective term and the diffusive term. Without the non-linear convective term, equation (1.1) becomes the linear, parabolic partial differential equation

$$u_t(t,x) = \epsilon u_{xx}(t,x) + f(t,x) \tag{1.2}$$

known as the heat equation. Dropping the diffusive term, equation (1.1) becomes

$$u_t(t,x) + u(t,x)u_x(t,x) = f(t,x) \tag{1.3}$$

which is a hyperbolic partial differential equation modeling the convection of disturbances in inviscid flow. A typical convecting wave from this equation has points with larger $u$ convecting faster than points on the wave with smaller $u$ causing the function to take on more than one value at a future time which is illustrated in [11]. For this reason, it is necessary to postulate a shock at which $u$ is discontinuous in order to have a unique solution.

Fletcher also illustrates the evolution of the solution in which the processes of convection and diffusion work together. Omitting $u_t$ from (1.1), the equation becomes

$$u(t,x)u_x(t,x) = \epsilon u_{xx}(t,x) + f(t,x) \tag{1.4}$$

which is an elliptic partial differential equation. As time progresses, the maximum amplitude of $u$ becomes smaller and the profile steepens. The dissipative term, $\epsilon u_{xx}$, becomes larger as the steepening occurs. This compensates for the convective term and does not allow a multivalued solution to develop. With these simplifications, Burgers' equation furnishes a simple non-linear model for convection/diffusion interactions. By the nature of Burgers' equation, many problems can be modeled, approximately or exactly, for different combinations of initial and boundary conditions.

Pugh [14] uses the finite element method with the Galerkin and the Galerkin/Conservation forms of Burgers' equation to approximate solutions to the initial/boundary value problem (IBVP)

$$u_t(t,x) + u(t,x)u_x(t,x) = \epsilon u_{xx}(t,x) + f(t,x) \tag{1.5}$$

$$u(0,x) = \phi(x) \tag{1.6}$$

$$u_x(t,0) = u_x(t,1) = 0 \tag{1.7}$$

where $\epsilon = \frac{1}{Re}$ is the reciprocal of the Reynolds number. The Galerkin/Conservation form of Burgers' equation changes the non-linear term so that Burgers' equation is represented

as

$$u_t(t,x) + \frac{1}{2}(u^2(t,x))_x = \epsilon u_{xx}(t,x) + f(t,x). \tag{1.8}$$

A solution to this IBVP describes the evolution of the function $u(t,x)$ over time, starting at $u(0,x) = \phi(x)$, constrained to the homogeneous Neumann boundary conditions (1.7) on the closed x-interval [0,1]. The methods show accurate modeling of Burgers' equation, when compared to exact solutions, for specific examples of $u(t,x)$ with N=16 elements. Pugh compares computational results for the Galerkin and Galerkin/Conservation methods for specific initial conditions and Reynolds numbers. Both methods produce virtually identical results when both approximations converge to a steady state solution. In a few examples, however, the Galerkin solution grows exponentially in time despite the convergent behavior of the Galerkin/Conservation solution. Also, Pugh noted that in many cases the Galerkin/Conservation method executed faster and was often more accurate than the Galerkin method.

Pugh's primary goal was to determine the form of the steady state solutions, i.e. the solutions $u_{ss}(x)$ such that $u(t,x) \to u_{ss}(x)$ as $t \to \infty$, where $u(t,x)$ solves the initial/boundary value problem (1.5, 1.6, and 1.7). Byrnes, Gilliam, and Shubov [6] proved that for $\epsilon > 0$ there is a constant $k$ such that $u_{ss}(x) \equiv C$ for some constant $C$ as long as $\|\phi(x)\| \leq k\epsilon$. Pugh showed that for the initial function $\phi(x) = x^2(1-x)^2$ and the Reynolds number $Re = 10$, the finite elements method reproduced the results on page 44 of [6]. However, some of Pugh's numerical results (see Examples 3.2.4, 3.3.1, 3.4.2, 3.4.7) illustrated

phenomena where large initial conditions evolved to steady state solutions which are not everywhere constant. Also, this same behavior was observed when $\epsilon \approx 0$.

Pugh hypothesizes that there may be several reasons for the solutions to converge to a nonconstant steady state. First, he suggests that the steady state solutions reached may not be in equilibrium. In other words, the steady state solutions, $u_{ss}(x)$, may not satisfy the steady state equation:

$$u(x)u_x(x) - \epsilon u_{xx}(x) = 0 \tag{1.9}$$

$$u_x(0) = u_x(1) = 0. \tag{1.10}$$

However, in [3], it is shown that all steady state solutions of Burgers' equation with homogeneous Neumann boundary conditions are constant. Thus, the first conjecture is incorrect. Second, the numerical solution $u^N(t, x)$ may not accurately reflect the steady state solution, $u_{ss}(x)$, as $t \to \infty$ due to discrepancies in the solutions between the Galerkin methods used. A final possibility for the numerical solutions converging to a nonconstant steady state is that there may exist $L_2$ functions, $u_{ss}(x)$, that satisfy a weak form of the steady state equation, but are not in the solution space. All of these results were limited to Neumann boundary conditions.

Smith [16] explored these same issues for Burgers' equation with Robin boundary conditions. He viewed Burgers' equation as a perturbation of the heat equation to model the flow of heat through a one dimensional rod with a conducting film at each end. Smith produced similar results to that of Pugh [14]. He investigated two specific problems where

the exact solution is known and found that the Galerkin and Galerkin/Conservation approximation methods both produce solutions that converged to the exact solution at a rate better than $(\frac{1}{N})^{\frac{1}{2}}$.

Smith also conducted experiments where the Robin boundary conditions are allowed to approach Dirichlet and Neumann boundary conditions by varying the physical properties of the rod and films. His results suggest that when the Robin boundary conditions are allowed to approach the Dirichlet and the Neumann boundary conditions, the solutions obtained approach the corresponding Dirichlet and Neumann solutions.

Burns and Kang considered a linear regulator problem ([4], [5]) for Burgers' equation. Using a linearization of the non-linear equation they compute feedback control laws which enhance the stability of the solution by requiring a certain fixed exponential decay rate, which is dependent upon the Reynolds number. The first paper, [4], involves a bounded input/unbounded output problem (See p.66 of [4]). The second paper, [5], involves an unbounded input/unbounded output problem. They use LQR theory to obtain linear feedback control laws and apply them to the non-linear Burgers' equation. They consider the following boundary control problem:

$$u_t(t,x) = \epsilon u_{xx}(t,x) + \alpha u(t,x), \qquad 0 < x < l, \qquad t > 0, \tag{1.11}$$

$$u(0,x) = u_o(x), \tag{1.12}$$

$$u(t,0) = 0, u(t,l) = v(t). \tag{1.13}$$

For this boundary control problem, Burns and Kang find a unique feedback functional gain $k_{\alpha,\epsilon}(\cdot) \in L^2(0,l)$ such that the feedback control has the form

$$v(t) = -K_{\alpha,\epsilon}u = -\int_0^l k_{\alpha,\epsilon}(s)u(t,s)ds. \tag{1.14}$$

If the feedback law $K_{\alpha,\epsilon}$ in equation (1.14) is applied to the boundary control problem for Burgers' equation, then the closed-loop controlled Burgers' equation becomes

$$u_t(t,x) = \epsilon u_{xx}(t,x) - u(t,x)u_x(t,x), \qquad 0 < x < l, t > 0, \tag{1.15}$$

$$u(0,x) = u_o(x), \tag{1.16}$$

and

$$u(t,0) = 0, \qquad u(t,l) = -\int_0^l k_{\alpha,\epsilon}(s)u(t,s)ds. \tag{1.17}$$

In ([4],[5]), they find that for a Reynolds number greater than 60, the open-loop solution creates a 'steep' gradient in finite time. This suggests that the convection term, $-u(t,x)u_x(t,x)$, dominates the diffusion term, $\epsilon u_{xx}(t,x)$, in Burgers' equation (1.1). Burns and Kang show the closed-loop non-linear system is stabilizable by linear feedback laws. The steep gradients in the finite element model of Burgers' equation are smoothed out by the feedback. To test the 'robustness' of the feedback control law, one experiment is performed. They obtain a specific functional gain from the control system for the Reynolds number, $Re = 60$, and apply it to the closed loop system at higher Reynolds numbers. Although the performance of the control is decreased, the system is still stabilizable and

the gradients are smoothed. Their numerical results also suggest convergence of the approximation scheme and provide insight into the possibility of using linear feedback laws for non-linear distributed parameter systems.

This paper will focus on numerical experiments with specific Robin boundary conditions. We consider LQR boundary control through these mixed boundary conditions. The linearized system is used to find an optimal LQR control. This control (dependent on Reynolds number) is then applied to the non-linear Burgers' equation subject to Robin boundary conditions. Various initial and boundary conditions will be used to test the effectiveness of the boundary control.

## 1.2 Boundary Conditions

There are several types of boundary conditions that may be considered for Burgers' equation. In [16], Smith focuses on Robin boundary conditions since they describe the realistic physics of a system modeled by Burgers' equation. Smith models a one dimensional rod of length $L$ with thermal conductivity $\kappa$. It is bounded by a thin film at each end with thicknesses $L_1$, $L_2$ and thermal conductivities $\kappa_1$, $\kappa_2$, respectively. The diffusivity of the rod, $\epsilon$, is related to the conductivity of the rod by $\epsilon = \frac{\kappa}{\rho c}$ where $\rho$ is the uniform density of the rod and $c$ is the specific heat of the rod.

The Robin boundary conditions for this problem (see [16]) are given by

$$\frac{\kappa_1}{L_1}u(t,0) - \kappa u_x(t,0) = \frac{\kappa_1}{L_1}q_o(t), \qquad \frac{\kappa_2}{L_2}u(t,L) + \kappa u_x(t,L) = \frac{\kappa_2}{L_2}q_1(t) \tag{1.18}$$

where $\kappa_1 > 0$, $\kappa_2 > 0$, $L_1 > 0$, $L_2 > 0$, and $\kappa > 0$ are real constants. Here, $q_o(t)$ and $q_1(t)$ are control inputs. Robin boundary conditions describe the forced change in time of $u$ at the boundaries. In [16] Smith views these conditions as describing the heating or cooling of a body through contact with a secondary body, which is in contact with a heat source.

If $\kappa_1$ and $\kappa_2$ are zero, then the resulting boundary conditions

$$\kappa u_x(t,0) = 0, \qquad \kappa u_x(t,L) = 0 \qquad 0 \le x \le L \tag{1.19}$$

are known as Neumann boundary conditions. If $\kappa_1$ and $\kappa_2$ are zero, then there is no conduction at the ends of the rod and, therefore, no heat flux out of the ends of the rod.

There is a third and final set of boundary conditions. These conditions result from setting $\kappa$ equal to zero in (1.18) so that the equations

$$u(t,0) = q_o(t), \qquad u(t,L) = q_1(t) \qquad 0 \le x \le L \tag{1.20}$$

become what are known as the Dirichlet boundary conditions.

The Neumann boundary conditions (1.19) and the Dirichlet boundary conditions (1.20) are special cases of Robin boundary conditions (1.18). In the three types of boundary conditions above, if either $q_o(t)$ or $q_1(t)$ is identically zero, then the boundary condition is known as homogeneous.

# Chapter 2

# Galerkin Approximations

## 2.1 Finite Dimensional Approximations

In this section we will start with an introduction to the finite dimensional approximation schemes we will employ. Precise formulation will follow in later sections. The unit interval $[0,1]$ is divided into $(N+1)$ subintervals $[x_i, x_{i+1}]$, each of length $h = \frac{1}{N+1}$, where $x_i = \frac{i}{N+1}$ for $i = 0, 1, 2, ..., N+1$. For each $i$, let $h_i(x)$ denote the linear basis function defined as follows:

$$h_o(x) = \begin{cases} -(N+1)(x - x_1), & \mathrm{x}_0 \leq \mathrm{x} \leq \mathrm{x}_1 \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

$$h_i(x) = \begin{cases} (N+1)(x - x_{i-1}), & \mathrm{x_{i-1} \leq x \leq x_i} \\ -(N+1)(x - x_{i+1}), & \mathrm{x_i \leq x \leq x_{i+1}} \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

for $1 \leq i \leq N$, and

$$h_{N+1}(x) = \begin{cases} (N+1)(x - x_N), & \mathrm{x_N \leq x \leq x_{N+1}} \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

These basis functions will be multiplied by time dependent 'weights' in order to approximate values of the exact solution to Burgers' equation, $u(t, x)$. The approximate solution can be written as

$$u^N(t, x) = \sum_{i=0}^{N+1} \alpha_i(t) h_i(x) \tag{2.4}$$

where each $\alpha_i(t)$ is a nodal unknown and $h_i(x)$ is the $i^{th}$ linear basis function defined on $[0, 1]$. These approximations may be used to develop an $N+2$ dimensional time-dependent system in the weak formulation of Galerkin approximations.

## 2.2 Weak Form of the Solution and Discretization

In this section, the weak formulation of the Galerkin approximation is developed for inhomogeneous Robin boundary conditions. Consider the homogeneous Burgers' equation ($f = 0$) defined on the interval $[0, 1]$, with initial condition and inhomogeneous Robin boundary data given by

$$u_t(t, x) + u(t, x) u_x(t, x) = \epsilon u_{xx}(t, x) \tag{2.5}$$

$$u(0,x) = \phi(x) \tag{2.6}$$

$$\frac{\kappa_1}{L_1}u(t,0) - \kappa u_x(t,0) = \frac{\kappa_1}{L_1}q_o(t), \qquad \frac{\kappa_2}{L_2}u(t,1) + \kappa u_x(t,1) = \frac{\kappa_2}{L_2}q_1(t) \tag{2.7}$$

where $\epsilon > 0$ is a diffusivity coefficient. If $u(t,x)$ solves the initial/boundary value problem given by (2.5), then

$$u_t(t,x) + u(t,x)u_x(t,x) - \epsilon u_{xx}(t,x) = 0 \tag{2.8}$$

for all $(t,x) \in \mathcal{R}^2$. Therefore, for $j = 0,1,2,\ldots,N+1$, multiplying on the right by the piecewise linear basis function $h_j(x)$ yields

$$[u_t(t,x) + u(t,x)u_x(t,x) - \epsilon u_{xx}(t,x)]h_j(x) = 0. \tag{2.9}$$

Integrating equation (2.9) on $[0,1]$ yields the equation

$$\int_0^1 [u_t(t,x) + u(t,x)u_x(t,x)]h_j(x)dx - \int_0^1 \epsilon u_{xx}(t,x)h_j(x)dx = 0. \tag{2.10}$$

Integrating the term $\int_0^1 \epsilon u_{xx}(t,x)h_j(x)dx$ by parts yields:

$$\int_0^1 [u_t(t,x) + u(t,x)u_x(t,x)]h_j(x)dx + \int_0^1 \epsilon u_x(t,x)h_j'(x)dx - \epsilon(h_j(x)u_x(t,x))|_0^1 = 0. \tag{2.11}$$

The approximate solution, $u^N(t,x)$, is now substituted for $u(t,x)$ to get the equation

$$\int_0^1 [u_t^N(t,x) + u^N(t,x)u_x^N(t,x)]h_j(x)dx + \int_0^1 \epsilon u_x^N(t,x)h_j'(x)dx - \epsilon(h_j(x)u_x^N(t,x))|_0^1 = 0. \tag{2.12}$$

Note from the approximate solution (2.4) that $u_t^N(t,x) = \sum_{i=0}^{N+1} \dot{\alpha}_i(t)h_i(x)$ and $u_x^N(t,x) = \sum_{i=0}^{N+1} \alpha_i(t)h_i'(x)$. Hence, substituting these terms into the approximating equation (2.12)

results in the following:

$$\int_0^1 \left[ \left( \sum_{i=0}^{N+1} \dot{\alpha}_i(t) h_i(x) \right) + \left( \sum_{i=0}^{N+1} \alpha_i(t) h_i(x) \right) \left( \sum_{k=0}^{N+1} \alpha_k(t) h_k'(x)) \right) \right] h_j(x) dx$$
$$= -\epsilon \int_0^1 \left( \sum_{i=0}^{N+1} \alpha_i(t) h_i'(x) \right) h_j'(x) dx + \epsilon \left( \sum_{i=0}^{N+1} \alpha_i(t) h_i'(x) \right) h_j(x)|_0^1. \tag{2.13}$$

Rearranging terms so that the linear basis functions are integrated against each other yields the equation

$$\sum_{i=0}^{N+1} \left[ \int_0^1 h_i(x) h_j(x) dx \right] \dot{\alpha}_i(t) + \sum_{i=0}^{N+1} \sum_{k=0}^{N+1} \left[ \int_0^1 h_i(x) h_k'(x) h_j(x) dx \right] \alpha_i(t) \alpha_k(t)$$
$$= -\epsilon \sum_{i=0}^{N+1} \left[ \int_0^1 h_i'(x) h_j'(x) dx \right] \alpha_i(t) + \epsilon \sum_{i=0}^{N+1} \left[ h_i'(1) h_j(1) - h_i'(0) h_j(0) \right] \alpha_i(t). \tag{2.14}$$

By the definition of the hat functions, $h_j(0) = 0$ for $j \neq 0$, and $h_j(1) = 0$ for $j \neq N+1$. Let $m_{ij} = \int_0^1 h_i(x) h_j(x) dx$ and $k_{ij} = -\int_0^1 h_i'(x) h_j'(x) dx$ for $i, j = 0, 1, 2, ...N+1$. With these terms substituted into equation (2.14), the following equivalent equation results:

$$\sum_{i=0}^{N+1} m_{ij} \dot{\alpha}_i(t) + \sum_{i=0}^{N+1} \sum_{k=0}^{N+1} \left[ \int_0^1 h_i(x) h_k'(x) h_j(x) dx \right] \alpha_i(t) \alpha_k(t)$$
$$= \epsilon \sum_{i=0}^{N+1} k_{ij} \alpha_i(t) + \epsilon \left[ (N+1)\alpha_0(t) - (N+1)\alpha_1(t) \right] h_j(0)$$
$$+ \epsilon \left[ -(N+1)\alpha_N(t) + (N+1)\alpha_{N+1}(t) \right] h_j(1). \tag{2.15}$$

The approximate solution evaluated at $x = 0$ and $x = 1$ yields the following boundary conditions:

$$\frac{\kappa_1}{L_1} q_0(t) = \frac{\kappa_1}{L_1} u^N(t, 0) - \kappa u_x^N(t, 0) =$$
$$\frac{\kappa_1}{L_1} \alpha_0^N(t) - \kappa [-(N+1)\alpha_0^N(t) + (N+1)\alpha_1^N(t)] \tag{2.16}$$

$$\frac{\kappa_2}{L_2}q_1(t) = \frac{\kappa_2}{L_2}u^N(t,1) + \kappa u_x^N(t,1) =$$

$$\frac{\kappa_2}{L_2}\alpha_{N+1}^N(t) + \kappa[-(N+1)\alpha_N^N(t) + (N+1)\alpha_{N+1}^N(t)]. \tag{2.17}$$

These equations may be written as

$$\frac{\kappa_1}{\kappa L_1}[q_0(t) - \alpha_0^N] = [(N+1)\alpha_0^N(t) - (N+1)\alpha_1^N(t)]$$

$$\frac{\kappa_2}{\kappa L_2}[q_1(t) - \alpha_{N+1}^N] = [-(N+1)\alpha_N^N(t) + (N+1)\alpha_{N+1}^N(t)]. \tag{2.18}$$

When substituted into the weak formulation, equation (2.15) becomes

$$\sum_{i=0}^{N+1} m_{ij}\dot{\alpha}_i(t) + \sum_{i=0}^{N+1}\sum_{k=0}^{N+1}\left[\int_0^1 h_i(x)h_k'(x)h_j(x)dx\right]\alpha_i(t)\alpha_k(t)$$

$$= \epsilon\sum_{i=0}^{N+1} k_{ij}\alpha_i(t) + \epsilon\{\frac{\kappa_1}{\kappa L_1}[q_0(t) - \alpha_0^N(t)]\}h_j(0) + \epsilon\{\frac{\kappa_2}{\kappa L_2}[q_1(t) - \alpha_{N+1}^N(t)]\}h_j(1). \tag{2.19}$$

Since (2.19) must be valid for all $h_j(x)$, $j = 0, 1, 2, ..., N+1$, we obtain the system

$$\left[M^N\right]\begin{bmatrix}\dot{\alpha}_0(t)\\ \dot{\alpha}_1(t)\\ \vdots\\ \dot{\alpha}_{N+1}(t)\end{bmatrix} = \epsilon\left[K^N\right]\begin{bmatrix}\alpha_0(t)\\ \alpha_1(t)\\ \vdots\\ \alpha_{N+1}(t)\end{bmatrix} + \epsilon[G^N]\begin{bmatrix}q_0(t)\\ q_1(t)\end{bmatrix} - F_E^N(\alpha^N(t)). \tag{2.20}$$

The matrix $\left[M^N\right] = [m_{ij}]$ is commonly known as the mass matrix and has the form

$$[M^N] = \frac{1}{6(N+1)} \begin{bmatrix} 2 & 1 & 0 & 0 & \dots & 0 \\ 1 & 4 & 1 & 0 & \dots & 0 \\ 0 & 1 & 4 & 1 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & & & 1 & 4 & 1 \\ 0 & & \dots & 0 & 1 & 2 \end{bmatrix}_{(N+2)\times(N+2)} . \tag{2.21}$$

The matrix $\left[K^N\right]$ is known as the stiffness matrix and for this system has the form

$$[K^N] = (N+1) \begin{bmatrix} -1-\frac{\kappa_1}{\kappa L_1(N+1)} & 1 & 0 & 0 & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & & & 1 & -2 & 1 \\ 0 & & \dots & 0 & 1 & -1-\frac{\kappa_2}{\kappa L_2(N+1)} \end{bmatrix}_{(N+2)\times(N+2)} . \tag{2.22}$$

The matrix $\left[G^N\right]$ is multiplied on the right by the vector of boundary functions $\begin{bmatrix} q_0(t) \\ q_1(t) \end{bmatrix}$, and has the form

$$\left[G^N\right] = \begin{bmatrix} \frac{\kappa_1}{\kappa L_1} & 0 \\ 0 & \vdots \\ \vdots & 0 \\ 0 & \frac{\kappa_2}{\kappa L_2} \end{bmatrix}_{(N+2\times 2)} .$$

The term

$$\sum_{i=0}^{N+1}\sum_{k=0}^{N+1}\left[\int_0^1 h_i(x)h_k'(x)h_j(x)dx\right]\alpha_i(t)\alpha_k(t),$$

in (2.19) becomes the non-linear term

$$F_E^N(\alpha(t)) = \frac{1}{6}\begin{bmatrix} -2(\alpha_0(t))^2 + \alpha_0(t)\alpha_1(t) + (\alpha_1(t))^2 \\ -(\alpha_0(t))^2 - \alpha_0(t)\alpha_1(t) + \alpha_1(t)\alpha_2(t) + (\alpha_2(t))^2 \\ \vdots \\ -(\alpha_{N-1}(t))^2 - \alpha_{N-1}(t)\alpha_N(t) + \alpha_N(t)\alpha_{N+1}(t) + (\alpha_{N+1}(t))^2 \\ -(\alpha_N(t))^2 + \alpha_N(t)\alpha_{N+1}(t) + 2(\alpha_{N+1}(t))^2 \end{bmatrix}_{(N+2)\times 1} ,$$

and approximates the non-linear term in Burgers' equation, $u(t,x)u_x(t,x)$. This term complicates the computational solution for large values of N, [14].

The mass matrix $\left[M^N\right]$ is known to be nonsingular. Therefore, the matrix equation may be written as

$$\dot{\alpha}^N(t) = [A_\epsilon^N]\alpha^N(t) + [B_\epsilon^N]q(t) - \mathcal{F}_E^N(\alpha(t)) \tag{2.23}$$

where

$$[A_\epsilon^N] = \epsilon[M^N]^{-1}[K^N],$$

$$[B_\epsilon^N] = \epsilon[M^N]^{-1}G^N,$$

$$\mathcal{F}_E^N(\alpha(t)) = [M^N]^{-1}F_E^N(\alpha(t)),$$

and

$$q(t) = \begin{bmatrix} q_0(t) \\ \\ q_1(t) \end{bmatrix}.$$

## 2.3 Initial Condition and Forcing Function

Let

$$\phi^N(x) = \sum_{i=0}^{N+1} \phi_i h_i(x) \approx \phi(x)$$

denote the best approximation of the initial data $\phi(x)$. It is well known (see [10]) that

$$< \phi^N(\cdot) - \phi(\cdot), h_j(\cdot) >_{\mathcal{L}_\in(0,1)} = 0$$

for each $j = 0, 1, 2, \ldots, N+1$. Hence, for all $j = 0, 1, 2, \ldots, N+1$

$$\int_0^1 \phi^N(x)h_j(x)dx = \int_0^1 \phi(x)h_j(x)dx, \tag{2.24}$$

or equivalently,

$$\int_0^1 \left(\sum_{i=0}^{N+1} \phi_i h_i(x)\right) h_j(x)dx = \int_0^1 \phi(x)h_j(x)dx. \tag{2.25}$$

Therefore,

$$[M^N]\begin{bmatrix}\phi_0\\ \phi_1\\ \vdots\\ \phi_{N+1}\end{bmatrix}=\begin{bmatrix}<\phi(\cdot),h_0(\cdot)>\\ <\phi(\cdot),h_1(\cdot)>\\ \vdots\\ <\phi(\cdot),h_{N+1}(\cdot)>\end{bmatrix} \tag{2.26}$$

and

$$\begin{bmatrix}\phi_0\\ \phi_1\\ \vdots\\ \phi_{N+1}\end{bmatrix}=[M^N]^{-1}\begin{bmatrix}<\phi(\cdot),h_0(\cdot)>\\ <\phi(\cdot),h_1(\cdot)>\\ \vdots\\ <\phi(\cdot),h_{N+1}(\cdot)>\end{bmatrix} \tag{2.27}$$

Since $u^N(0,x)=\sum_{i=0}^{N+1}\alpha_i(0)h_i(x)=\phi^N(x)=\sum_{i=0}^{N+1}\phi_i h_i(x)$, we obtain the initial condition

$$\begin{bmatrix}\alpha_0(0)\\ \alpha_1(0)\\ \vdots\\ \alpha_{N+1}(0)\end{bmatrix}=\left[M^N\right]^{-1}\begin{bmatrix}<\phi(\cdot),h_0(\cdot)>\\ <\phi(\cdot),h_1(\cdot)\\ \vdots\\ <\phi(\cdot),h_{N+1}(\cdot)>\end{bmatrix}=\begin{bmatrix}\phi_0\\ \phi_1\\ \vdots\\ \phi_{N+1}\end{bmatrix}=\phi_o^N. \tag{2.28}$$

The system becomes the (N+2) dimensional initial value problem

$$\dot{\alpha}^N(t)=[A_\epsilon^N]\alpha^N(t)+[B_\epsilon^N]q(t)-\mathcal{F}_E^N(\alpha(t)) \tag{2.29}$$

$$\alpha^N(0)=\phi_o^N. \tag{2.30}$$

Note that if the Burgers' equation is not homogeneous ($f(t,x) \neq 0$), then equation (2.29) is modified by adding the term

$$f^N(t) = \left[M^N\right]^{-1} \begin{bmatrix} < f(t,x), h_0(x) > \\ < f(t,x), h_1(x) \\ \vdots \\ < f(t,x), h_{N+1}(x) > \end{bmatrix} \tag{2.31}$$

to the right hand side.

## 2.4 Galerkin/Conservation Form

The weak formulation for Galerkin/Conservation form of Burgers' equation, ([10],[14]) will be performed following the same format for the weak formulation of the Galerkin approximation. Consider the Galerkin/Conservation form of the homogeneous Burgers' equation given by

$$u_t(t,x) + \frac{1}{2}(u^2(t,x))_x = \epsilon u_{xx}(t,x). \tag{2.32}$$

Following the method in [14], for $j = 0, 1, 2, \ldots, N+1$, multiplying by $h_j(x)$ and integrating we get

$$\int_0^1 [u_t(t,x) + \frac{1}{2}(u^2(t,x))_x] h_j(x) dx = \int_0^1 \epsilon u_{xx}(t,x) h_j(x) dx. \tag{2.33}$$

Integrating the term $\int_0^1 \epsilon u_{xx}(t,x) h_j(x) dx$ by parts yields

$$\int_0^1 [u_t(t,x) + \frac{1}{2}(u^2(t,x))_x] h_j(x) dx = -\int_0^1 \epsilon u_x(t,x) h_j'(x) dx + \epsilon (u_x(t,x)) h_j(x)|_0^1. \tag{2.34}$$

In addition to the approximate solution

$$u^N(t,x) = \sum_{i=0}^{N+1} \alpha_i(t) h_i(x), \tag{2.35}$$

for $u(t,x)$, the following approximate solution for $u^2(t,x)$ is introduced:

$$[u^N(t,x)]^2 = \sum_{i=0}^{N+1} (\alpha_i(t))^2 h_i(x). \tag{2.36}$$

Substituting (2.36) into equation (2.34) for $(u^2(t,x))_x$ gives

$$\int_0^1 \{u_t^N(t,x) + \frac{1}{2}[(u^N(t,x))^2]_x\} h_j(x)dx = -\epsilon \int_0^1 u_x^N(t,x) h_j'(x)dx + \epsilon (u_x^N(t,x)) h_j(x)|_0^1. \tag{2.37}$$

Note that $[(u^N(t,x))^2]_x = \sum_{i=0}^{N+1} (\alpha_i(t))^2 h_i'(x)$. Also, as before $u_t^N(t,x) = \sum_{i=0}^{N+1} \dot{\alpha}_i(t) h_i(x)$ and $u_x^N(t,x) = \sum_{i=0}^{N+1} \alpha_i(t) h_i'(x)$. Making the appropriate substitutions into equation (2.37) results in the following discretized equation:

$$\begin{aligned} &\int_0^1 \{\sum_{i=0}^{N+1} \dot{\alpha}_i(t) h_i(x) + \frac{1}{2} \sum_{i=0}^{N+1} (\alpha_i(t))^2 h_i'(x)\} h_j(x)dx = \\ &-\epsilon \int_0^1 \sum_{i=0}^{N+1} \alpha_i(t) h_i'(x) h_j'(x)dx + \epsilon \sum_{i=0}^{N+1} \alpha_i(t) h_i'(x) h_j(x)|_0^1. \end{aligned} \tag{2.38}$$

Rearranging terms in (2.38) as in Section 2.2, yields

$$\begin{aligned} &\sum_{i=0}^{N+1} \left[\int_0^1 h_i(x) h_j(x)dx\right] \dot{\alpha}_i(t) + \frac{1}{2} \sum_{i=0}^{N+1} \left[\int_0^1 h_i'(x) h_j(x)dx\right] (\alpha_i(t))^2 \\ &= \epsilon \sum_{i=0}^{N+1} \left[-\int_0^1 h_i'(x) h_j'(x)dx\right] \alpha_i(t) + \epsilon \sum_{i=0}^{N+1} \left[h_i'(1) h_j(1) - h_i'(0) h_j(0)\right] \alpha_i(t). \end{aligned} \tag{2.39}$$

Performing the same boundary condition substitution and integral substitutions used in Section 2.2, equation (2.39) becomes

$$\sum_{i=0}^{N+1} m_{ij}\dot{\alpha}_i(t) + \sum_{i=0}^{N+1} d_{ij}(\dot{\alpha}_i(t))^2$$

$$= \epsilon \sum_{i=0}^{N+1} k_{ij}\alpha_i(t) + \epsilon\left[(N+1)\alpha_0(t) - (N+1)\alpha_1(t)\right] h_j(0)$$

$$+\epsilon\left[-(N+1)\alpha_N(t) + (N+1)\alpha_{N+1}(t)\right] h_j(1) \tag{2.40}$$

where $d_{ij} = \frac{1}{2}\int_0^1 h_i'(x)h_j(x)dx$ and $m_{ij}$, $k_{ij}$ are the same as in Section 2.2. In matrix form, equation (2.40) is equivalent to

$$\left[M^N\right]\begin{bmatrix} \dot{\alpha}_0(t) \\ \dot{\alpha}_1(t) \\ \vdots \\ \dot{\alpha}_{N+1}(t) \end{bmatrix} + \left[D^N\right]\begin{bmatrix} (\alpha_0(t))^2 \\ (\alpha_1(t))^2 \\ \vdots \\ (\alpha_{N+1}(t))^2 \end{bmatrix} = \epsilon\left[K^N\right]\begin{bmatrix} \alpha_0(t) \\ \alpha_1(t) \\ \vdots \\ \alpha_{N+1}(t) \end{bmatrix} + \epsilon[G^N]\begin{bmatrix} q_0(t) \\ q_1(t) \end{bmatrix}. \tag{2.41}$$

The matrix $\left[D^N\right]$ has the form

$$\left[D^N\right] = [d_{ij}] = \frac{1}{4}\begin{bmatrix} -1 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 0 & -1 & 0 & 1 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & & & -1 & 0 & 1 \\ 0 & & \dots & 0 & -1 & 1 \end{bmatrix}_{(N+2)\times(N+2)}.$$

The matrices $\left[M^N\right]$, $\left[K^N\right]$, and $[G^N]$ are the same as in Section 2.2. Equation (2.41) may be written as

$$\left[M^N\right]\begin{bmatrix}\dot{\alpha}_0(t)\\ \dot{\alpha}_1(t)\\ \vdots\\ \dot{\alpha}_{N+1}(t)\end{bmatrix} + F_D^N(\alpha(t)) = \epsilon\left[K^N\right]\begin{bmatrix}\alpha_0(t)\\ \alpha_1(t)\\ \vdots\\ \alpha_{N+1}(t)\end{bmatrix} + \epsilon[G^N]\begin{bmatrix}q_0(t)\\ q_1(t)\end{bmatrix} \tag{2.42}$$

where the term $F_D^N(\alpha(t))$ in equation (2.42) comes from the matrix product

$$\left[D^N\right]\begin{bmatrix}(\alpha_0(t))^2\\ (\alpha_1(t))^2\\ \vdots\\ (\alpha_{N+1}(t))^2\end{bmatrix}. \tag{2.43}$$

Carrying out this multiplication yields

$$F_D^N(\alpha(t)) = \frac{1}{4}\begin{bmatrix}(\alpha_1(t))^2-(\alpha_0(t))^2\\ (\alpha_2(t))^2-(\alpha_0(t))^2\\ (\alpha_3(t))^2-(\alpha_1(t))^2\\ \vdots\\ (\alpha_{N+1}(t))^2-(\alpha_{N-1}(t))^2\\ (\alpha_{N+1}(t))^2-(\alpha_N(t))^2\end{bmatrix} \tag{2.44}$$

which is related to the non-linear term $F_E^N(\alpha(t))$ in the Galerkin approximation. Again, $\left[M^N\right]$ is invertible so that equation (2.42) may be multiplied on the left by $\left[M^N\right]^{-1}$ to

obtain the system

$$\dot{\alpha}^N(t) = [A_\epsilon^N]\alpha^N(t) + [B_\epsilon^N]q(t) - \mathcal{F}_D^N(\alpha(t)) \tag{2.45}$$

where $[A_\epsilon^N]$ and $[B_\epsilon^N]$ are defined as before and $\mathcal{F}_D^N(\alpha(t)) = \left[M^N\right]^{-1} F_D^N(\alpha(t))$. The initial condition, $\alpha^N(0) = \phi_o^N$, for this problem is equivalent to the one generated in Section 2.2. Also, if the forcing function $f(t,x)$ is not identically zero, then $f^N(t)$ (defined in section 2.2) must be added to the right hand side of equation (2.45).

The initial value problems for the Galerkin and Galerkin/Conservation methods were initially set up in MATLAB by Marrekchi [12] and duplicated by Pugh [14], both for homogeneous Neumann boundary conditions, $\frac{\kappa_1}{L_1} = \frac{\kappa_2}{L_2} = 0$. The system was solved using the MATLAB ordinary differential equation solver ODE45 for various conditions. Smith, in his paper [16], repeated this process for Robin boundary conditions. The inhomogeneous Robin boundary system is completely regenerated in this paper and coded in MATLAB for the purpose of conducting control experiments.

# Chapter 3

# Control Experiments

In this chapter, boundary control experiments are performed on Burgers' equation with Robin boundary conditions. First, Example (3.2.1) in [16] is repeated to test the convergence of the numerical schemes to the exact solution for a specific initial boundary value problem. Then, closed loop and open loop systems are simulated with several initial functions for a specific rod and films. The gain from these experiments is saved and used in other experiments. Next, Example (3.5.1) from [16] is used to run control experiments on boundary conditions that approach Neumann and Dirichlet boundary conditions. Also, experiments are performed by varying a weighting constant which affects the amount of control imposed on a system. The final set of tests are analogous to those run by Burns and Kang in [5]. For a copper rod with aluminum films, $\epsilon = 1.14$, the optimal feedback gain matrix, $K_{1.14}$ is generated. It is then used to control closed loop systems whose $\epsilon$ is less

than 1.14. Comparisons are made between the closed loop systems with optimal control and those with the control, $u_{1.14}(t) = -K_{1.14}\alpha(t)$. Similarly, for various Reynolds numbers, a comparison is made between closed loop systems with optimal control and closed loop systems with the control $u_{.939}(t) = -K_{.939}\alpha(t)$.

## 3.1 Convergence of the Galerkin and Galerkin/Conservation Methods

Before turning to the control problem, we tested the MATLAB code for both the Galerkin and the Galerkin/Conservation methods. The purpose of this exercise is to compare our numerical results to those produced by Marrekchi [12], Pugh [14], and Smith [16].

In order to measure the numerical error introduced by the approximation scheme, the exact solution of the following initial/boundary value problem is computed by choosing $f(t,x)$ appropriately. We solve the system

$$\begin{gathered} u_t(t,x) + u(t,x)u_x(t,x) - \epsilon u_{xx}(t,x) = f(t,x) \\ u(0,x) = Ax^2 + Bx + C - \sin^2(\tfrac{\pi}{100}x) \\ u(t,0) - \tfrac{\kappa L_1}{\kappa_1}u_x(t,0) = 0, \qquad u(t,100) + \tfrac{\kappa L_2}{\kappa_2}u_x(t,100) = 0 \end{gathered} \tag{3.1}$$

by setting

$$u(t,x) = e^{-\beta t}[Ax^2 + Bx + C - \sin^2(\frac{\pi}{100}x)] \tag{3.2}$$

and substituting $u(t,x)$ into the equation and boundary conditions to compute $f(t,x)$ along

with A, B, and C. This is done in Section 3.2 of [16]. Here, we find that

$$f(t,x) = -(\beta[Ax^2 + Bx + C + \sin^2(\frac{\pi}{100}x)] + \epsilon[2A + 2\frac{\pi^2}{100^2}(\cos^2(\frac{\pi}{100}x) - \sin^2(\frac{\pi}{100}x))])e^{-\beta t}$$
$$+[Ax^2 + Bx + C + \sin^2(\frac{\pi}{100}x)][2Ax + B + 2\frac{\pi}{100}\sin(\frac{\pi}{100}x)\cos(\frac{\pi}{100}x)]e^{-2\beta t},$$

$$A = \frac{100\kappa_1\kappa_2 + \kappa\kappa_2 L_1 + \kappa\kappa_1 L_2}{250,000},$$
$$B = -\frac{100^2\kappa_1\kappa_2 + 200\kappa\kappa_1 L_2}{250,000},$$

and

$$C = -\frac{100^2\kappa\kappa_2 L_1 + 200\kappa^2 L_1 L_2}{250,000}.$$

Smith [16] shows that the numerical solution converges to the exact solution proportionally to $(\frac{1}{N})^{\frac{1}{2}}$. The rod is chosen to be one meter of copper and the films are each ten centimeters of aluminum. The Reynolds number ($Re = \frac{1}{\epsilon}$) for this experiment is .8805 and $\beta = .3$. Figures 3.1.1, 3.1.2, and 3.1.3, show the convergence of the Galerkin numerical solutions to the exact solution (3.2) for the initial/boundary value problem (3.1). Similarly, Figures 3.1.4, 3.1.5, and 3.1.6, show the convergence of the Galerkin/Conservation numerical solutions to the exact solution (3.2) for the initial/boundary value problem (3.1).

Clearly, both methods converge to the exact solution. The results for this example are the same as those in [16]. The Galerkin/Conservation method is computationally faster and is used throughout the rest of the experiments with boundary control.

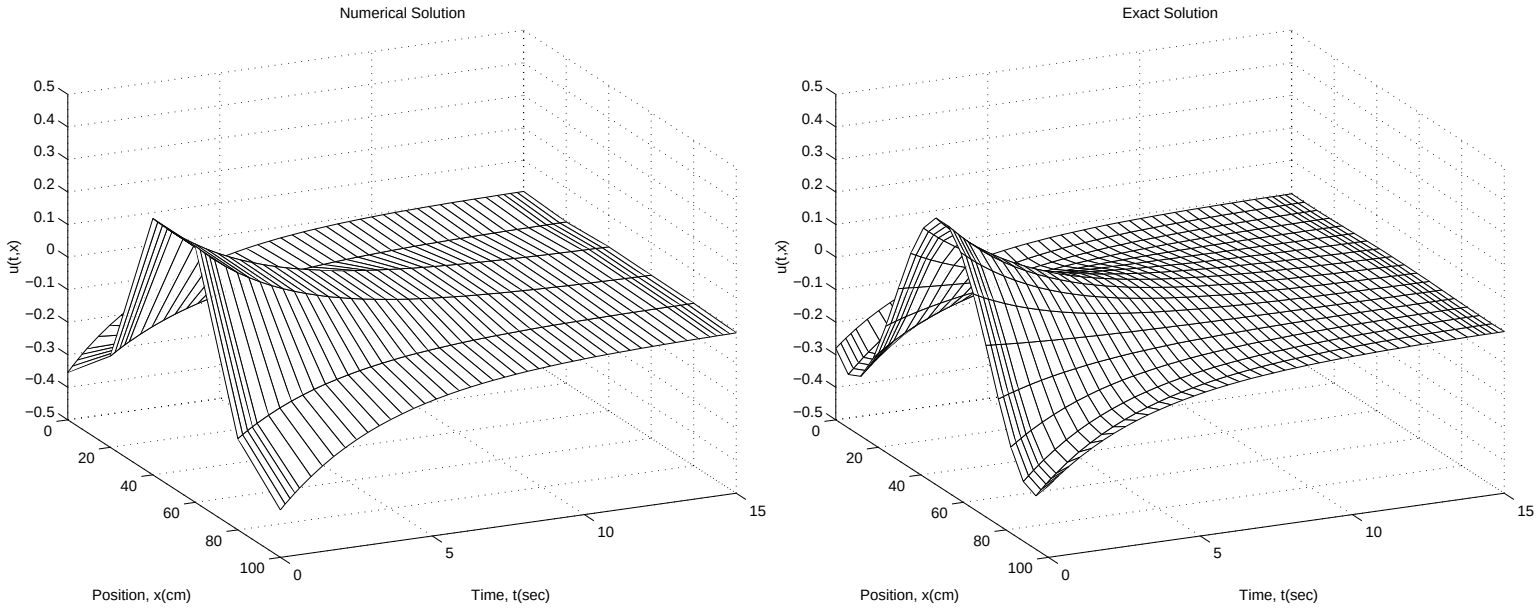


Figure 3.1.1: The Galerkin solution for N = 4 versus the exact solution.

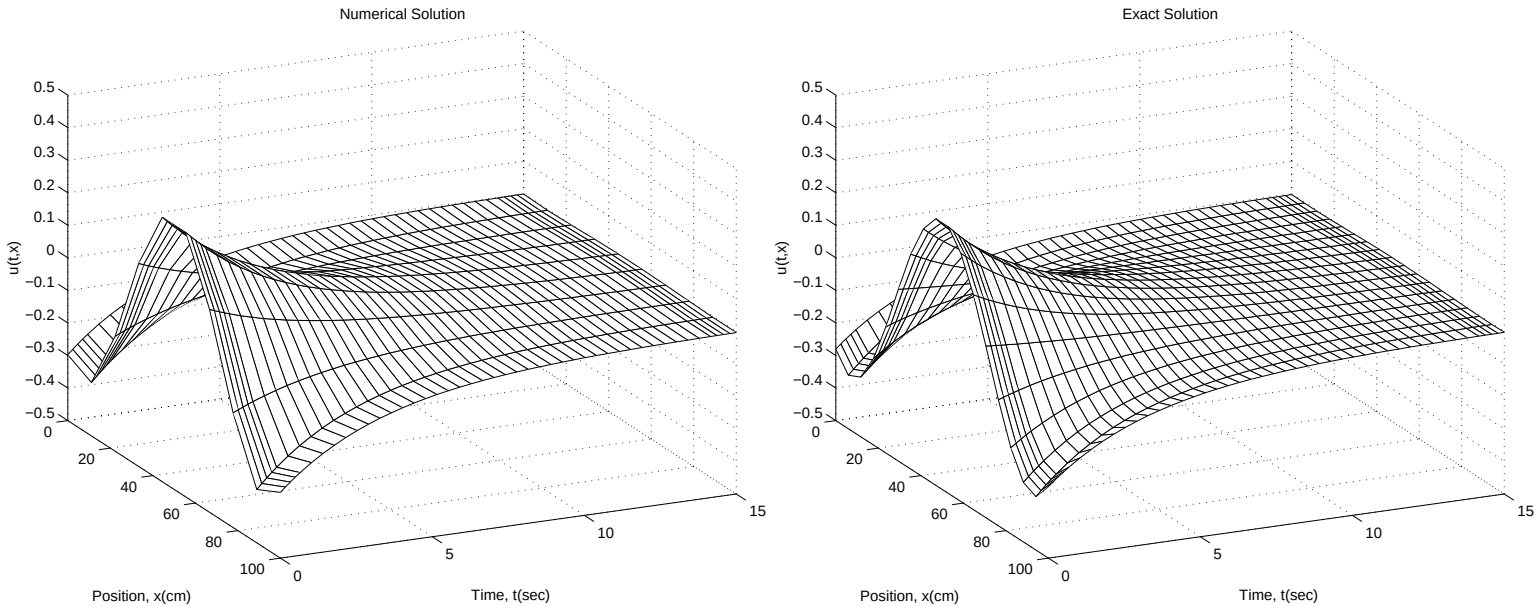


Figure 3.1.2: The Galerkin solution for N = 8 versus the exact solution.

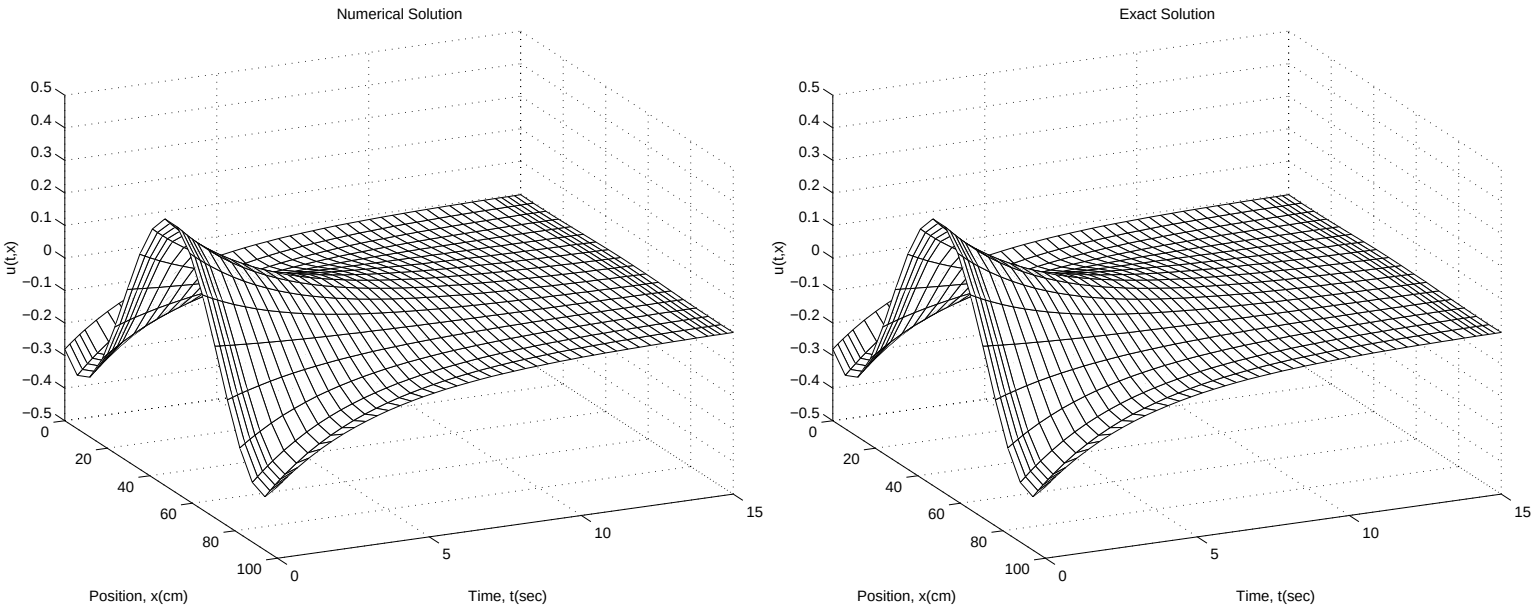


Figure 3.1.3: The Galerkin solution for N = 16 versus the exact solution.

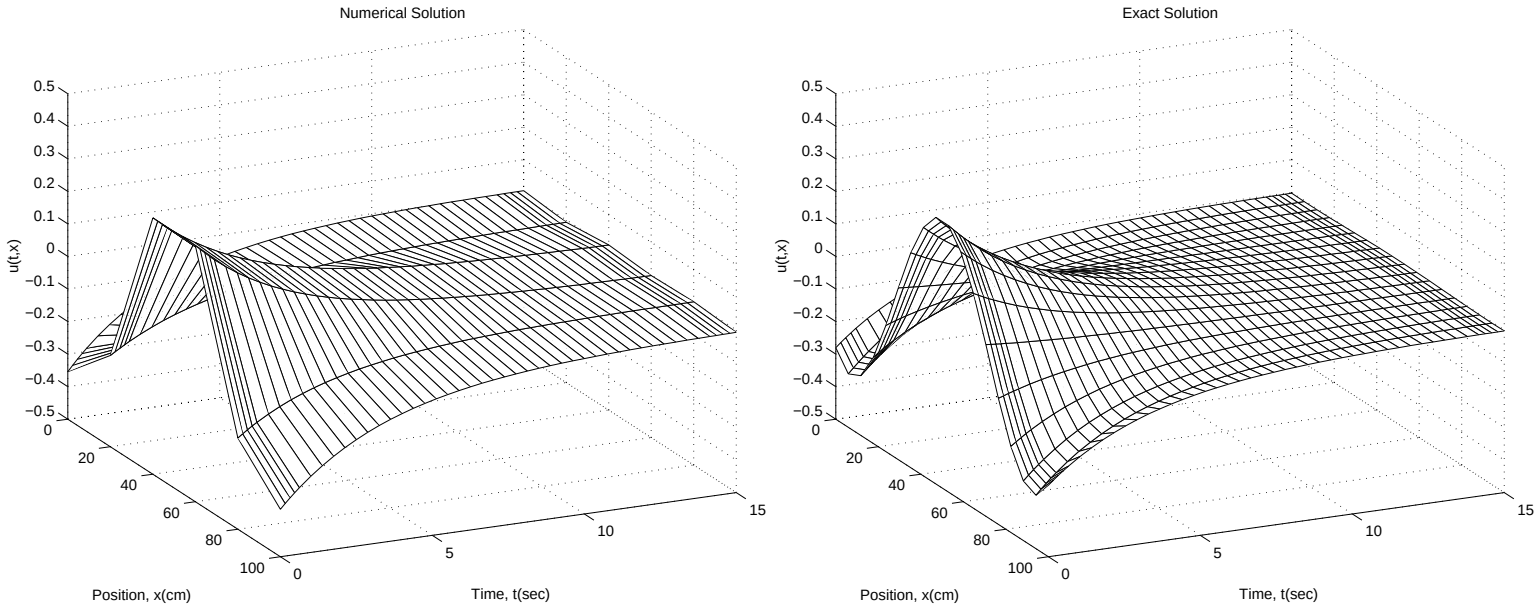


Figure 3.1.4: The Galerkin/Conservation solution for N = 4 versus the exact solution.

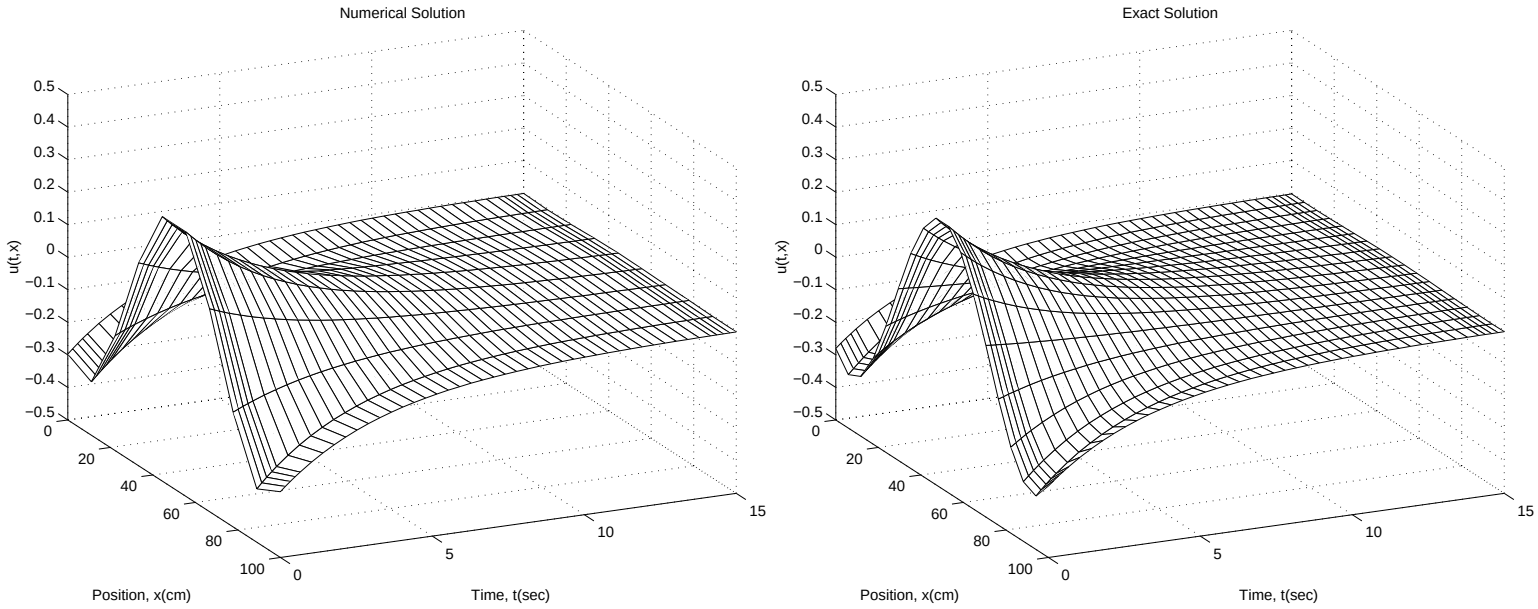


Figure 3.1.5: The Galerkin/Conservation solution for N = 8 versus the exact solution.

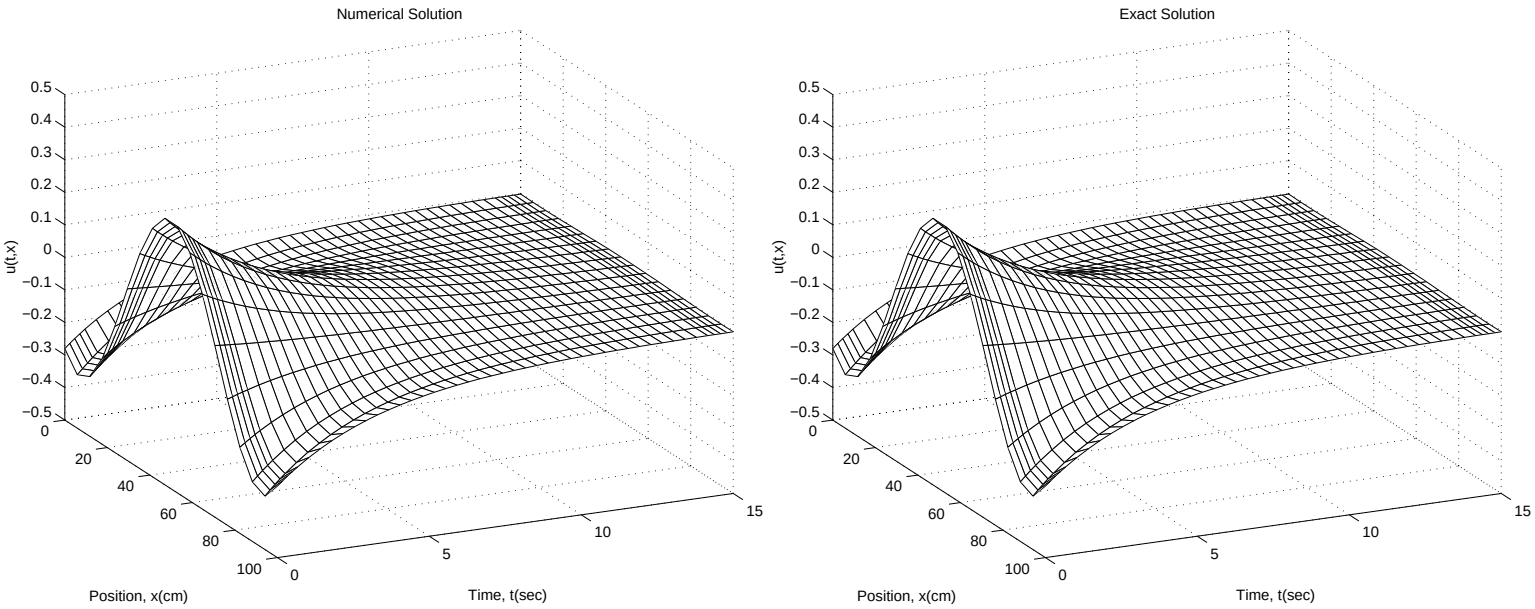


Figure 3.1.6: The Galerkin/Conservation solution for N = 16 versus the exact solution.

## 3.2 LQR Control Problem

In this section, we use the Galerkin/Conservation method to compute LQR controllers. These controllers are used as boundary control on the system (2.45). Open loop and closed loop responses are generated for various initial conditions, Reynolds numbers, and mixed boundary conditions. These experiments are used to analyze the decay of the solutions toward the zero steady state.

Linear-Quadratic Regulator (LQR) design for finding a control is discussed in [13]. This design is used by Olds [13] to test the stability of the steady-state LQR solution for a two-dimensional airfoil. Consider the system

$$\dot{x}(t) = A^N x(t) + B^N u(t), \qquad x(0) = x_o \tag{3.3}$$

where $x(t)$ is the state vector and $u(t)$ is the control vector. The goal of LQR theory is to find a control $u^*(t)$ that minimizes the weighted performance measure

$$\min_u J = \int_0^\infty \{\langle Q^N x(t), x(t)\rangle + \langle Ru(t), u(t)\rangle\} dt \tag{3.4}$$

where $x(t)$ is the solution of (3.3). Here $Q^N = [Q^N]^T \geq 0$ and $R = R^T > 0$ are weighting matrices. It is well known that if an optimal control $u^*(t)$ exists, it has the form

$$u^*(t) = -K_c x(t), \tag{3.5}$$

where $K_c$ is a constant gain matrix. Moreover, the closed loop system

$$\dot{x}(t) = A^N x(t) - B^N K_c x(t) = (A^N - B^N K_c) x(t) \tag{3.6}$$

is stable.

A system is stabilizable if there exists a state feedback control, $u = -K_c x$, such that the closed loop system (3.6) is exponentially stable. This means that there exists an $M > 0$ and a $\gamma > 0$ such that if $x(t)$ is the solution to the closed loop system with $x(0) = x_o$, then

$$\| x(t) \| \leq M e^{-\gamma t} \| x_o \| . \tag{3.7}$$

If such a $K_c$ exists, then (3.3) is said to be stabilizable. If the system is stabilizable, then the LQR problem has a solution.

**Existence and Stability of the Steady-State LQR Solution**: Given the LQR problem with $R^N > 0$, and $Q^N = C^T C$, where the pair $(A^N, C)$ is detectable and the pair $(A^N, B^N)$ is stabilizable, it follows that a solution to the steady-state LQR problem exists. In particular, there exists a unique positive semidefinite solution, $\overline{P}$, to the algebraic Riccati equation

$$0 = [A^N]^T P + P[A^N] + Q - P[B^N][R^N]^{-1}[B^N]^T P, \tag{3.8}$$

and if

$$K_c = [R^N]^{-1}[B^N]^T \overline{P}, \tag{3.9}$$

then the closed loop system (3.6) is asymptotically stable.

## 3.3 Boundary Control of the Nonlinear System

As in ([4], [5]), LQR theory is used on the non-linear Burgers' equation in order study the effectiveness of the optimal control. The focus of this research is boundary control with

Robin boundary conditions.

The IBVP which is considered for these experiments is the following:

$$\begin{gathered} u_t(t,x) + u(t,x)u_x(t,x) = \epsilon u_{xx}(t,x) \\ u(0,x) = \phi(x) \\ u(t,0) - \tfrac{\kappa L_1}{\kappa_1} u_x(t,0) = q_o(t), \qquad u(t,L) + \tfrac{\kappa L_2}{\kappa_2} u_x(t,L) = q_1(t) \end{gathered} \tag{3.10}$$

where $\kappa_1 = \kappa_2$, $L_1 = L_2$, and $\epsilon = \frac{\kappa}{\rho c}$. The functions $q_o(t)$ and $q_1(t)$ are the control functions. In the following examples, the Galerkin/Conservation system is used for the control experiments. We view the non-linear terms $\mathcal{F}_D^N(\alpha(t))$ as unmodeled dynamics. The following time dependent ODE system is considered:

$$\dot{\alpha}^N(t) = [A_\epsilon^N]\alpha^N(t) + [B_\epsilon^N]q(t) - \mathcal{F}_D^N(\alpha(t)) \tag{3.11}$$

$$\alpha^N(0) = \phi_o^N \tag{3.12}$$

where $[A_\epsilon^N]$ and $[B_\epsilon^N]$ depend upon $\epsilon$. The linearization (about $u = 0$) of equation (3.11) is

$$\dot{\alpha}^N(t) = [A_\epsilon^N]\alpha^N(t) + [B_\epsilon^N]q(t) \tag{3.13}$$

and it is used to find the optimal feedback gain matrix $K_\epsilon$, such that the feedback law

$$q^*(t) = -K_\epsilon \alpha^N(t) \tag{3.14}$$

minimizes the weighted cost function

$$J(q) = \int_0^\infty \{\langle Q^N \alpha^N(t), \alpha^N(t)\rangle + \langle Rq(t), q(t)\rangle\} dt. \tag{3.15}$$

The matrix $[Q^N]$ is chosen to be the mass matrix, $[M^N]$, the matrix $R$ is chosen to be

$$R = \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix},$$

where $r > 0$ is a real constant. The vector

$$q(t) = \begin{bmatrix} q_o(t) \\ q_1(t) \end{bmatrix} \tag{3.16}$$

is the control vector. For an open loop system, there is no control imposed on the system, thus $q_o(t) = q_1(t) = 0$. The LQR control toolbox in MATLAB is used to calculate the optimal feedback gain matrix, $K_\epsilon$, for a selected, weighted ($r > 0$) cost function.

The form of the open loop systems is

$$\dot{\alpha}^N(t) = [A_\epsilon^N]\alpha^N(t) - \mathcal{F}_D^N(\alpha(t)) \tag{3.17}$$

and the form of the closed loop systems is

$$\begin{aligned} \dot{\alpha}(t)^N(t) &= A^N\alpha(t) - B^N K_\epsilon \alpha(t) - \mathcal{F}_D^N(\alpha(t)) \\ &= (A^N - B^N K_\epsilon)\alpha(t) - \mathcal{F}_D^N(\alpha(t)) \\ &= \mathcal{A}_\epsilon^N \alpha(t) - \mathcal{F}_D^N(\alpha(t)) \end{aligned} \tag{3.18}$$

The Galerkin/Conservation form of the numerical approximations for (3.17) and (3.18) is programmed in MATLAB in order to run the open loop and closed loop simulations. The parameters chosen are obtained from Smith's Table 1.2.1, [16].

### 3.3.1 Response to Various Initial Conditions

The purpose of the following experiment is to investigate performance for fixed boundary conditions and various initial conditions. The weighting constant, $r$, for the LQR cost function (3.4) is chosen to be .005. The gain matrix for a ten centimeter copper rod with one centimeter aluminum films at each end is calculated by the $LQR.M$ function in MATLAB. It is denoted as $K_{1.14}$, since $\epsilon \equiv \frac{\kappa}{\rho c} = \frac{.93}{(8.9)(.092)} = 1.14$ for a copper rod. This gain is used to run closed loop simulations where the controls are at the boundaries of the rod.

The optimal controllers have the form

$$q_o(t) = -\int_0^1 k_\epsilon^o(s)u(t,s)ds \tag{3.19}$$

$$q_o(t) = -\int_0^1 k_\epsilon^1(s)u(t,s)ds \tag{3.20}$$

where $k_\epsilon^o(s)$ and $k_\epsilon^1(s)$ are the functional gains. We consider the boundary control problem given by the Robin boundary conditions

$$\begin{aligned} \frac{\kappa_1}{L_1}u(t,0) - \kappa u_x(t,0) &= \frac{\kappa_1}{L_1}q_o(t) \\ \frac{\kappa_2}{L_2}u(t,L) + \kappa u_x(t,L) &= \frac{\kappa_2}{L_2}q_1(t) \end{aligned} \tag{3.21}$$

with $\kappa_1 = \kappa_2 = .55$, $L_1 = L_2 = 1cm$, and $\kappa = .93$ (see Table 1.2.1 in [16]).

Figure 3.3.1 shows the functional gains for a ten centimeter copper rod with one centimeter aluminum films at each end. The functional gain represents the amount of 'control' each controller applies to a point along the rod. The controller on the left end of the rod is

represented by $q_o(t)$ and the controller on the right end of the rod is represented by $q_1(t)$. The functional gains converge to a continuous function as the number of elements in the numerical approximation, N, approaches infinity.

Figures 3.3.2, 3.3.3, 3.3.4, 3.3.5, 3.3.6, and 3.3.7 show open loop and closed loop simulations for periodic initial functions $A_o \sin(\frac{\pi}{10}x)$, $A_o \cos(\frac{\pi}{10}x)$, and $A_o \sin(\frac{2\pi}{10}x)$ with $A_o = \frac{1}{4}$. The number of elements used for the numerical approximation is sixteen. Both the open loop and closed loop solutions decay toward the steady state value of zero. However, the closed loop solutions decay to the steady state solution at a faster rate. Similar results occur when the amplitude is increased to $A_o = \frac{1}{2}$ and Figures 3.3.8, 3.3.9, 3.3.10, 3.3.11, 3.3.12, and 3.3.13 show the response for this case.

When the amplitude is increased even more, the 'shock' wave associated with Burgers' equation becomes more visible in the open loop runs. Figures 3.3.14, 3.3.16, 3.3.18, 3.3.20, 3.3.22, and 3.3.24 contain the open loop responses for the cases $A_o = 1$ and $A_o = 2$. The corresponding closed loop responses are given in Figures 3.3.15, 3.3.17, 3.3.19, 3.3.21, 3.3.23, and 3.3.25, respectively. The effects of the boundary controls are enhanced as a result of the increase in the amplitude of the initial functions. Note, specifically in Figure 3.3.15, how far the function is driven below zero at the boundaries by the control.

Finally, the amplitude of the initial function is set to $A_o = 10$. In Figures 3.3.26 and 3.3.28, there is a clearly defined 'shock' traveling toward the right end of the rod. In Figures 3.3.30 and 3.3.32, there are two shock waves which travel to the center of the rod.

Figure 3.3.27 is the closed loop response for $u_o(x) = 10\sin(\frac{\pi}{10}x)$. The closed loop system is changing rapidly in the first two seconds of the simlulation and sixteen elements does not model the change accurately. Increasing the number of approximating elements to $N = 64$ and, then, closing the loop results in Figure 3.3.29. The effects of the control are more visible with the increase in N. In Figures 3.3.31 and 3.3.33, N is still 64.

The response to an initial function of the form $A_o\cos(\frac{\pi}{10}x)$ or $A_o\sin(\frac{2\pi}{10}x)$ stabilizes faster than the closed loop system with an initial function of the form $A_o\sin(\frac{\pi}{10}x)$.

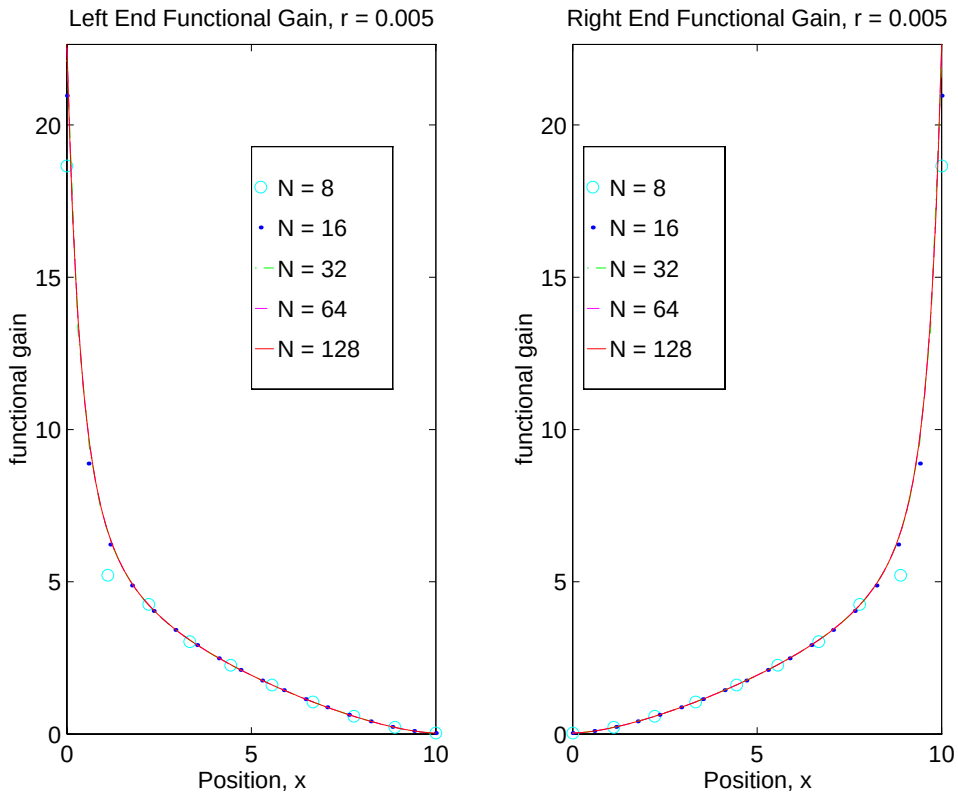


Figure 3.3.1: Functional gains, r = .005, for a 10cm Cu rod with 1cm Al films

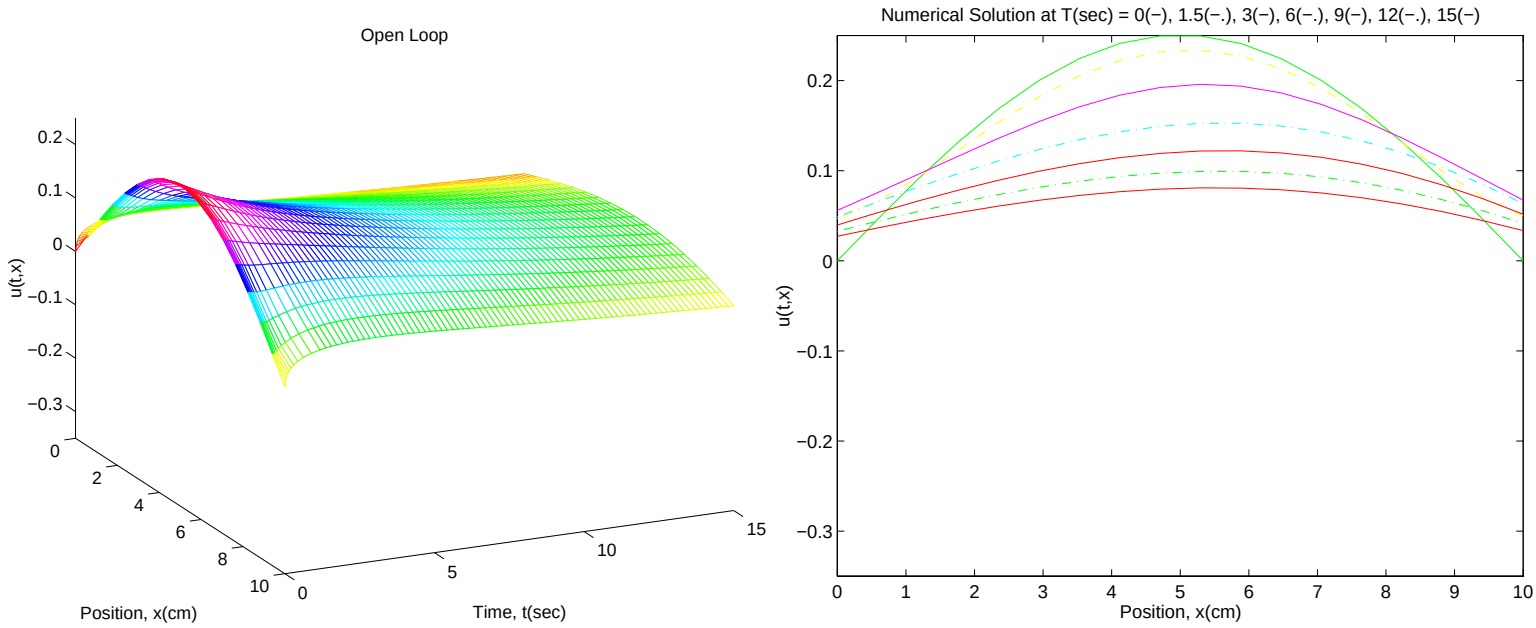


Figure 3.3.2: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\sin(\frac{\pi}{10}x)$, N = 16

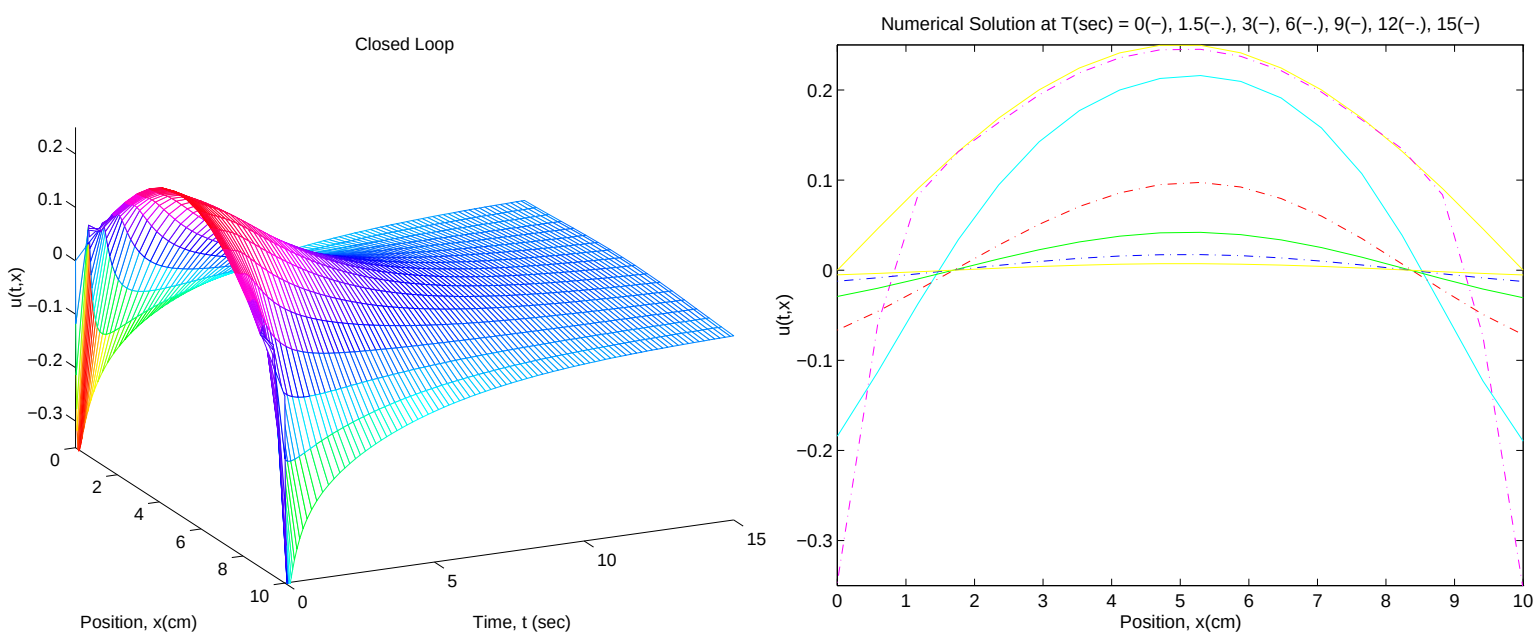


Figure 3.3.3: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\sin(\frac{\pi}{10}x)$, N = 16

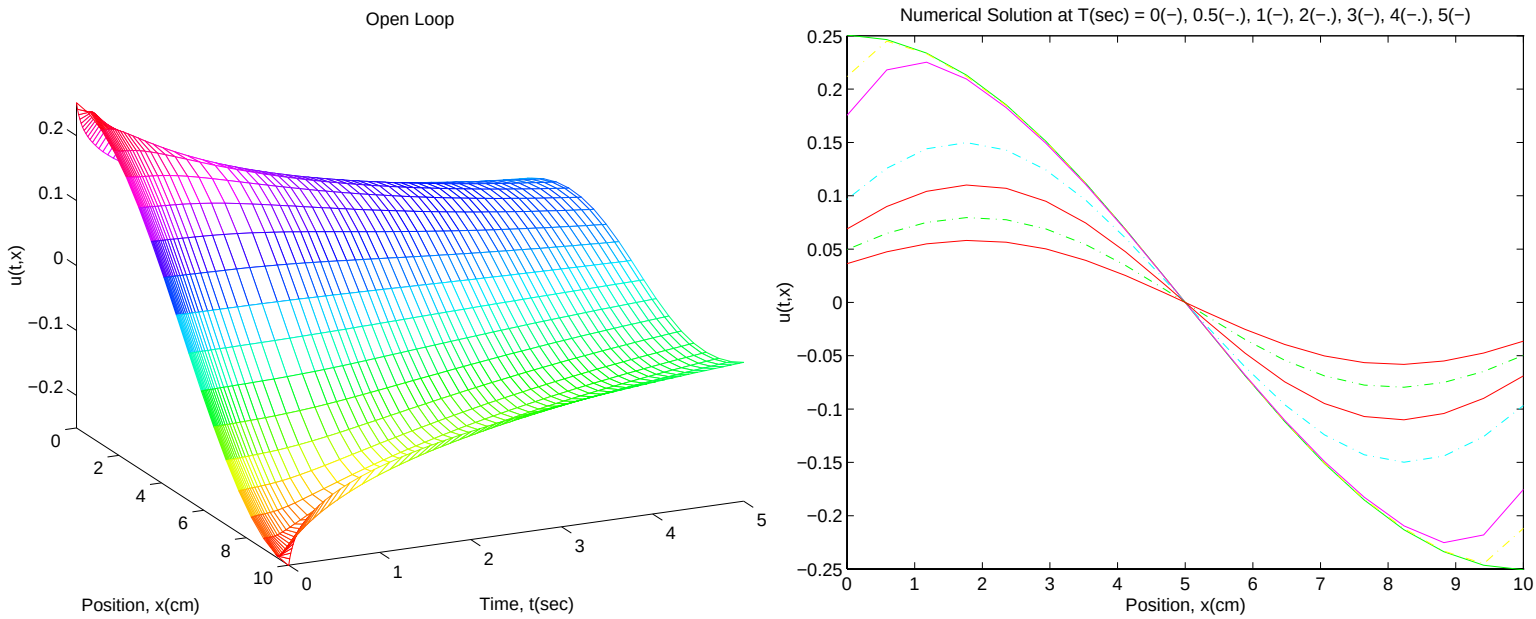


Figure 3.3.4: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\cos(\frac{\pi}{10}x)$, N = 16

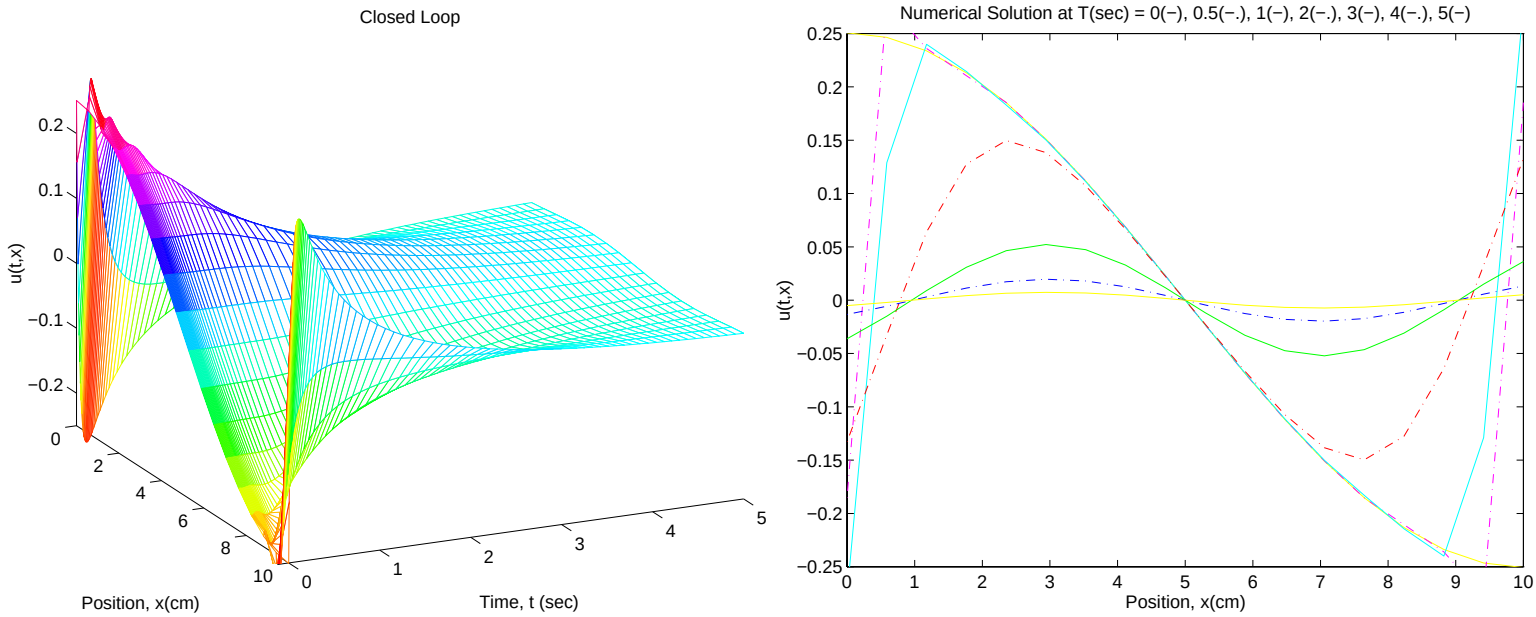


Figure 3.3.5: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\cos(\frac{\pi}{10}x)$, N = 16

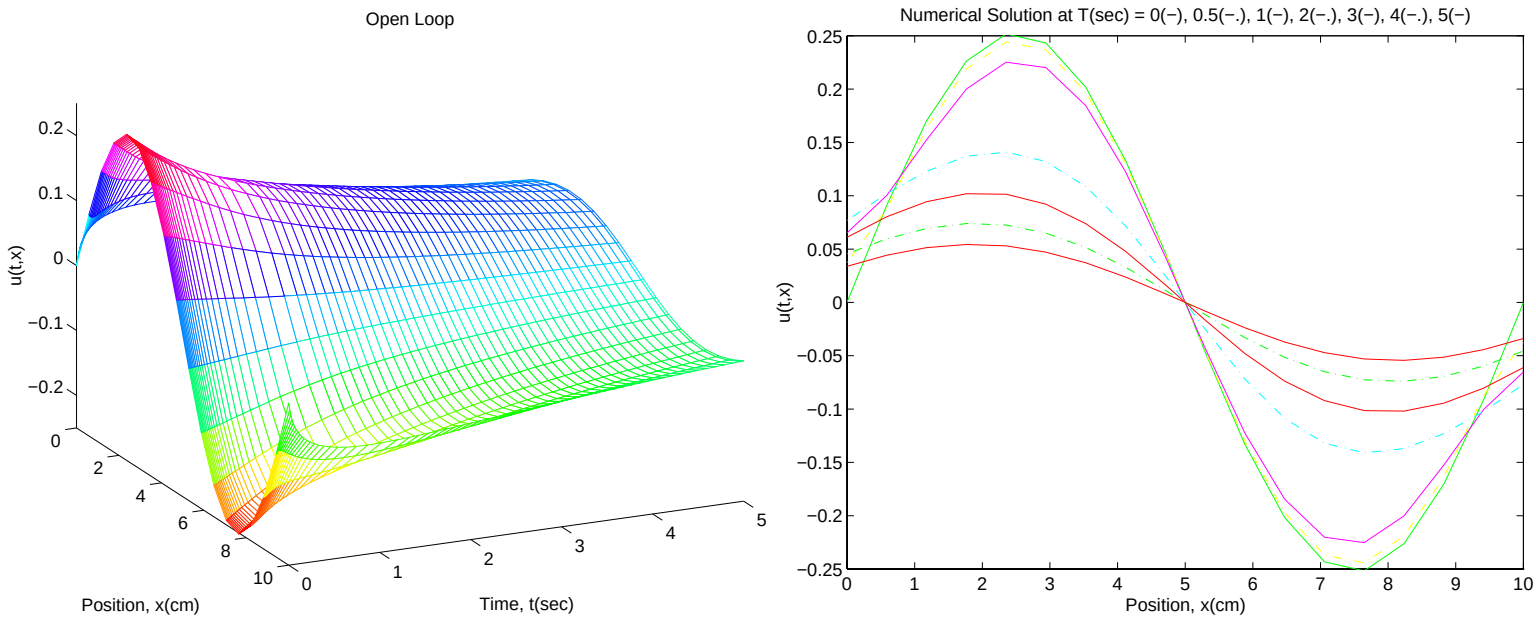


Figure 3.3.6: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\sin(\frac{2\pi}{10}x)$, N = 16

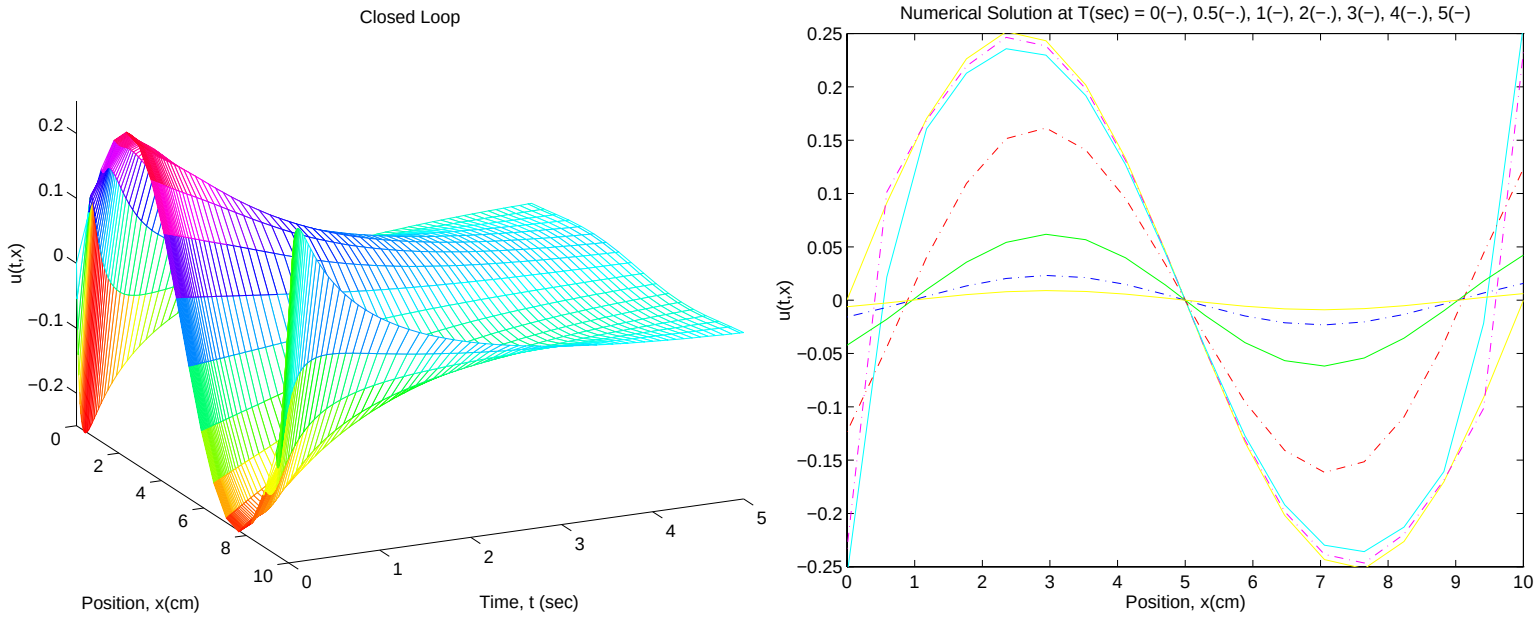


Figure 3.3.7: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{4}\sin(\frac{2\pi}{10}x)$, N = 16

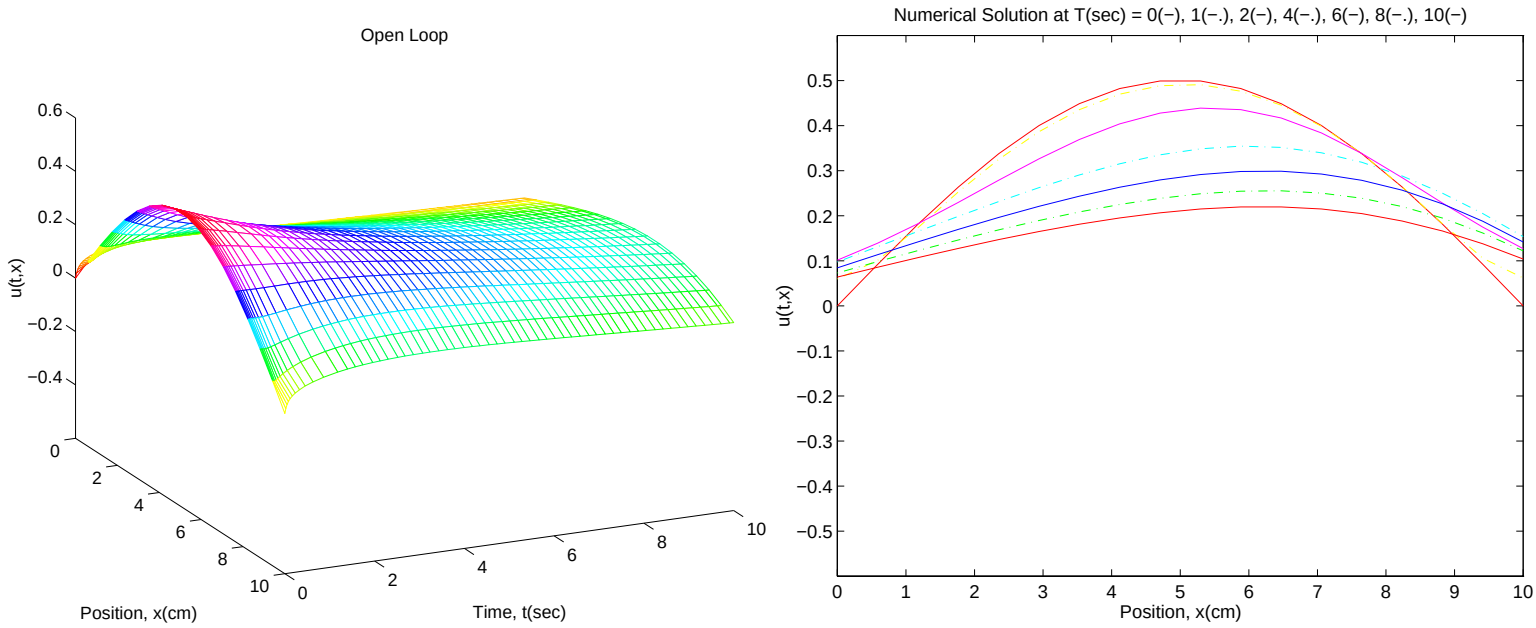


Figure 3.3.8: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\sin(\frac{\pi}{10}x)$, N = 16

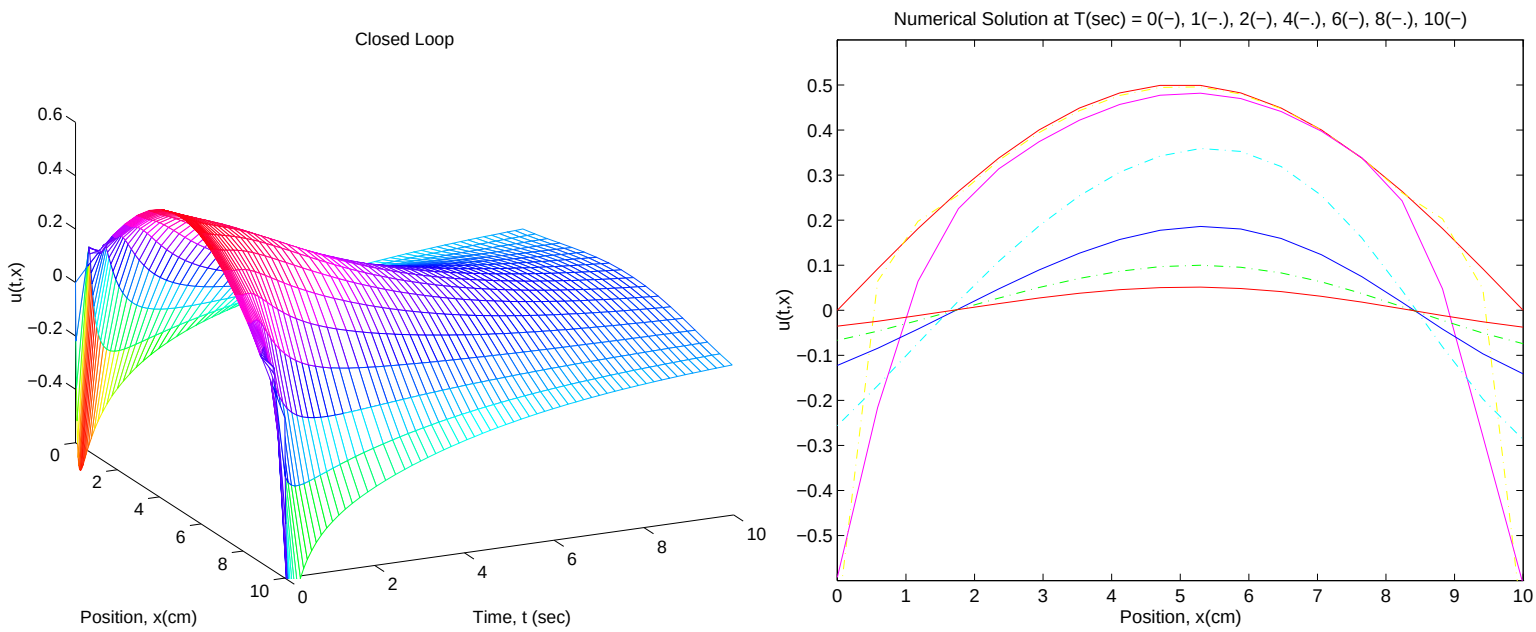


Figure 3.3.9: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\sin(\frac{\pi}{10}x)$, N = 16

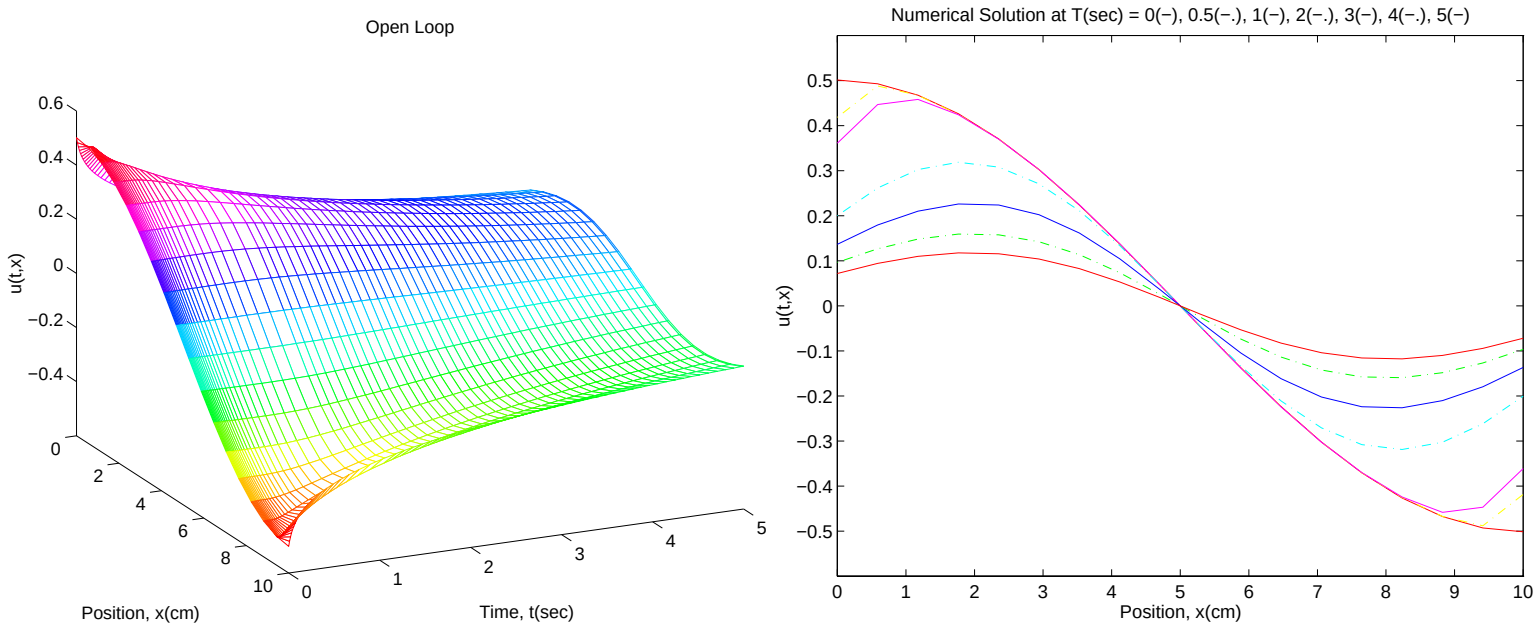


Figure 3.3.10: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\cos(\frac{\pi}{10}x)$, N = 16

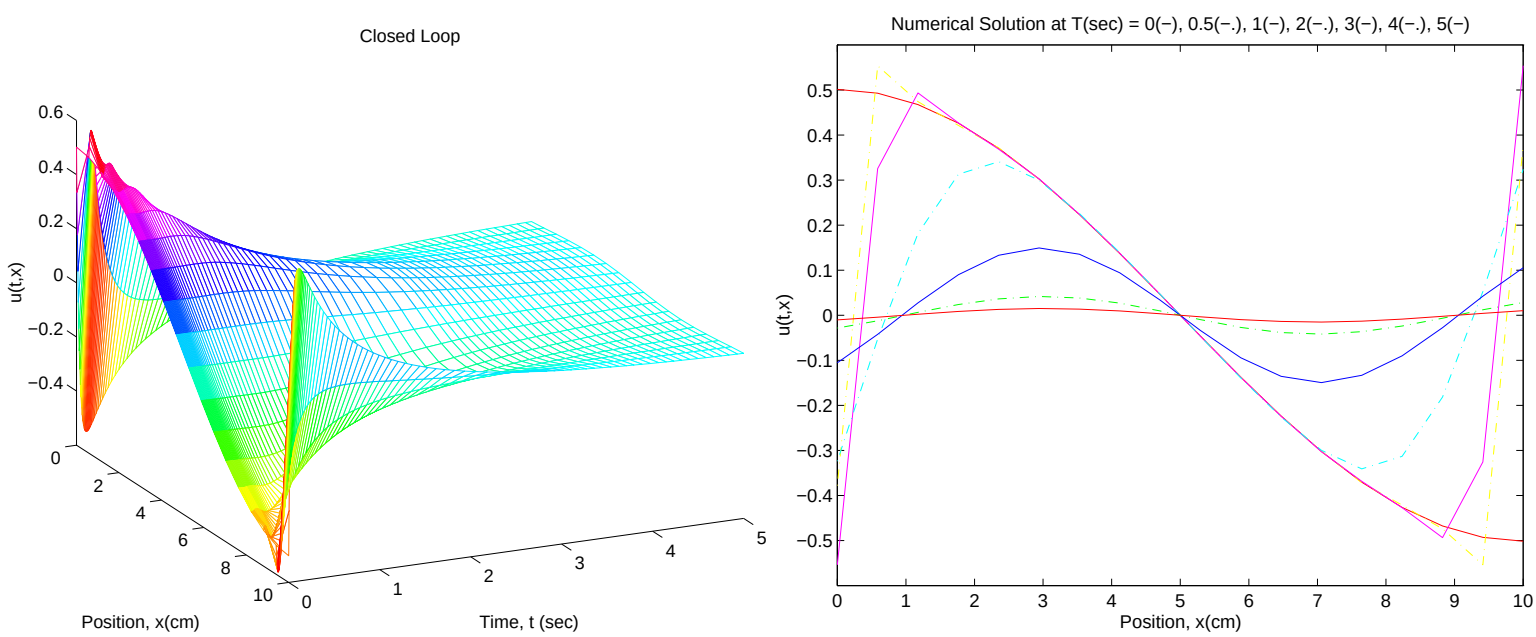


Figure 3.3.11: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\cos(\frac{\pi}{10}x)$, N = 16

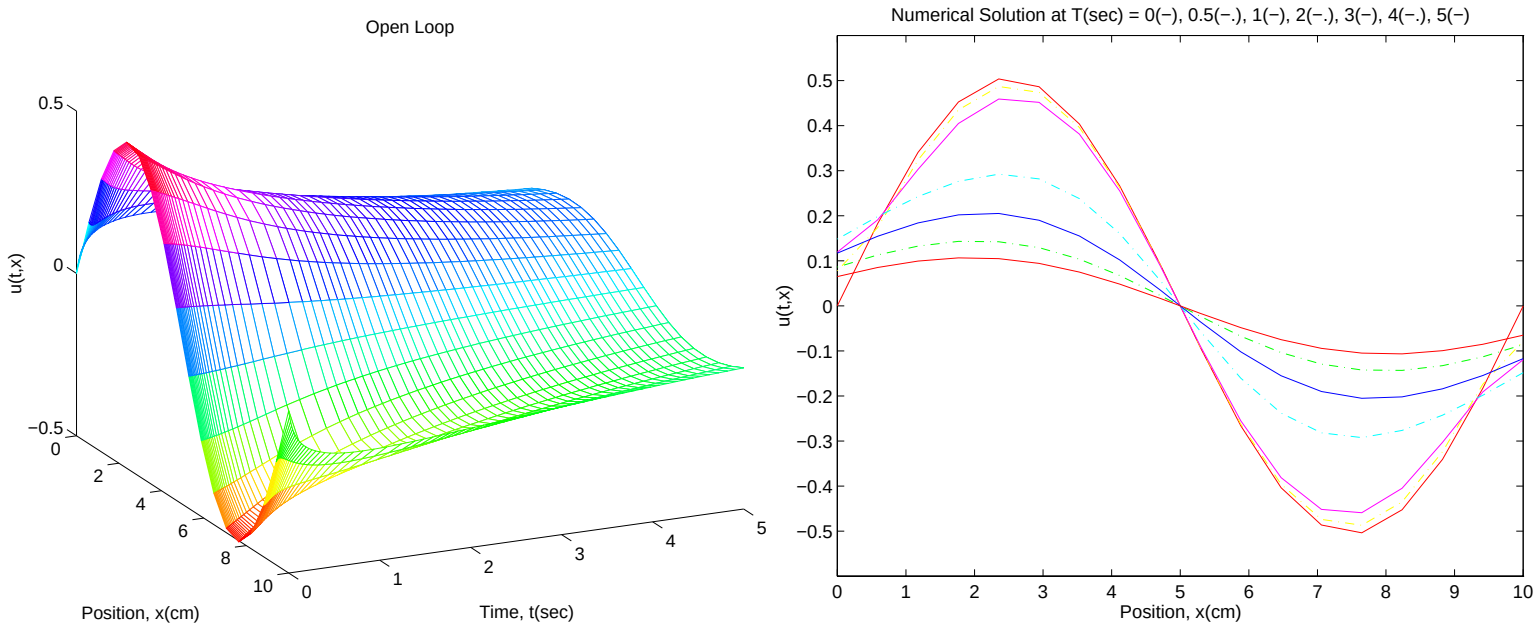


Figure 3.3.12: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\sin(\frac{2\pi}{10}x)$, N = 16

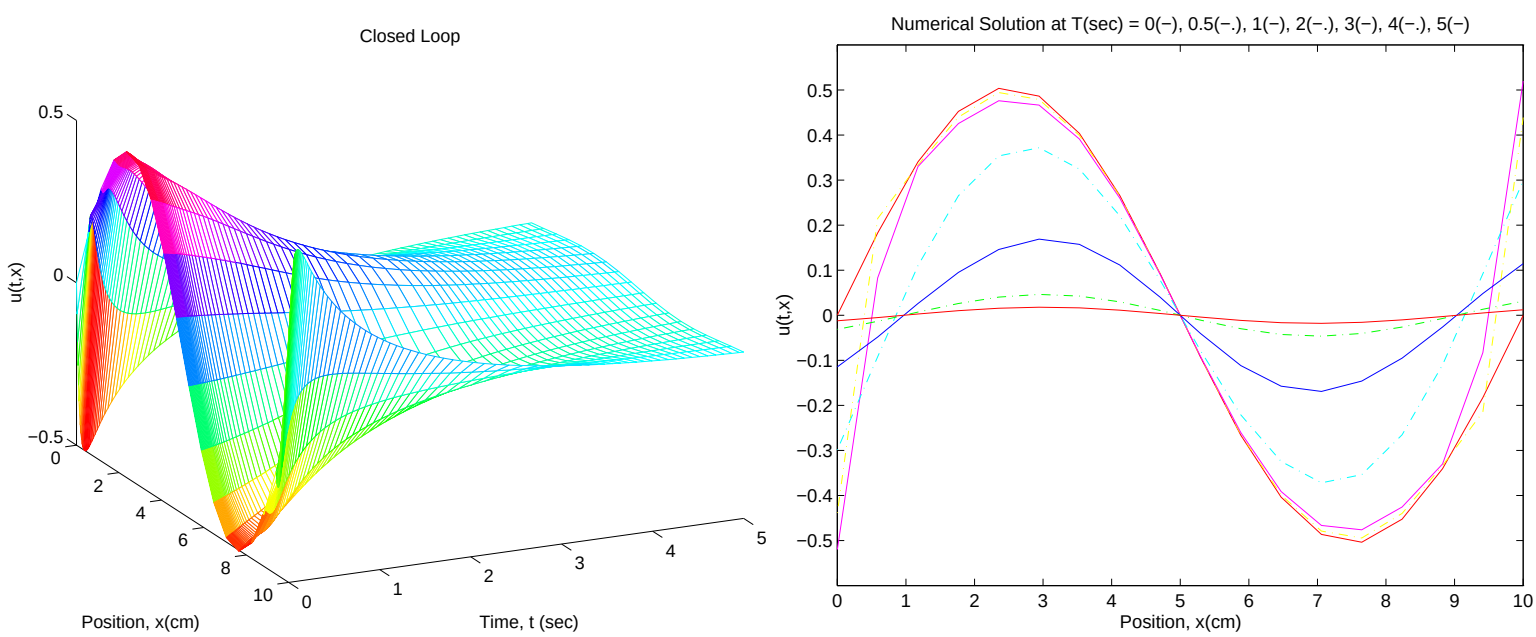


Figure 3.3.13: 10cm Cu rod, 1cm Al films, $u_o(x) = \frac{1}{2}\sin(\frac{2\pi}{10}x)$, N = 16

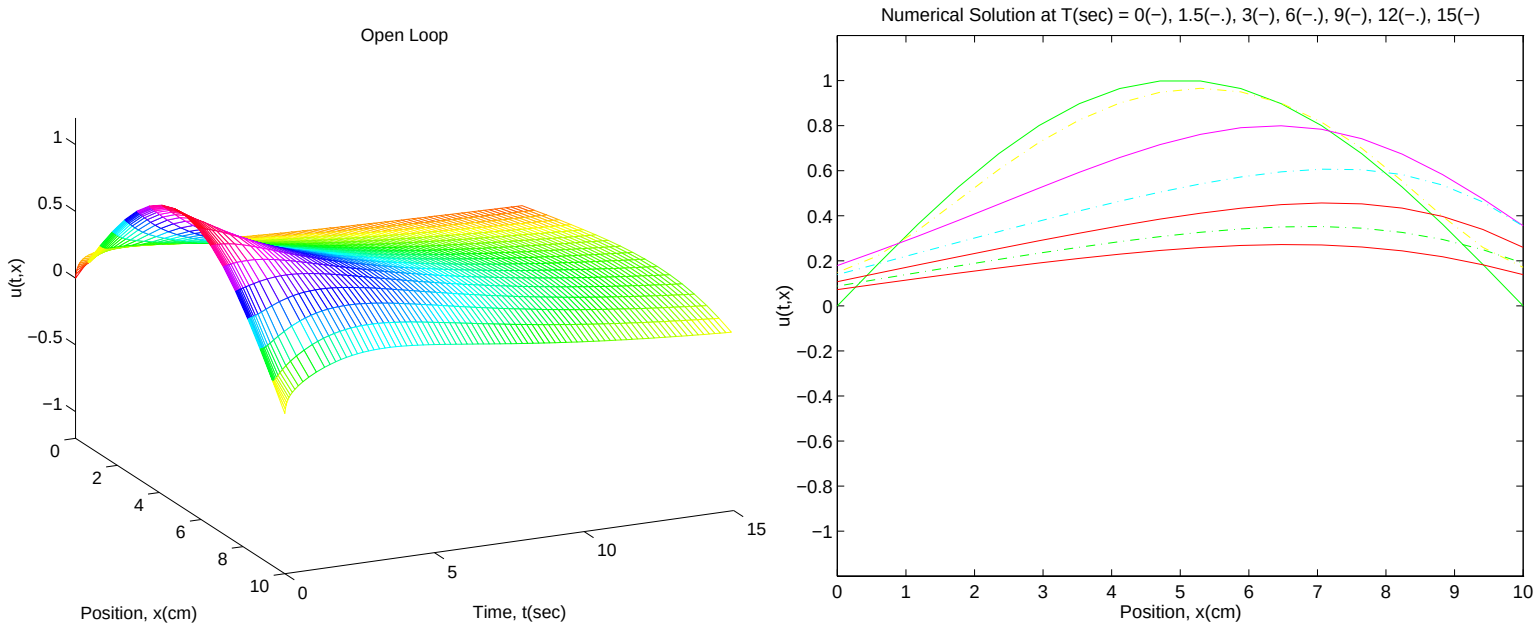


Figure 3.3.14: 10cm Cu rod, 1cm Al films, $u_o(x) = \sin(\frac{\pi}{10}x)$

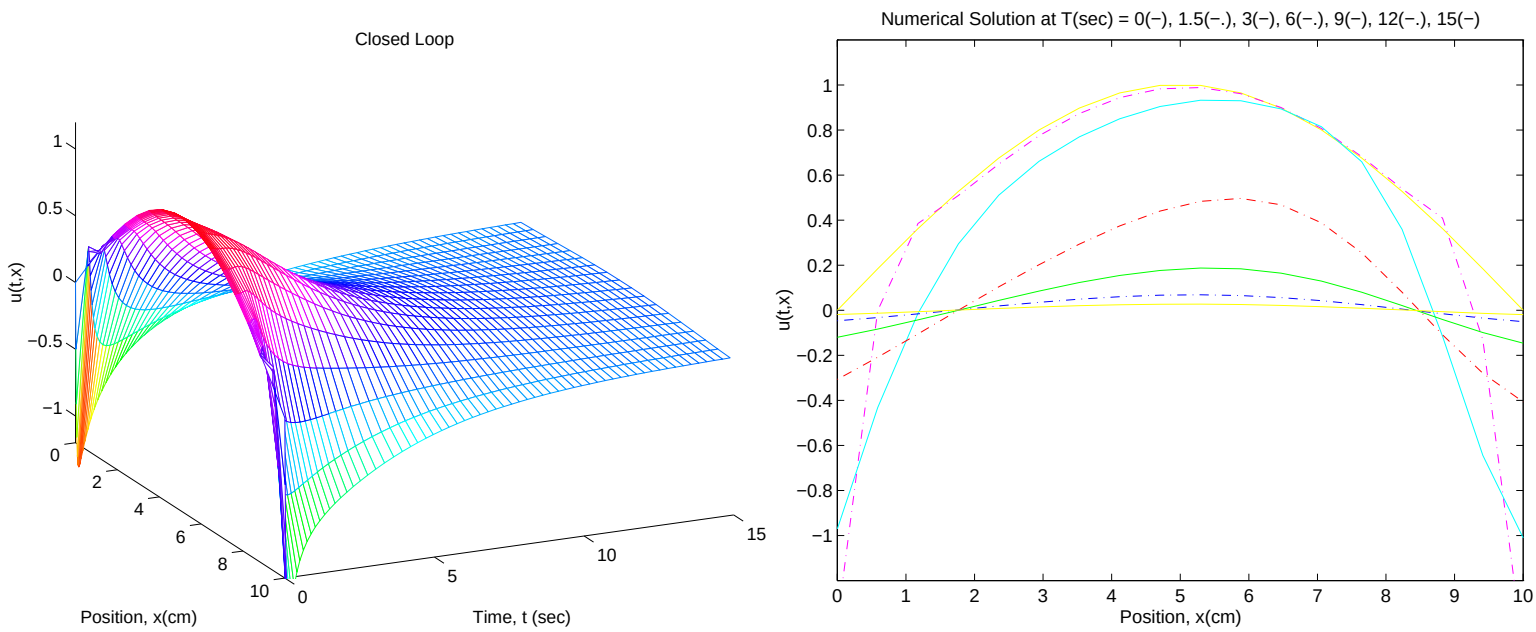


Figure 3.3.15: 10cm Cu rod, 1cm Al films, $u_o(x) = \sin(\frac{\pi}{10}x)$, N = 16

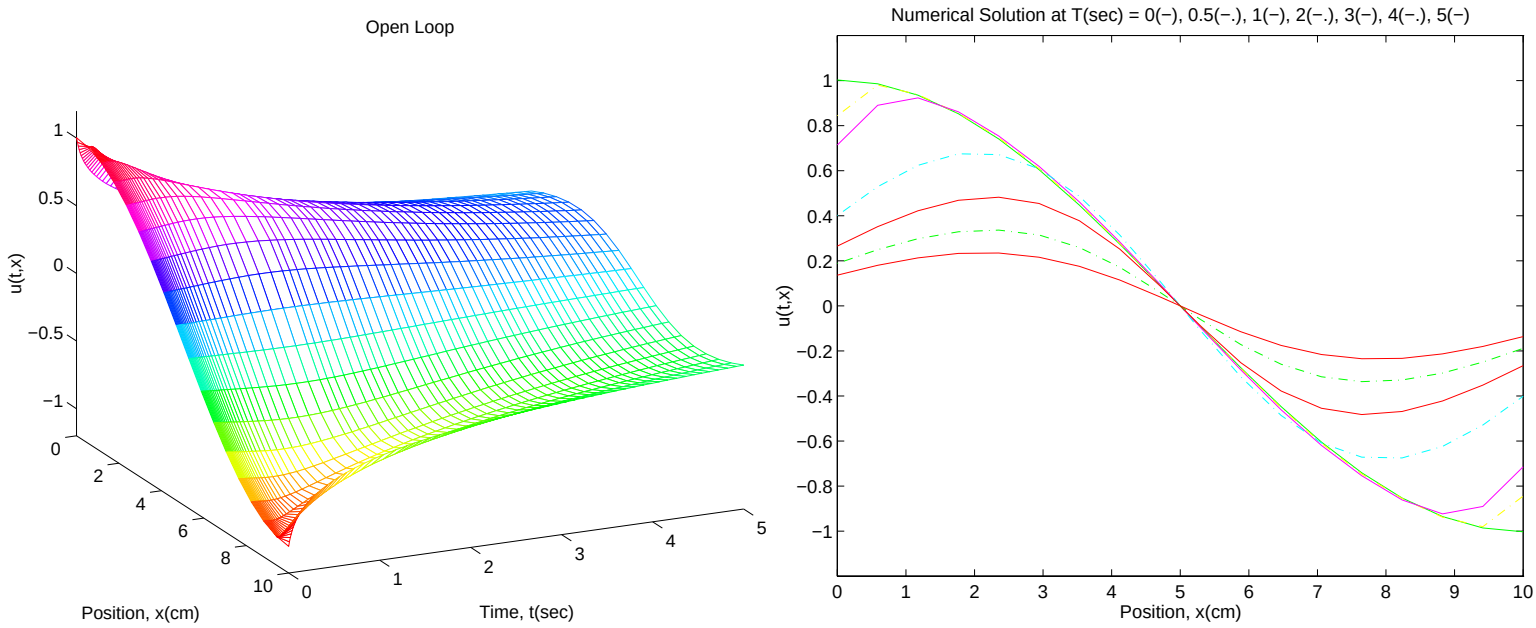


Figure 3.3.16: 10cm Cu rod, 1cm Al films, $u_o(x) = \cos(\frac{\pi}{10}x)$, N = 16

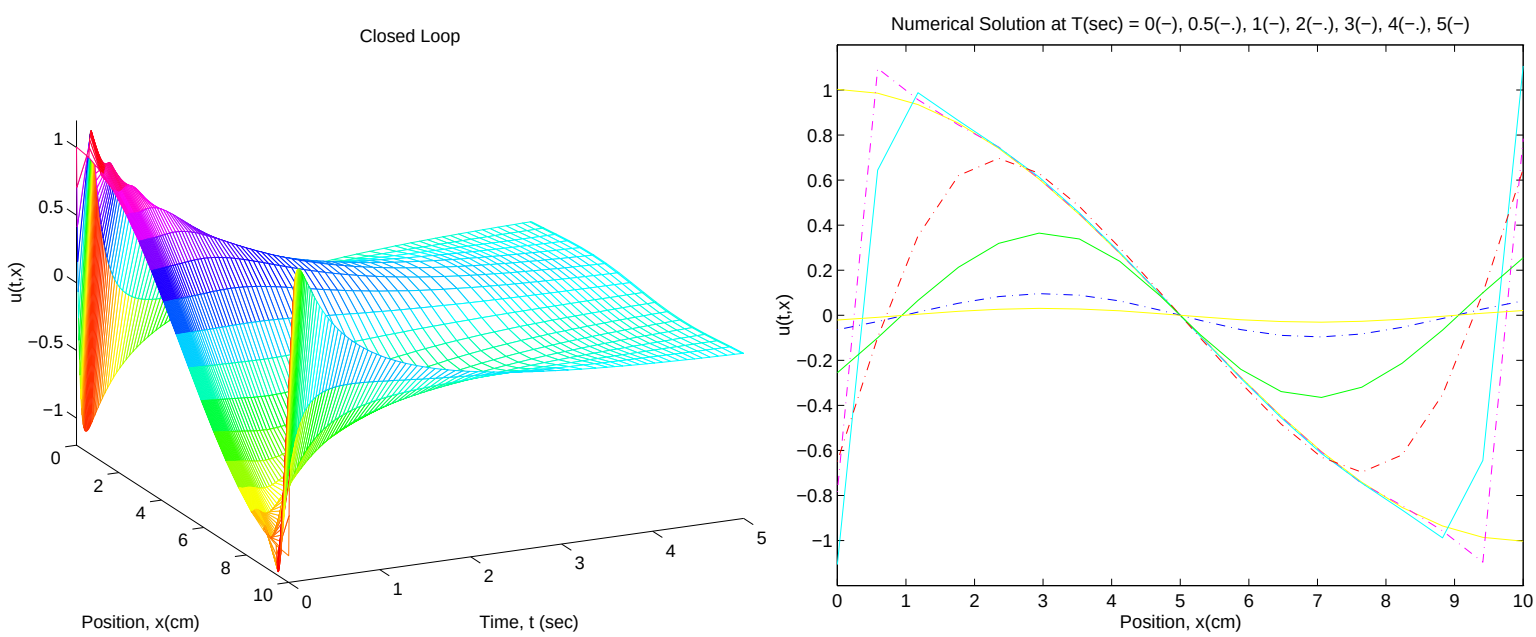


Figure 3.3.17: 10cm Cu rod, 1cm Al films, $u_o(x) = \cos(\frac{\pi}{10}x)$, N = 16

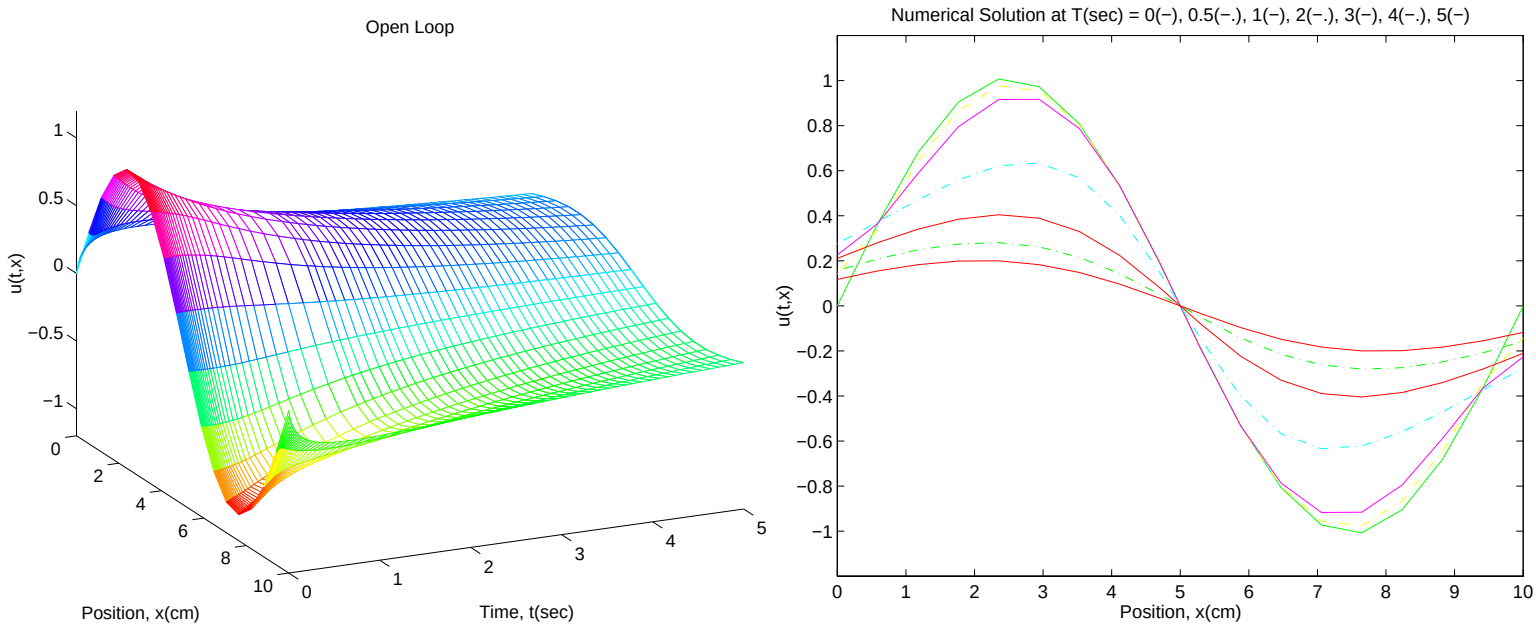


Figure 3.3.18: 10cm Cu rod, 1cm Al films, $u_o(x) = \sin(\frac{2\pi}{10}x)$, N = 16

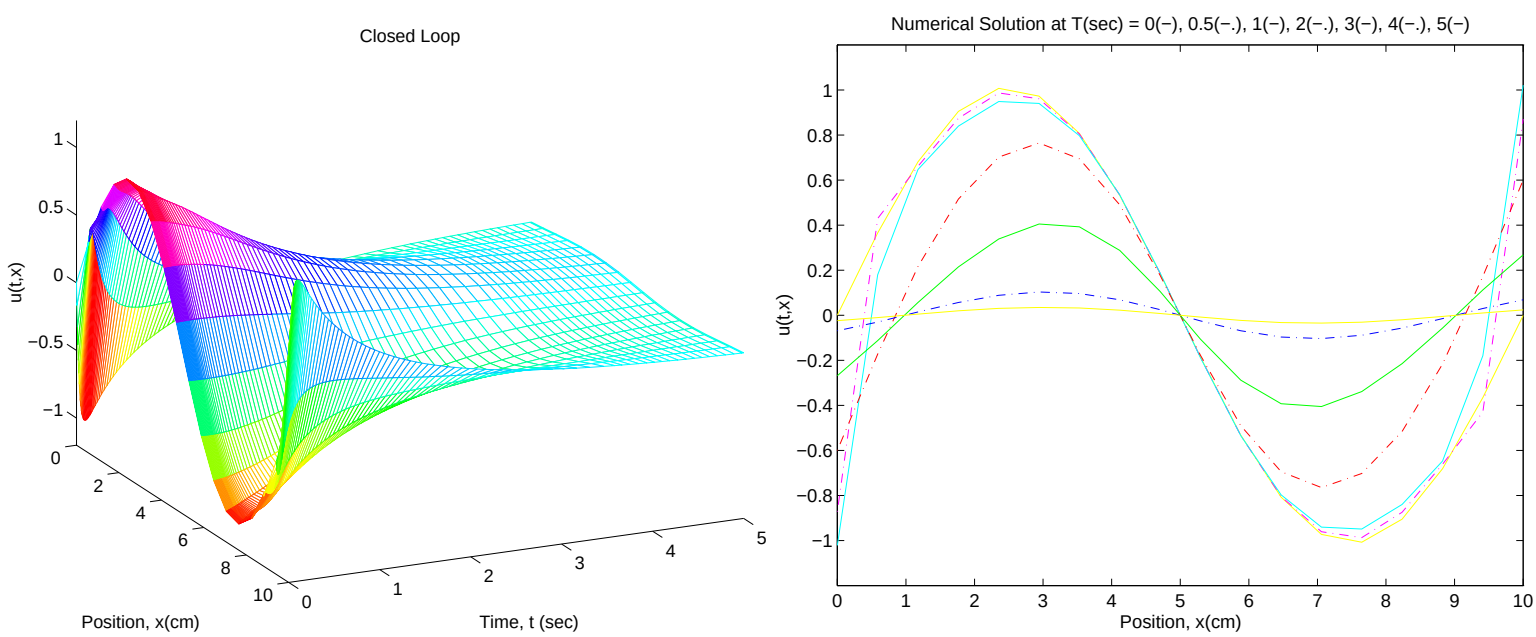


Figure 3.3.19: 10cm Cu rod, 1cm Al films, $u_o(x) = \sin(\frac{2\pi}{10}x)$, N = 16

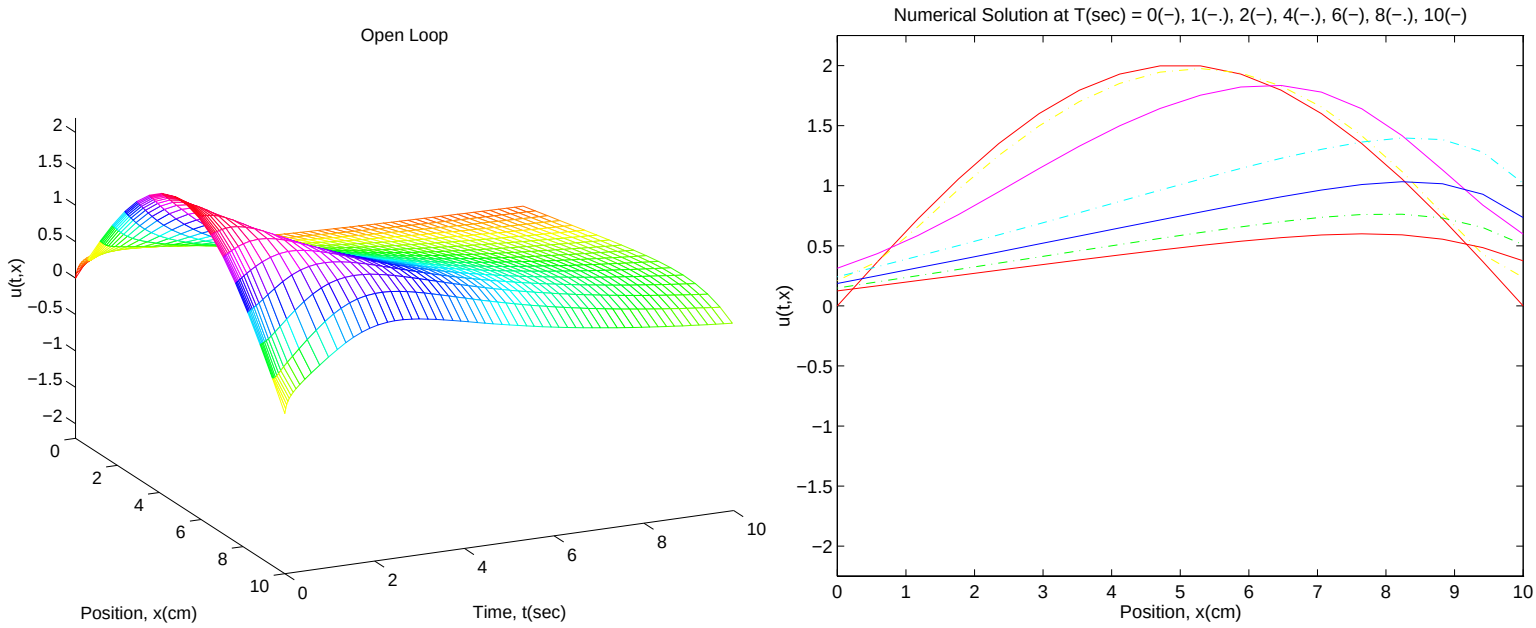


Figure 3.3.20: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\sin(\frac{\pi}{10}x)$, N = 16

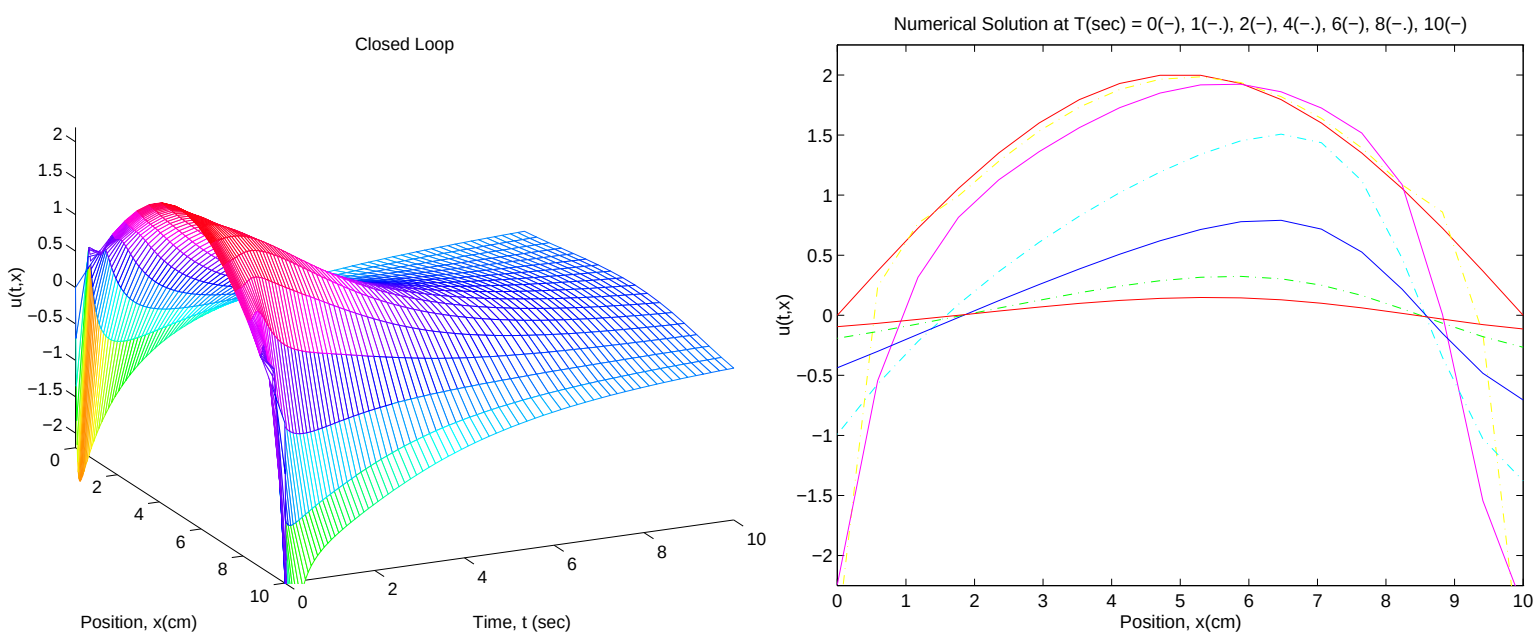


Figure 3.3.21: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\sin(\frac{\pi}{10}x)$, N = 16

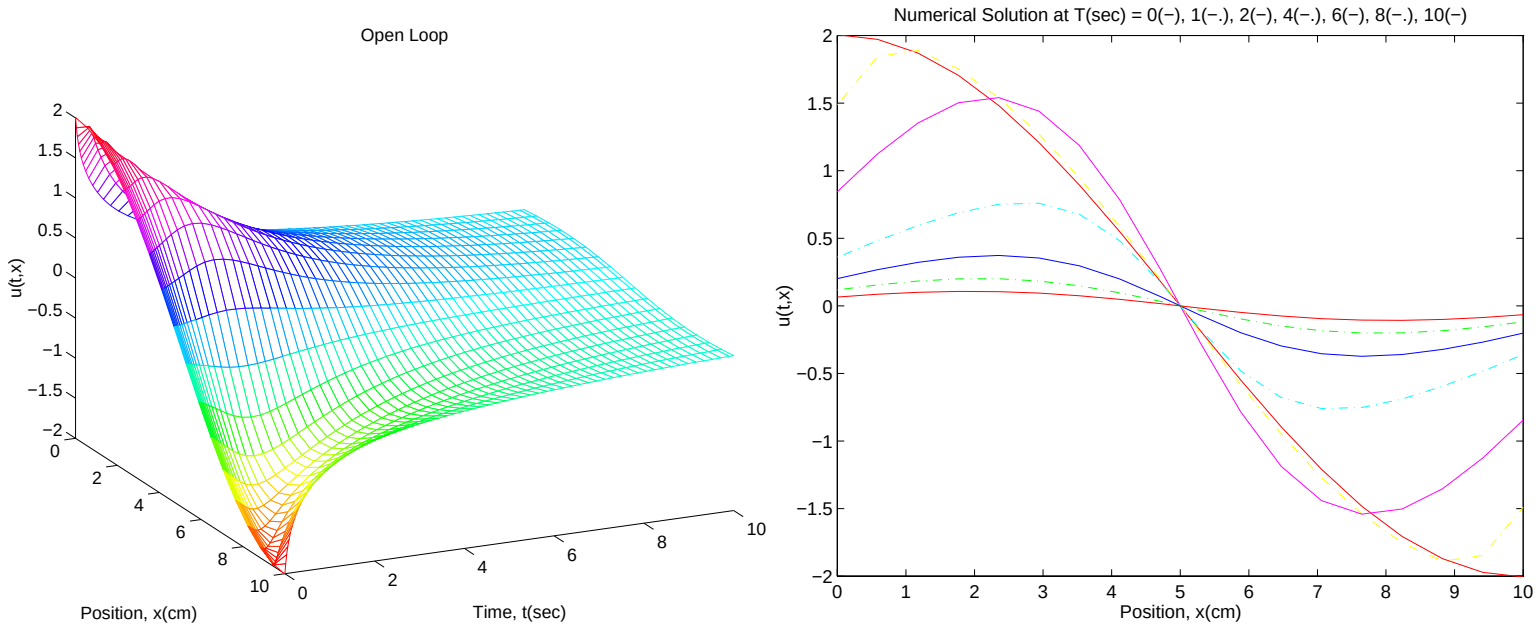


Figure 3.3.22: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\cos(\frac{\pi}{10}x)$, N = 16

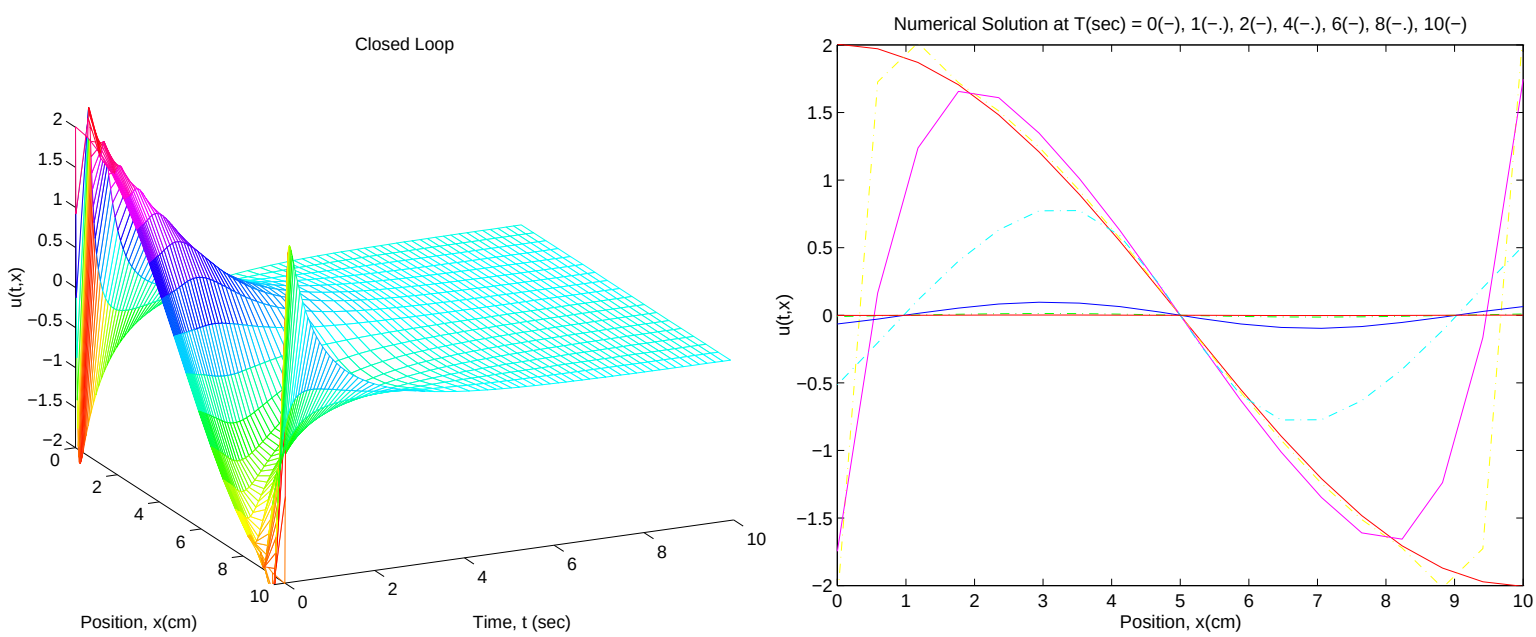


Figure 3.3.23: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\cos(\frac{\pi}{10}x)$, N = 16

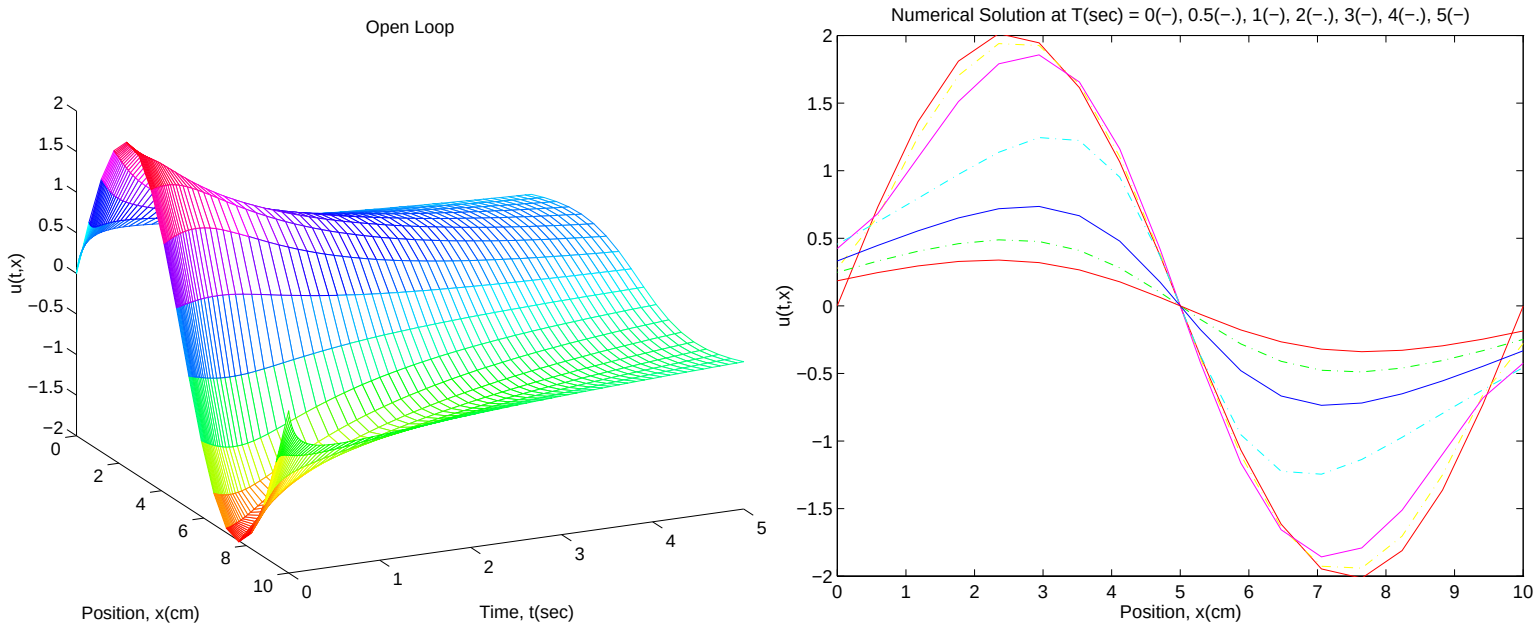


Figure 3.3.24: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\sin(\frac{2\pi}{10}x)$, N = 16

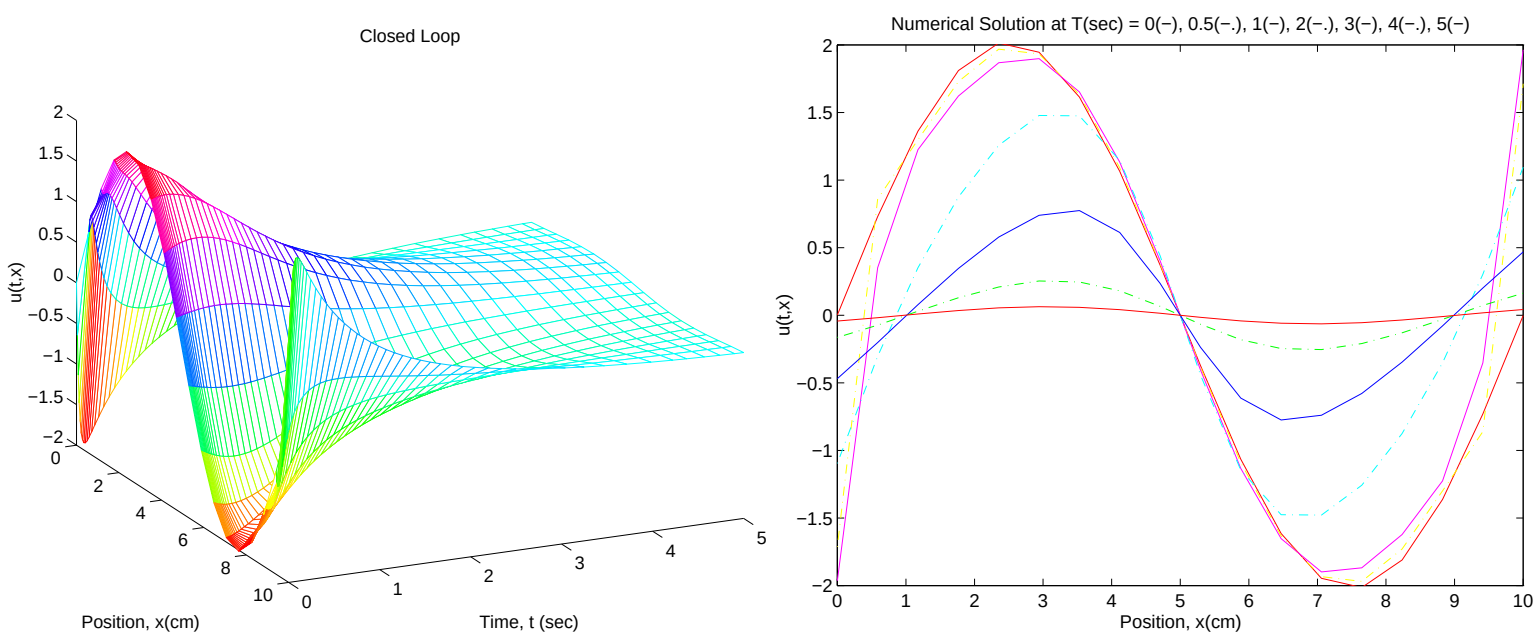


Figure 3.3.25: 10cm Cu rod, 1cm Al films, $u_o(x) = 2\sin(\frac{2\pi}{10}x)$, N = 16

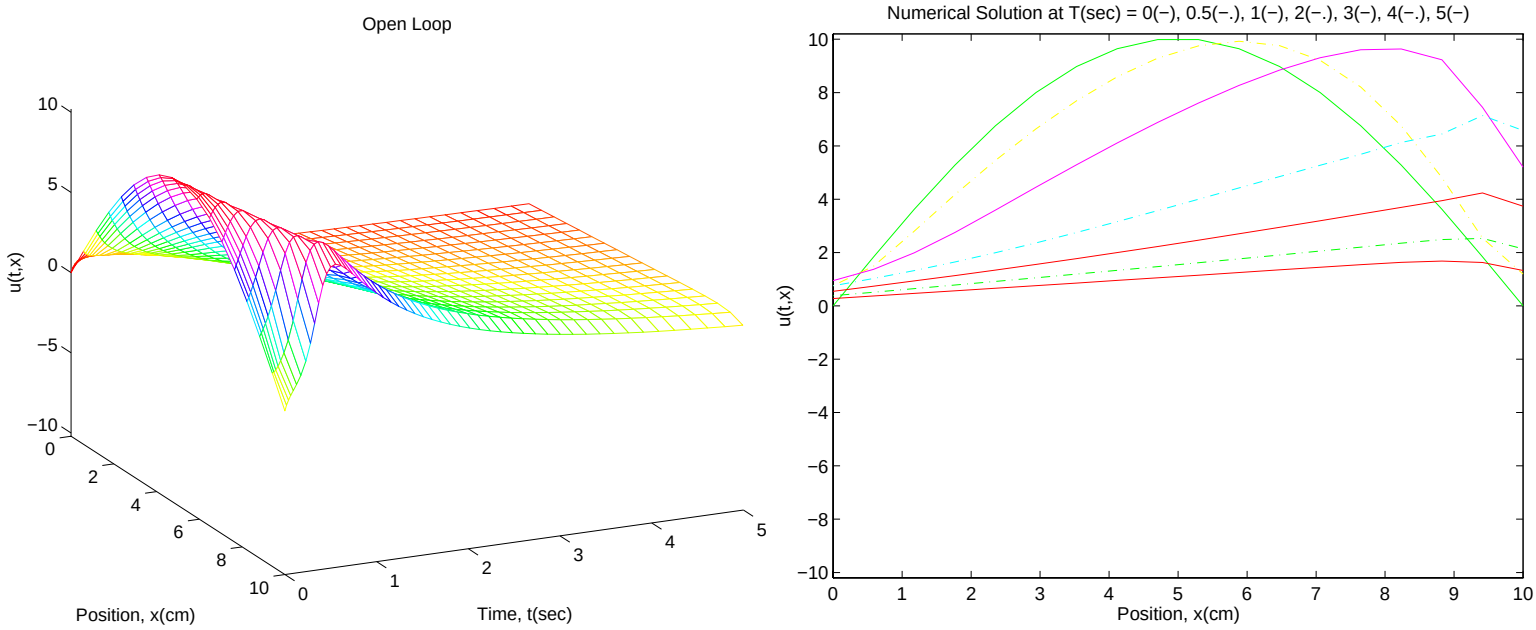


Figure 3.3.26: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{\pi}{10}x)$, N = 16

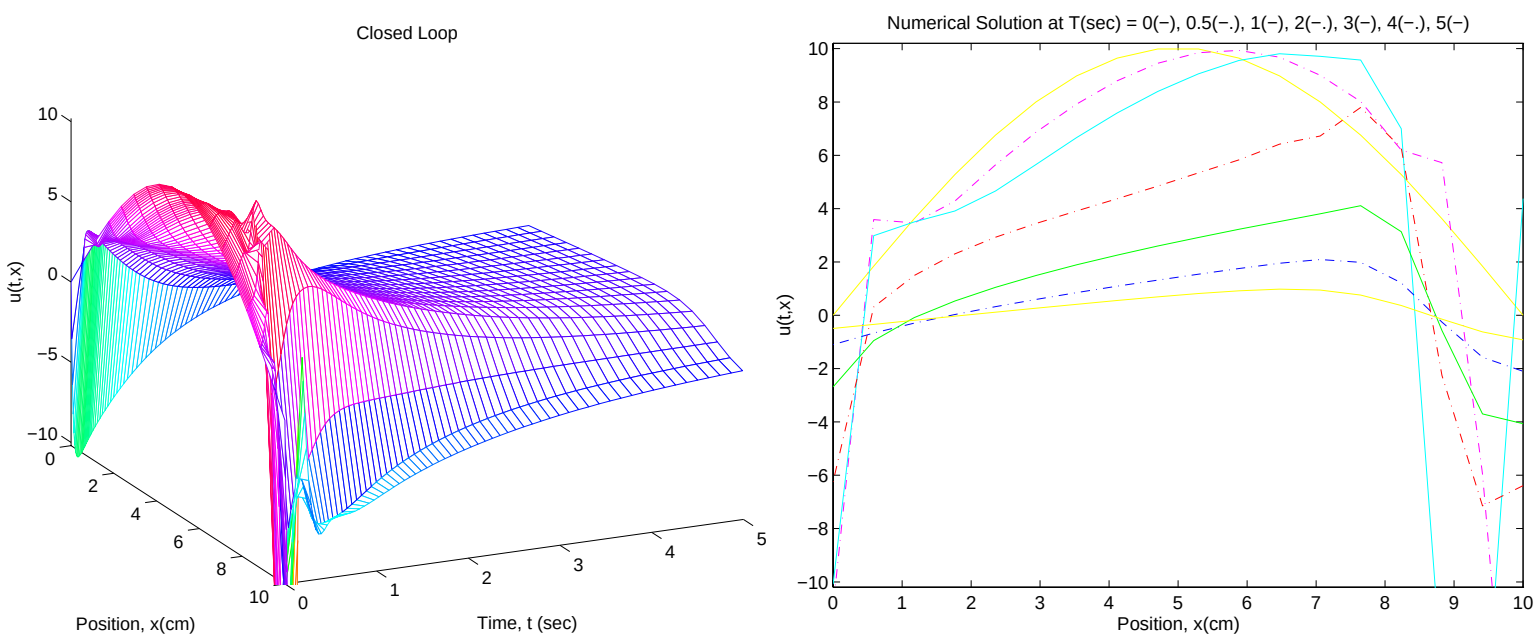


Figure 3.3.27: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{\pi}{10}x)$, N = 16

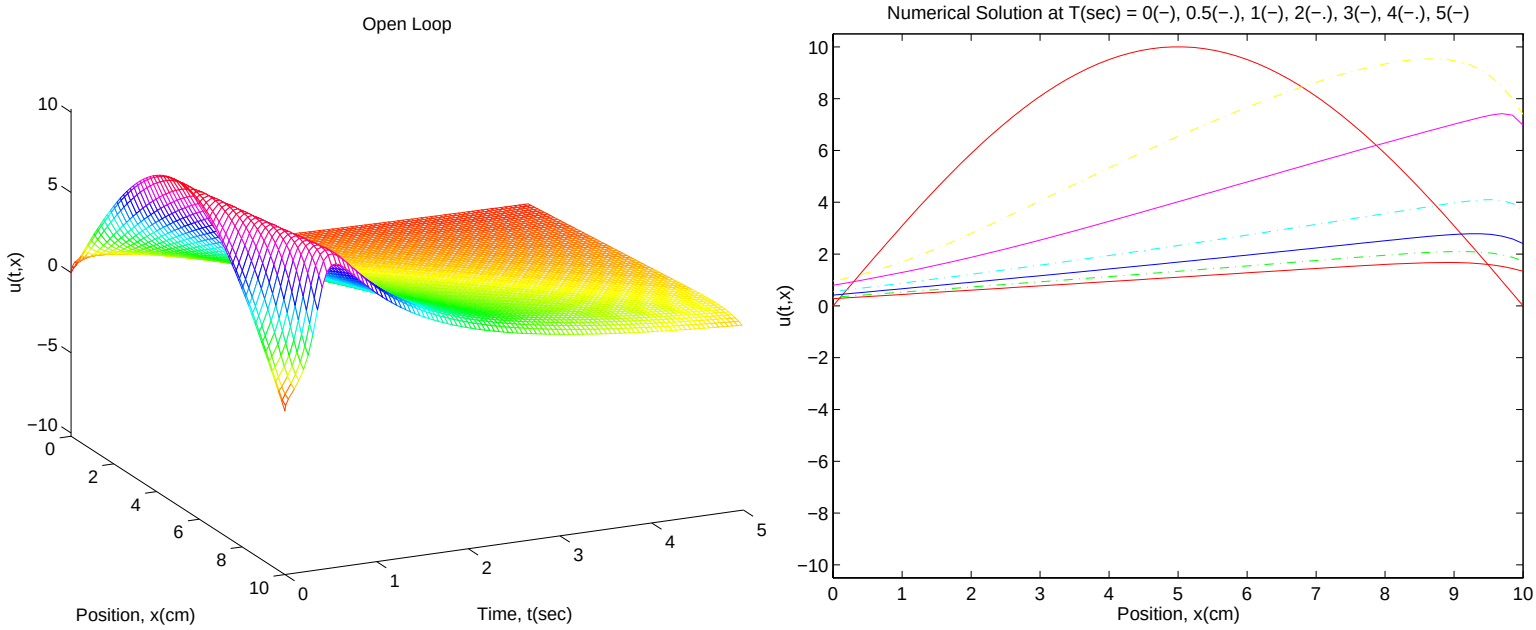


Figure 3.3.28: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{\pi}{10}x)$, N = 64

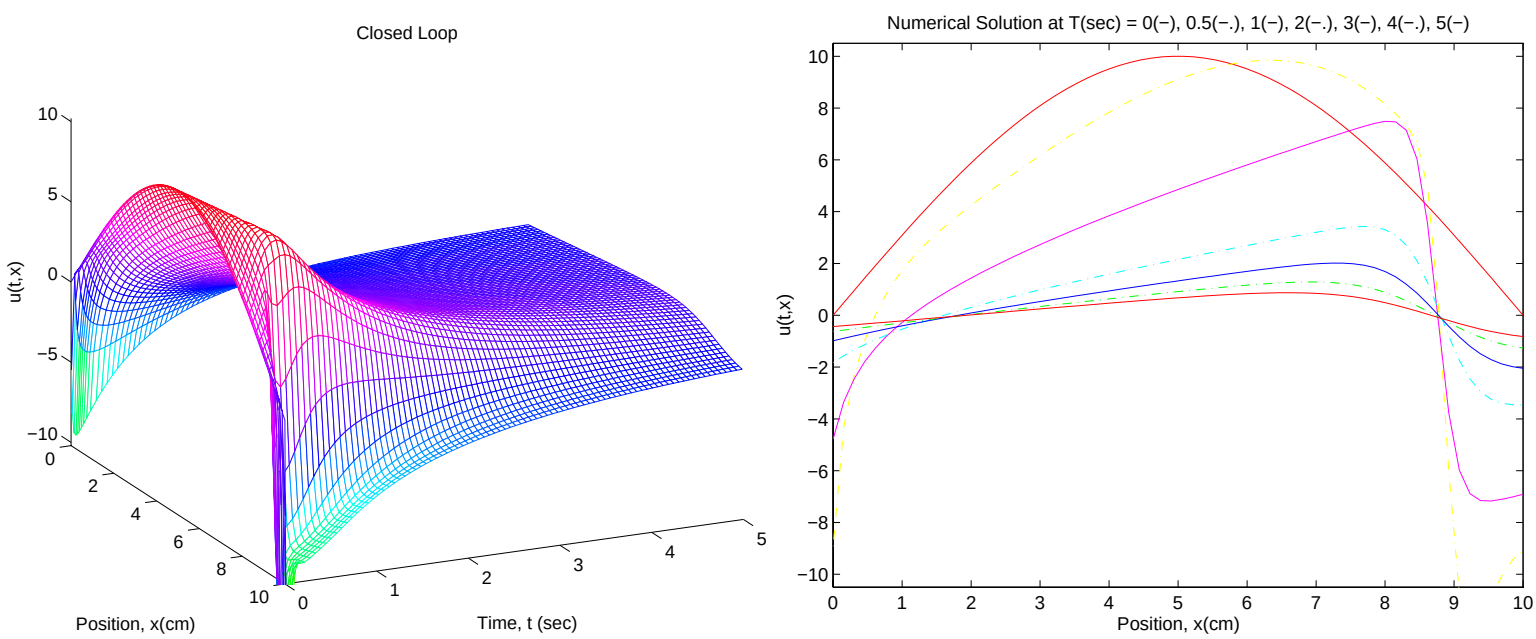


Figure 3.3.29: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{\pi}{10}x)$, N = 64

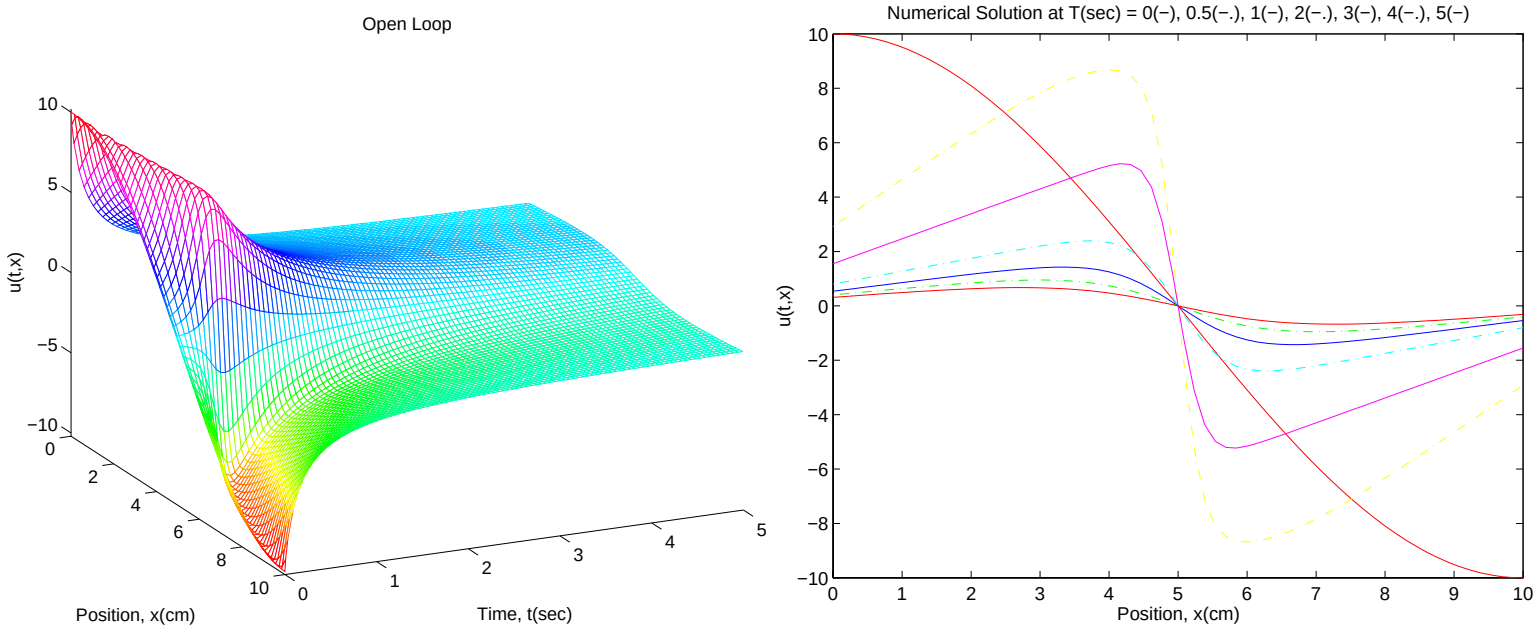


Figure 3.3.30: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\cos(\frac{\pi}{10}x)$, N = 64

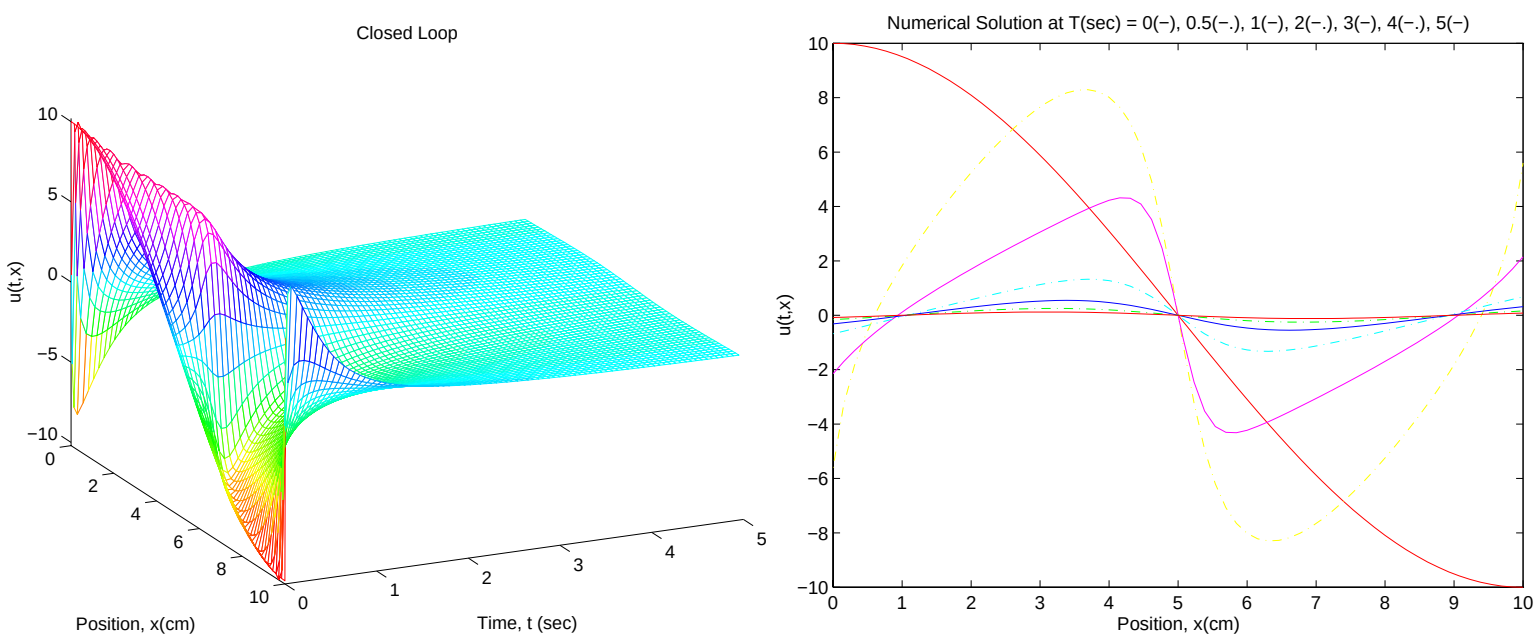


Figure 3.3.31: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\cos(\frac{\pi}{10}x)$, N = 64

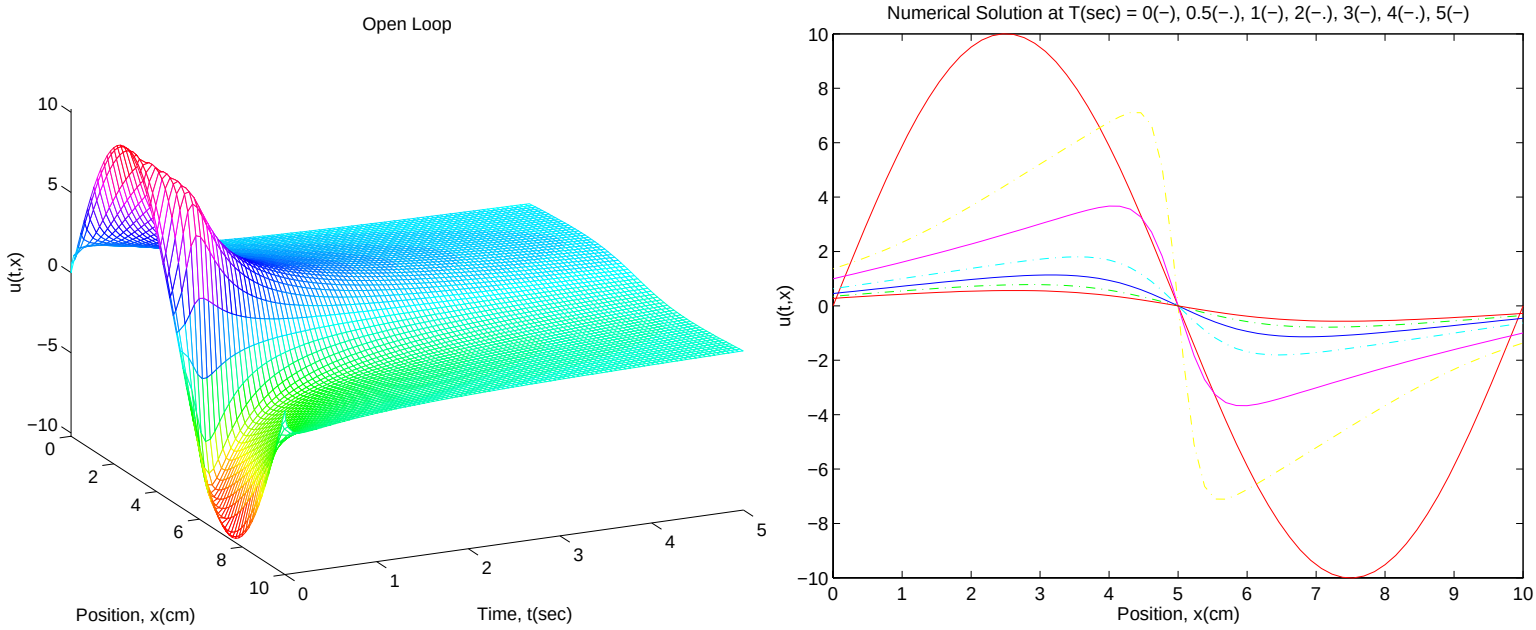


Figure 3.3.32: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{2\pi}{10}x)$, N = 64

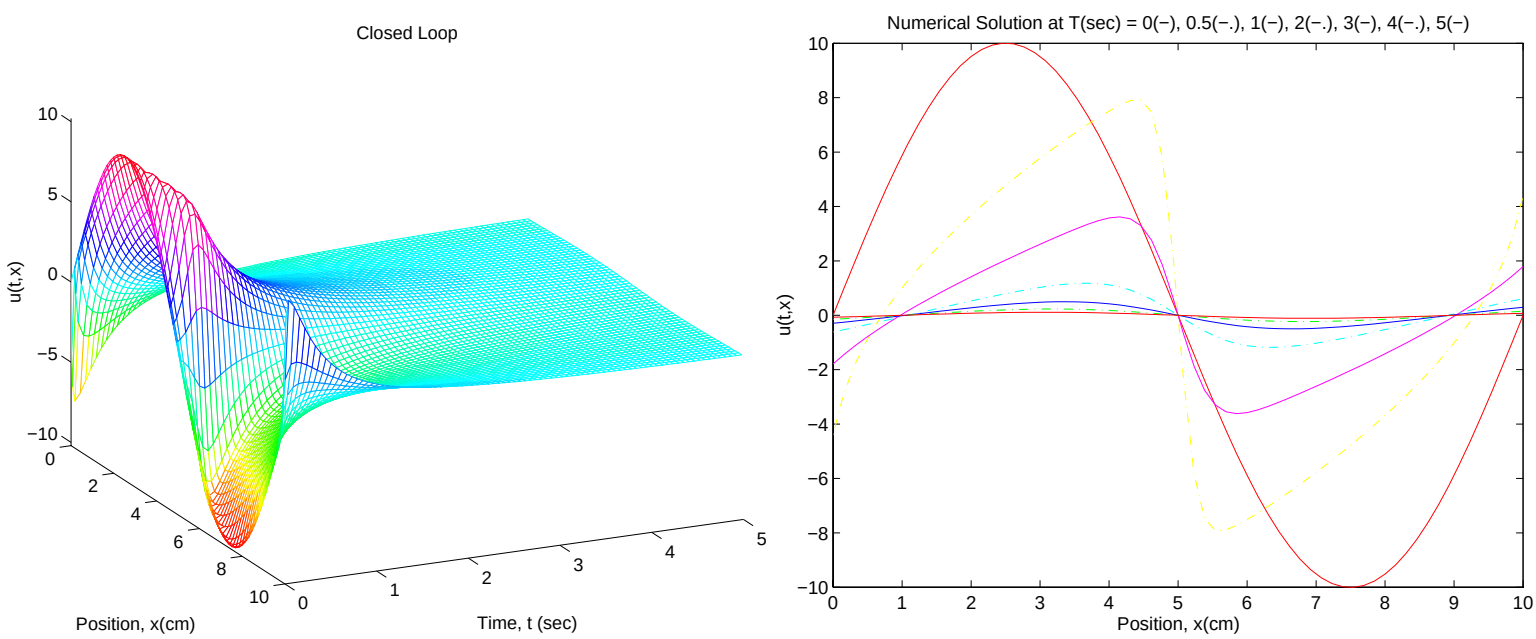


Figure 3.3.33: 10cm Cu rod, 1cm Al films, $u_o(x) = 10\sin(\frac{2\pi}{10}x)$, N = 64

### 3.3.2 Varying the Weighting Constant, $r > 0$

In order to study its effects on the control imposed on a system, the weighting constant, $r > 0$, is varied for two specific systems. The first system considered is from Example 3.5.3 in [16]. The rod is one meter of aluminum with a ten meter iron film at each end. From Table 1.2.1 in [16] $\kappa_1 = \kappa_2 = .124$, $L_1 = L_2 = 1000cm$, $\kappa = .55$, and $\epsilon = \frac{.55}{(2.7)(.217)} = .939$. The initial function is $u_o(x) = .4\cos(\frac{\pi}{100}x)$. Figure 3.3.34 shows the open loop solution. Note that the boundary conditions approximate Neumann boundary conditions. This means $u_x$ is approximately zero at the boundaries for the open loop system. This keeps the solutions nearly horizontal at the boundaries.

Figures 3.3.36, 3.3.38, 3.3.40, 3.3.42, 3.3.44, 3.3.46, and 3.3.48 show the closed loop responses with the weighting constants decreasing from 5 to $5 \times 10^{-6}$ in powers of ten. Figures 3.3.35, 3.3.37, 3.3.39, 3.3.41, 3.3.43, 3.3.45, and 3.3.47 show the corresponding functional gains. The gain matrix for this system is denoted $K_{.939}$, since $\epsilon = .939$ for an aluminum rod. As $r$ is divided by ten, the maximum of the functional gain is four to five times greater than its previous value. Accordingly, the amount of control imposed on the system is increased. While the effects of the control are present for all the closed loop simulations, there is not a large change in the numerical solutions toward zero until $r$ is .005. When $r = 5 \times 10^{-5}$, the numerical solutions have an amplitude no greater than .0267 after five hundred seconds. When $r = 5 \times 10^{-6}$ the amplitude is no greater than .0178 after five hundred seconds. As $r$ decreases, the numerical solutions approach zero at a faster

rate, as expected.

The next system tested is a one meter aluminum rod with a twenty-five centimeter iron film at each end ($L_1 = L_2 = 25cm$). All other constants remain the same. These conditions are used to approximate Dirichlet boundary conditions. These conditions try to drive $u$ to zero in the open loop case. Figure 3.3.49 illustrates the open loop system. Note that the solutions are approaching zero with no control applied.

Again, when the loop is closed, the solutions asymptotically approach the zero steady state condition. Figures 3.3.51, 3.3.53, 3.3.55, 3.3.57, 3.3.59, 3.3.61, and 3.3.63 show the closed loop systems with the weighting constants decreasing from 5 to $5 \times 10^{-6}$ in the same manner as before. The functional gains exhibit the same property of increasing four to five times at the boundaries (Figures: 3.3.50, 3.3.52, 3.3.54, 3.3.56, 3.3.58, 3.3.60, 3.3.62) when $r$ is divided by ten. The effects of the control become more apparent as $r$ decreases below .005. When $r$ is .0005 the amplitude of the numerical solution at three hundred seconds is .055. When $r$ is $5 \times 10^{-5}$ and $5 \times 10^{-6}$, the amplitude at three hundred seconds is .0527. The decrease in amplitude expected is not apparent, when $r$ reaches $5 \times 10^{-6}$. This suggests there may be limit to the effectiveness of the control as $r \to 0$.

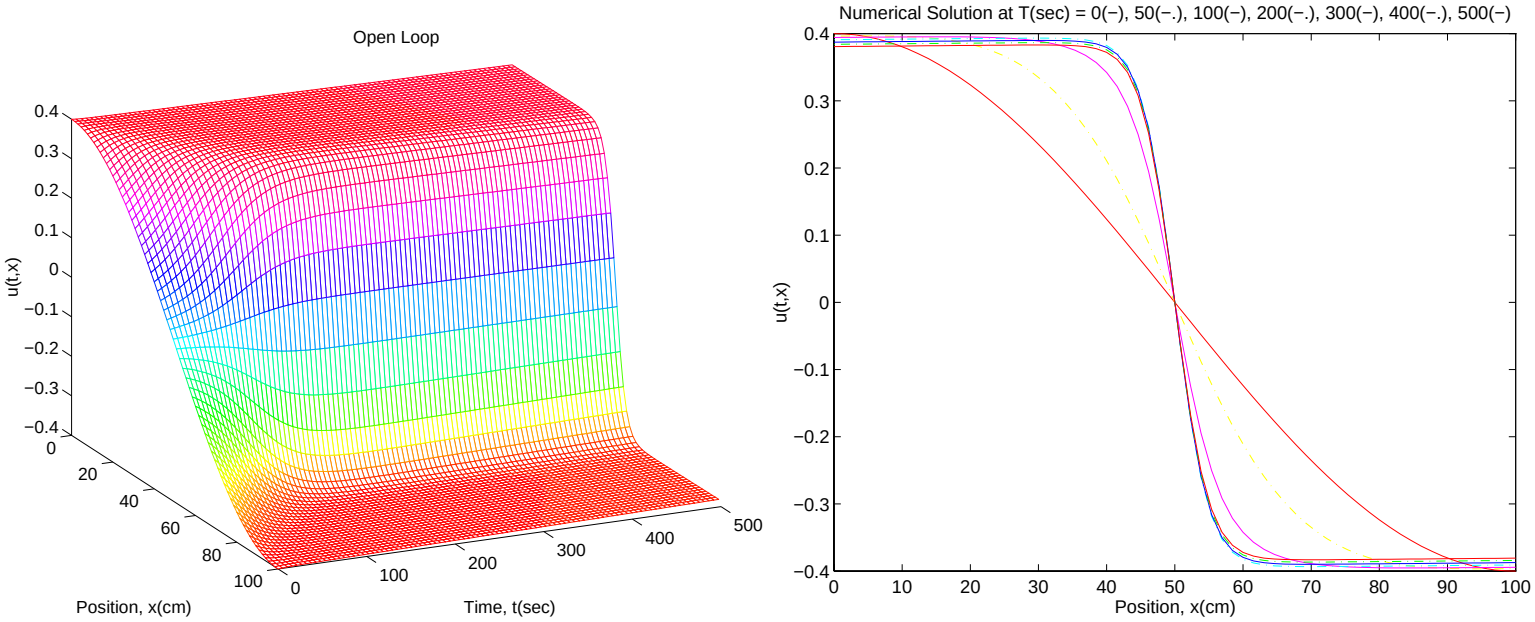


Figure 3.3.34: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$

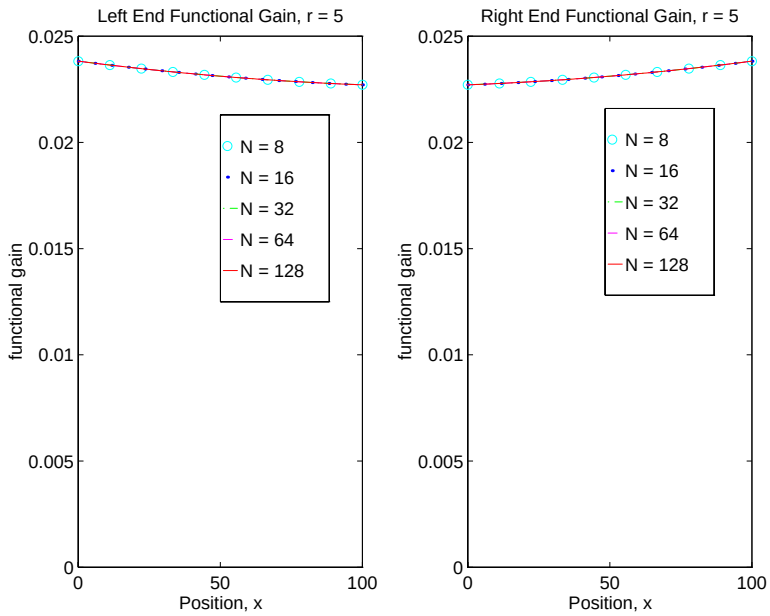


Figure 3.3.35: Functional gains, r = 5 for a 1m Al rod with 10m Fe films

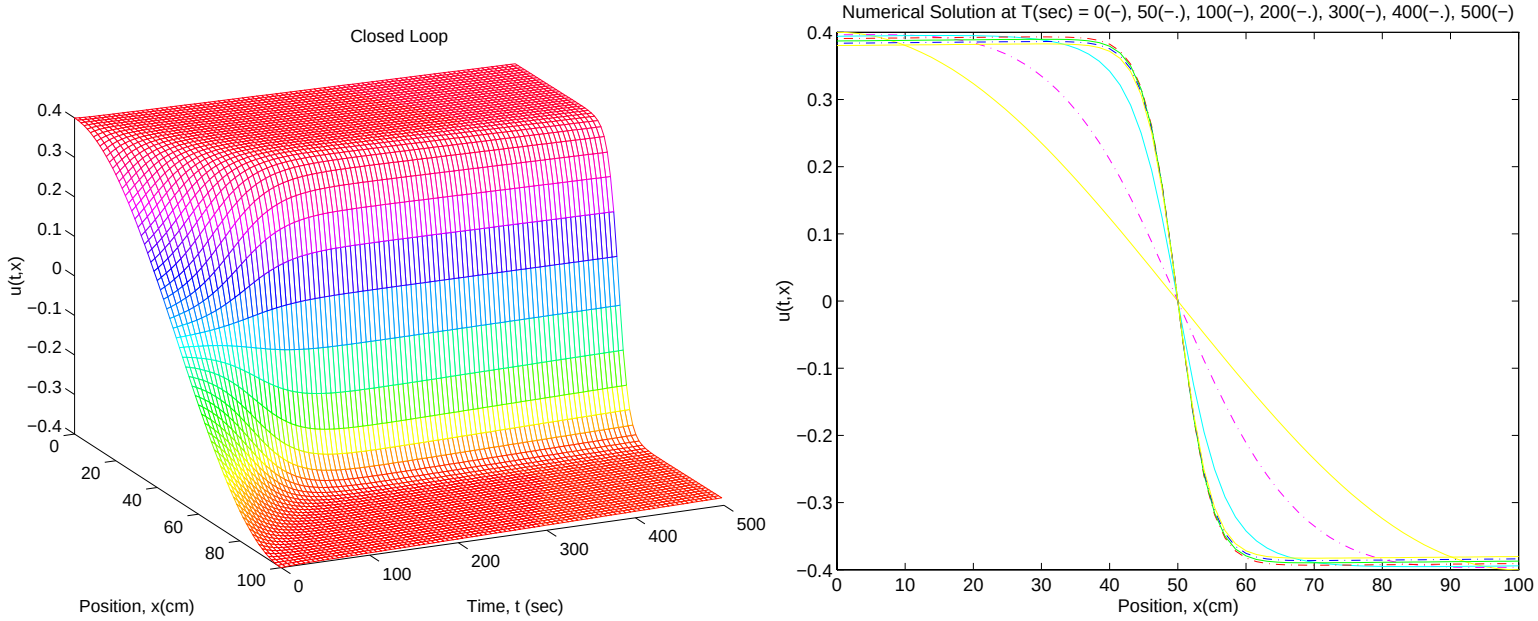


Figure 3.3.36: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = 5, $K_{.939}$, N = 64

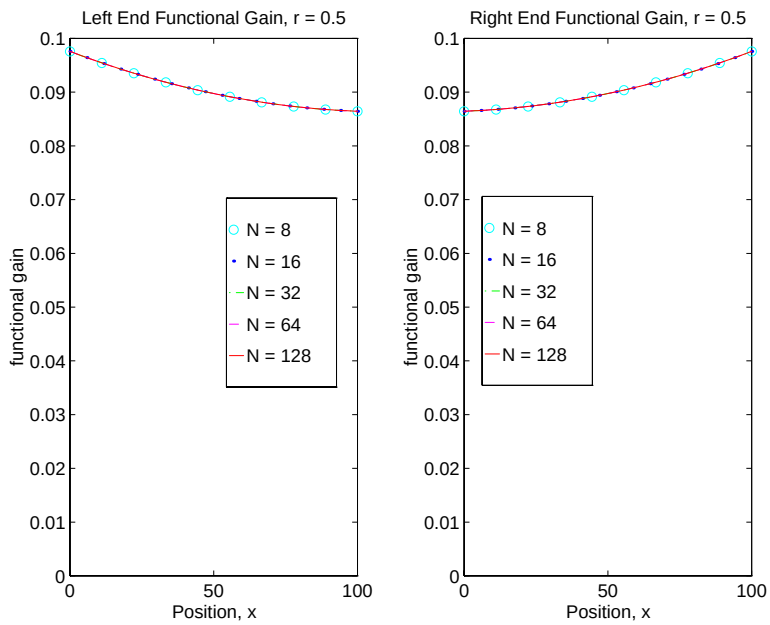


Figure 3.3.37: Functional gains, r = .5 for a 1m Al rod with 10m Fe films

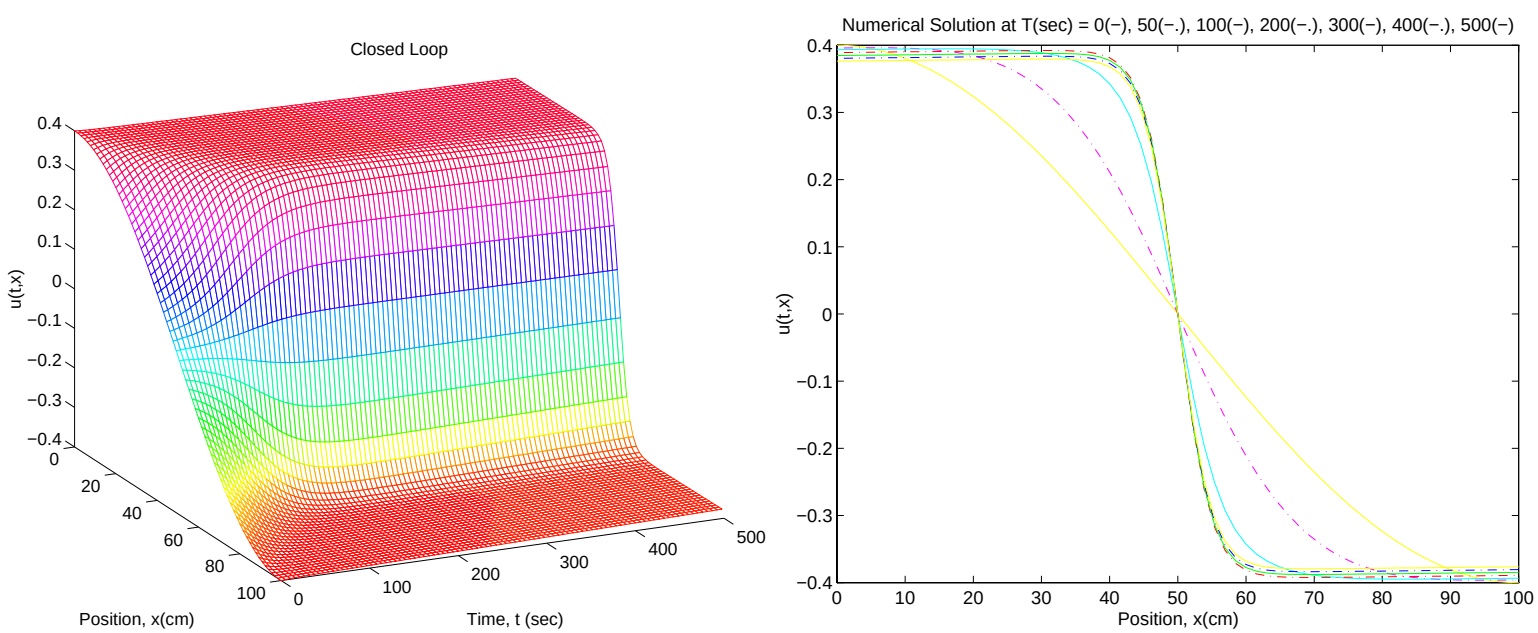


Figure 3.3.38: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .5, $K_{.939}$, N = 64

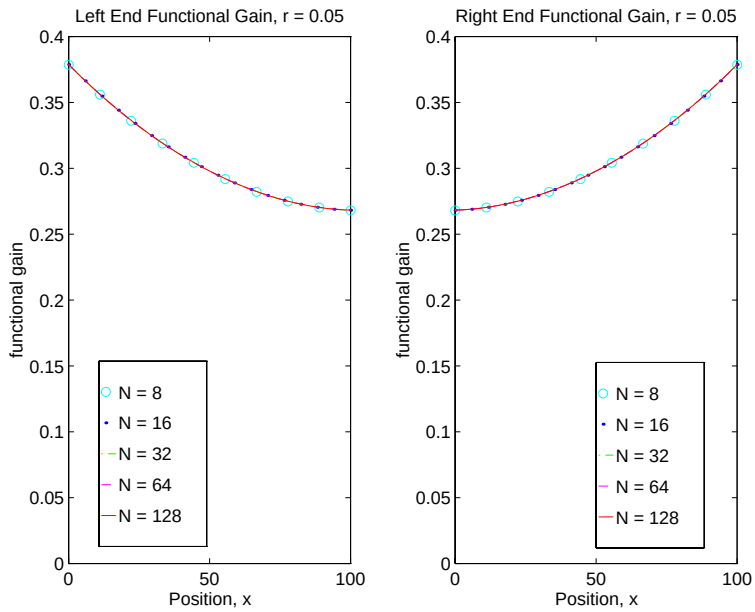


Figure 3.3.39: Functional gains, r = .05 for a 1m Al rod with 10m Fe films

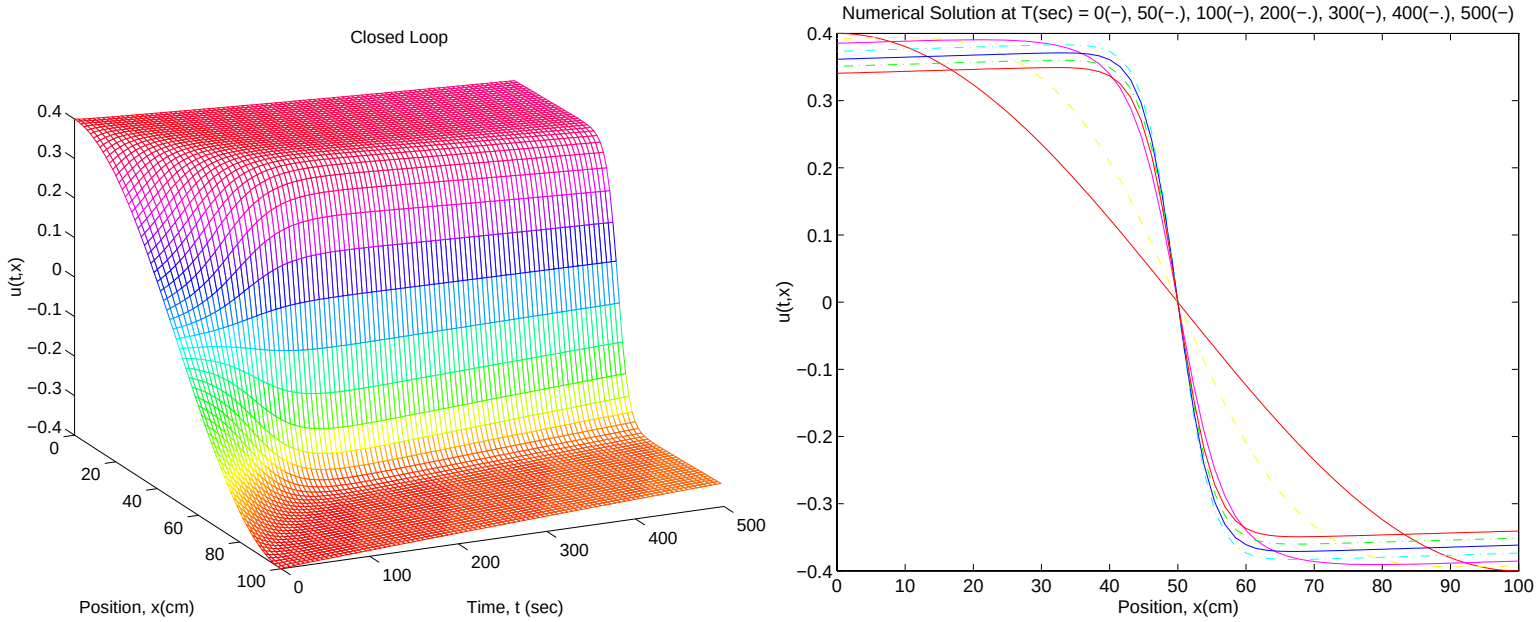


Figure 3.3.40: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .05, $K_{.939}$, N = 64

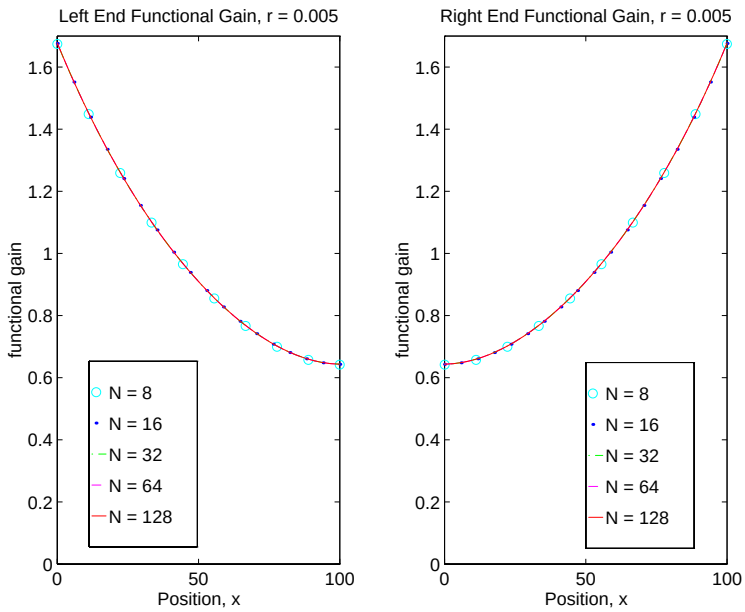


Figure 3.3.41: Functional gains, r = .005 for a 1m Al rod with 10m Fe films

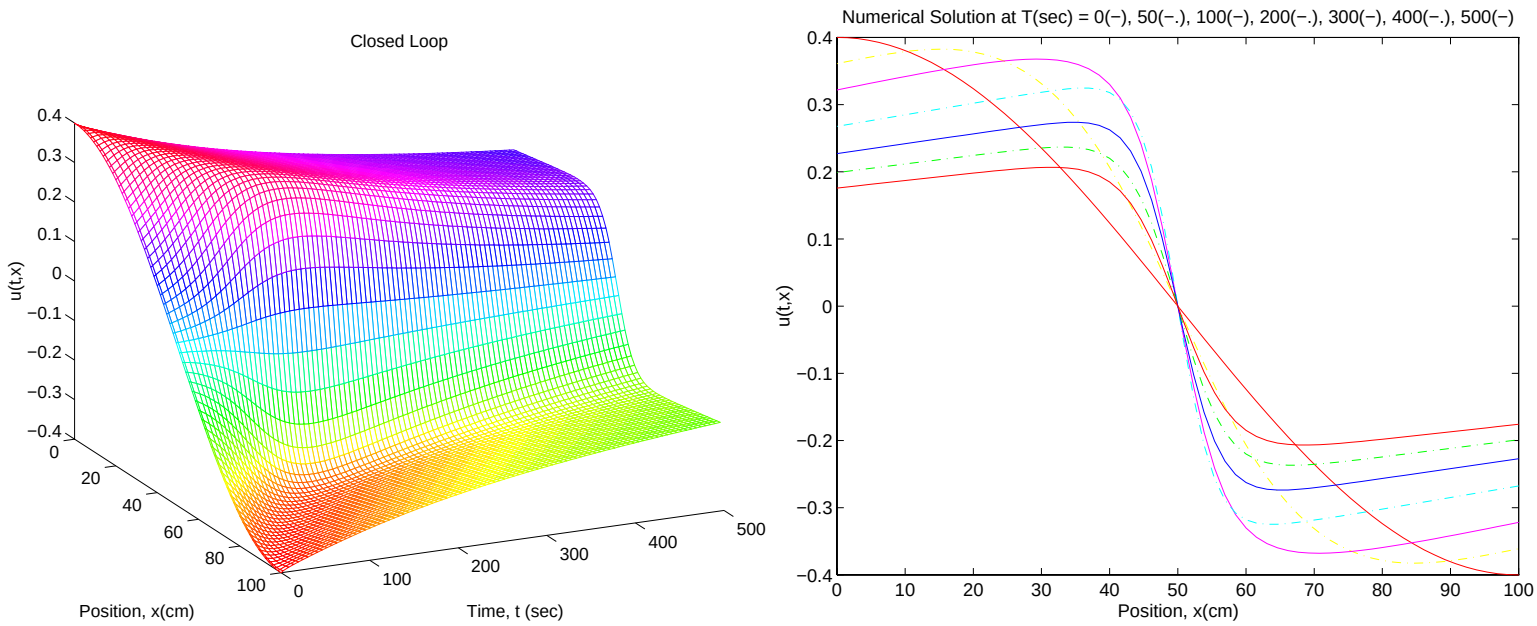


Figure 3.3.42: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .005, $K_{.939}$, N = 64

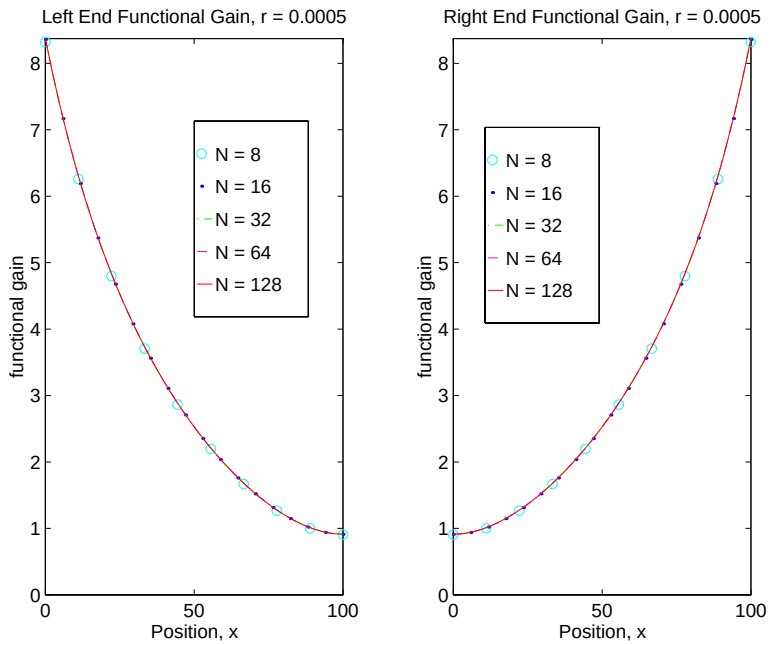


Figure 3.3.43: Functional gains, r = .0005 for a 1m Al rod with 10m Fe films

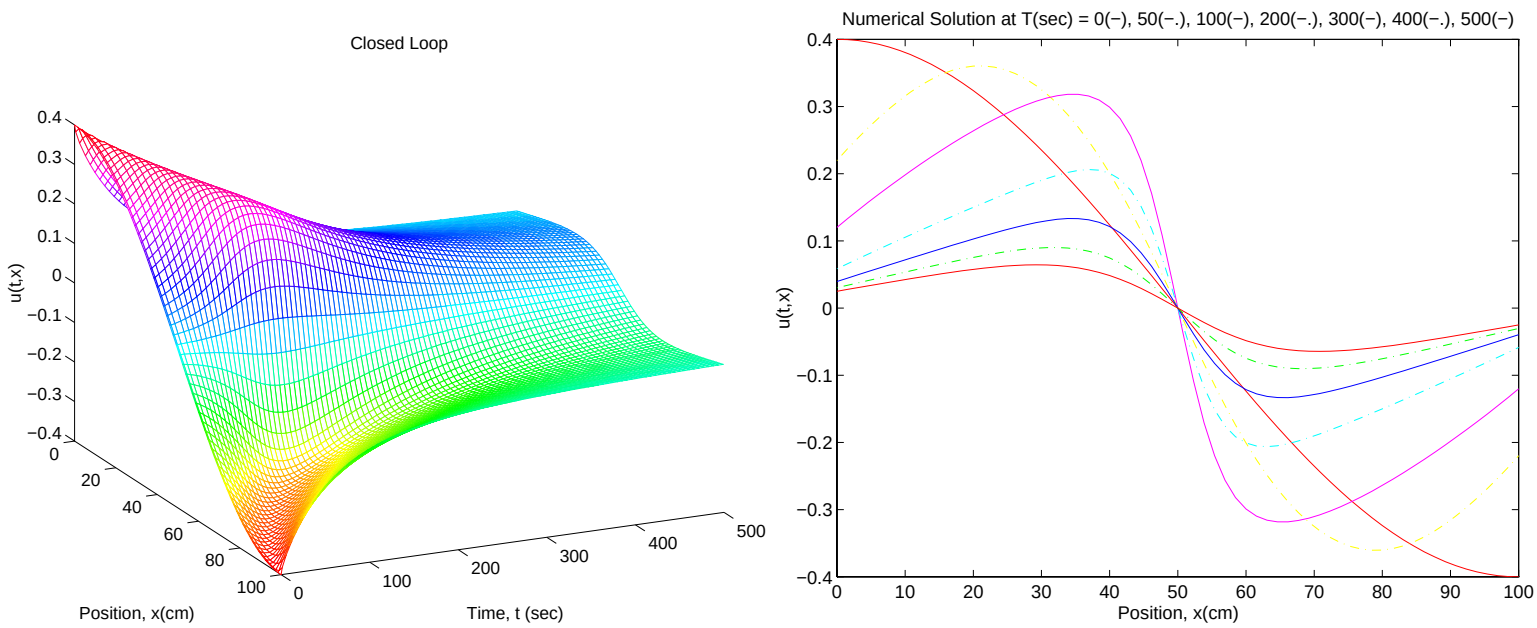


Figure 3.3.44: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .0005, $K_{.939}$, N = 64

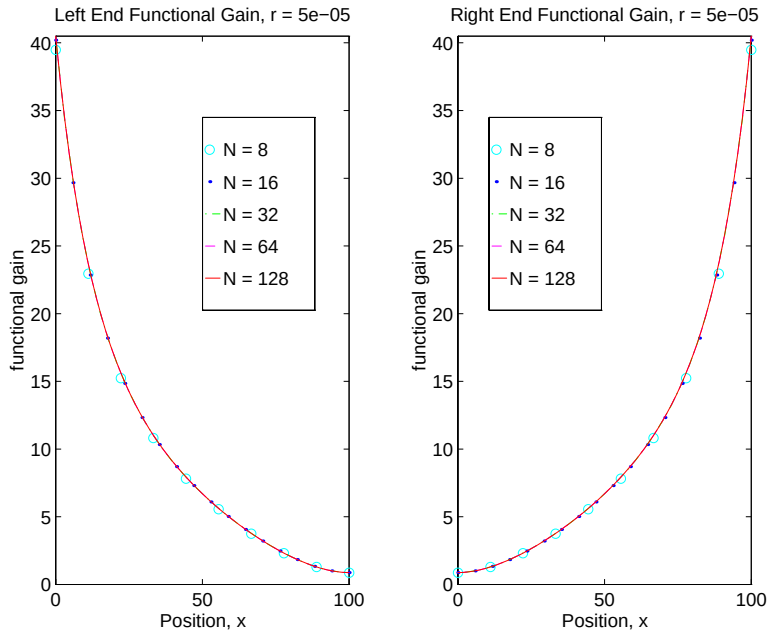


Figure 3.3.45: Functional gains, r = .00005 for a 1m Al rod with 10m Fe films

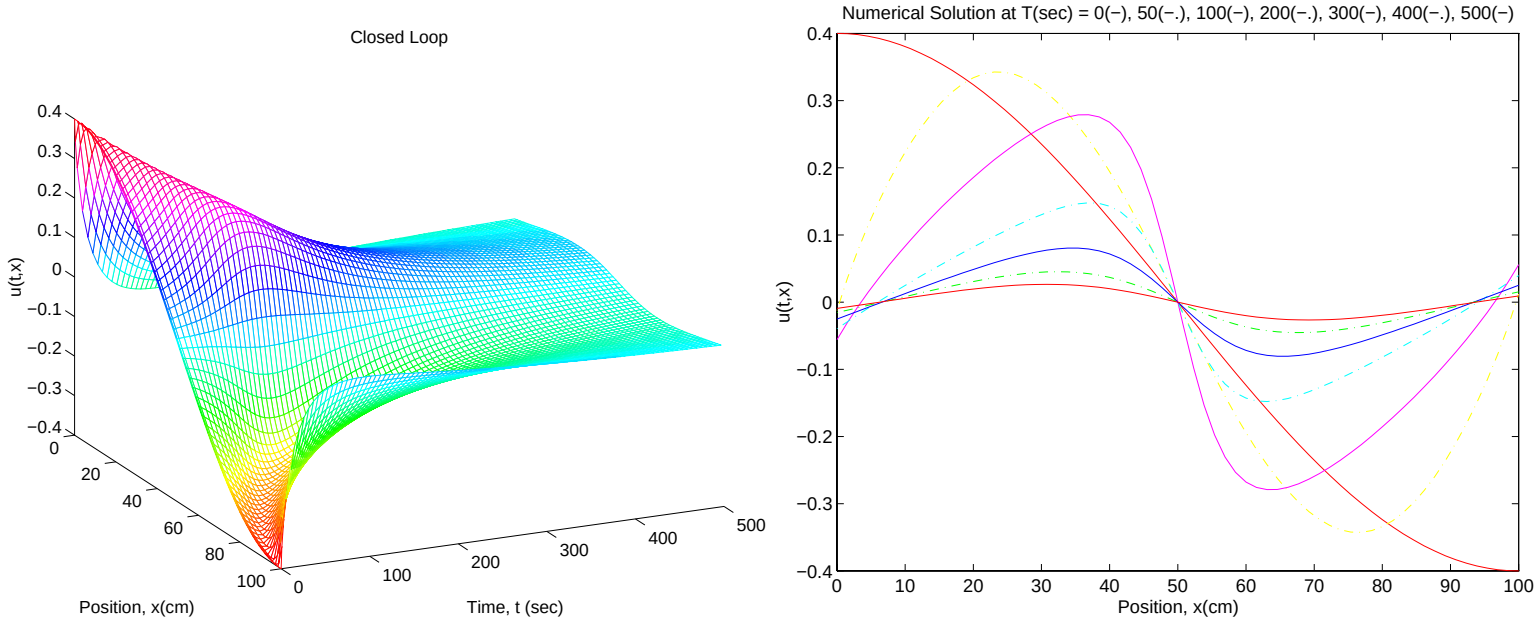


Figure 3.3.46: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .00005, $K_{.939}$, N = 64

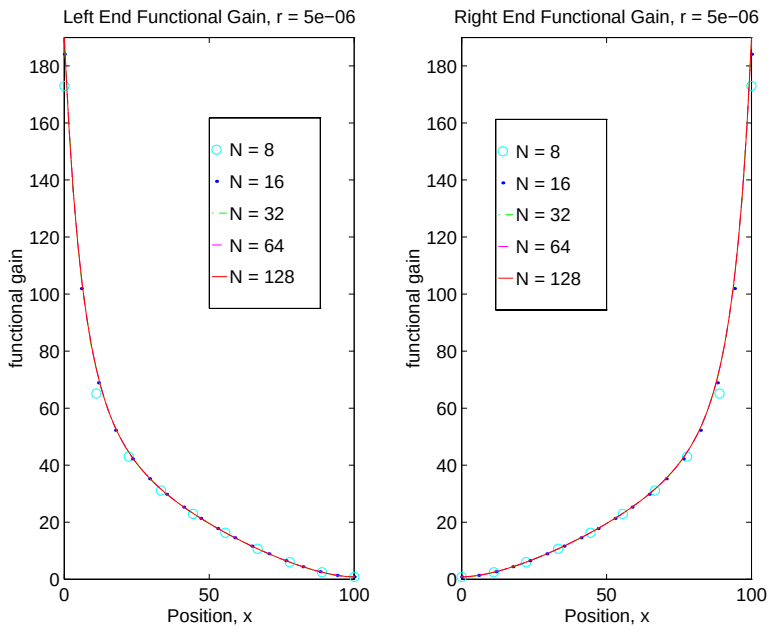


Figure 3.3.47: Functional gains, r = .000005 for a 1m Al rod with 10m Fe films

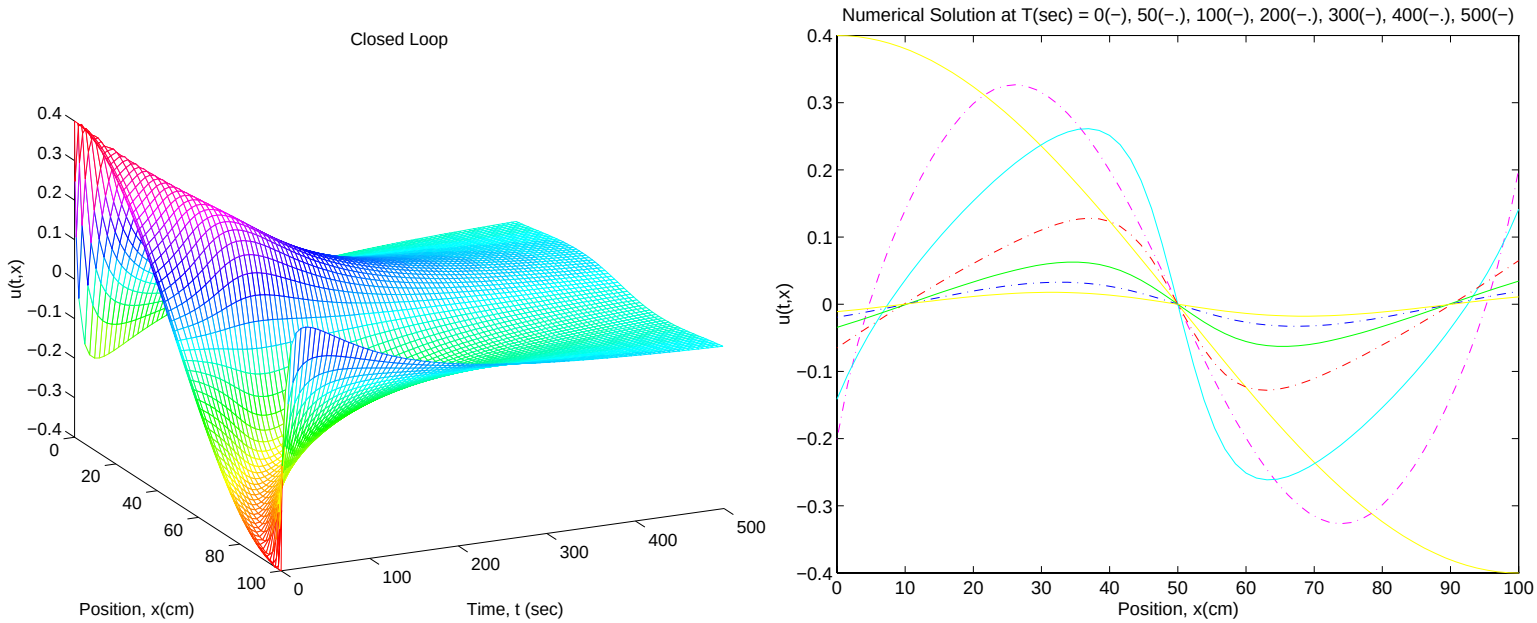


Figure 3.3.48: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .000005, $K_{.939}$, N = 64

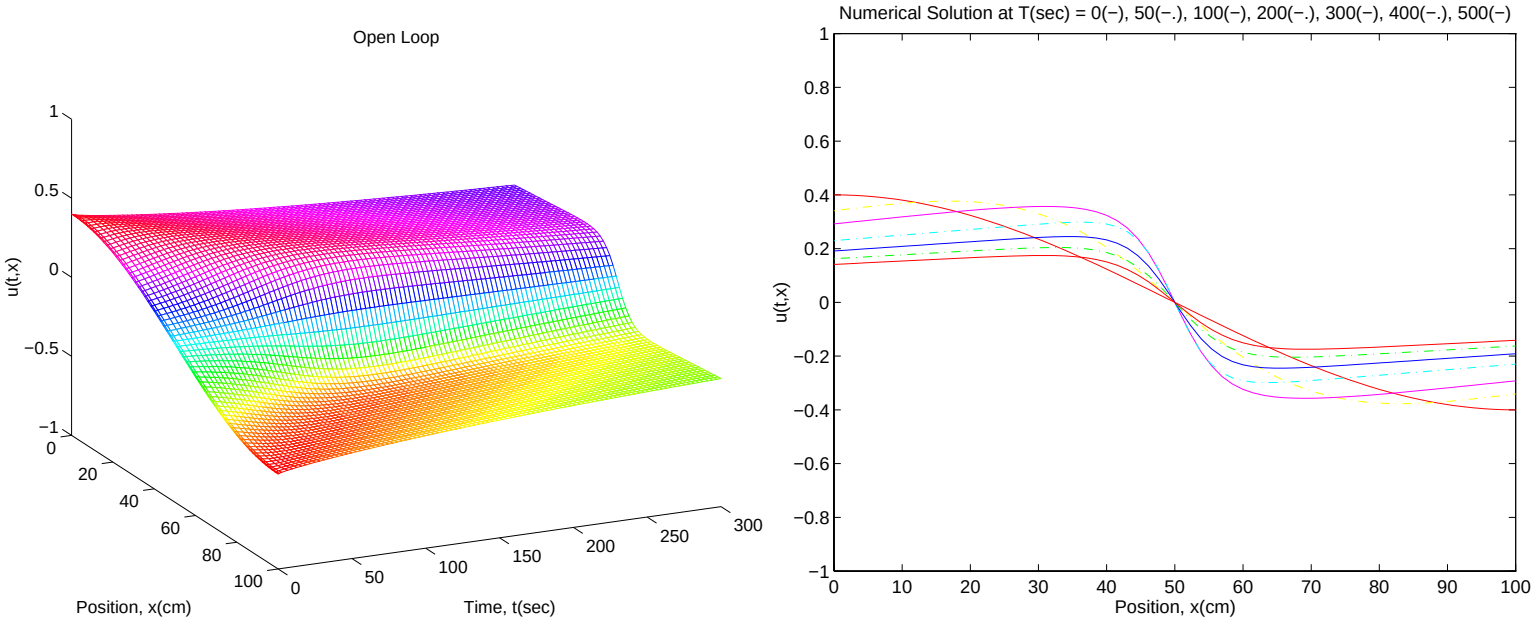


Figure 3.3.49: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 64

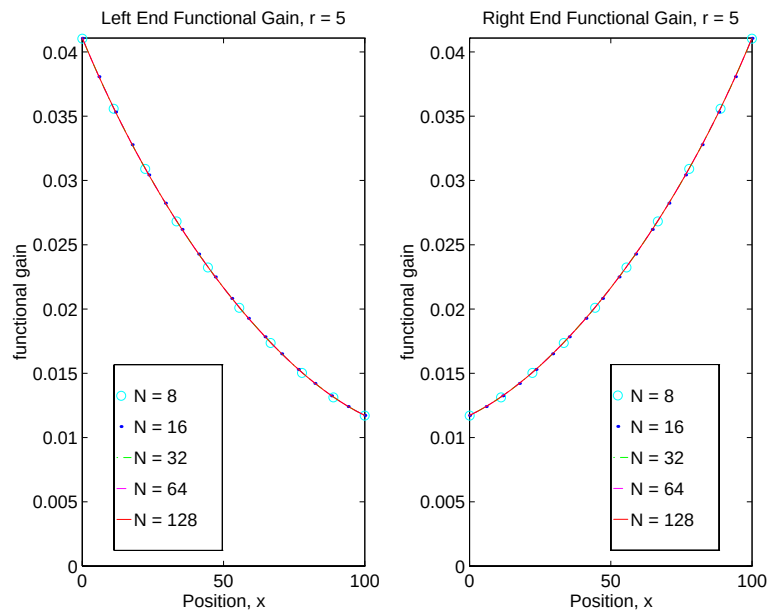


Figure 3.3.50: Functional gains, r = 5 for a 1m Al rod with 25cm Fe films

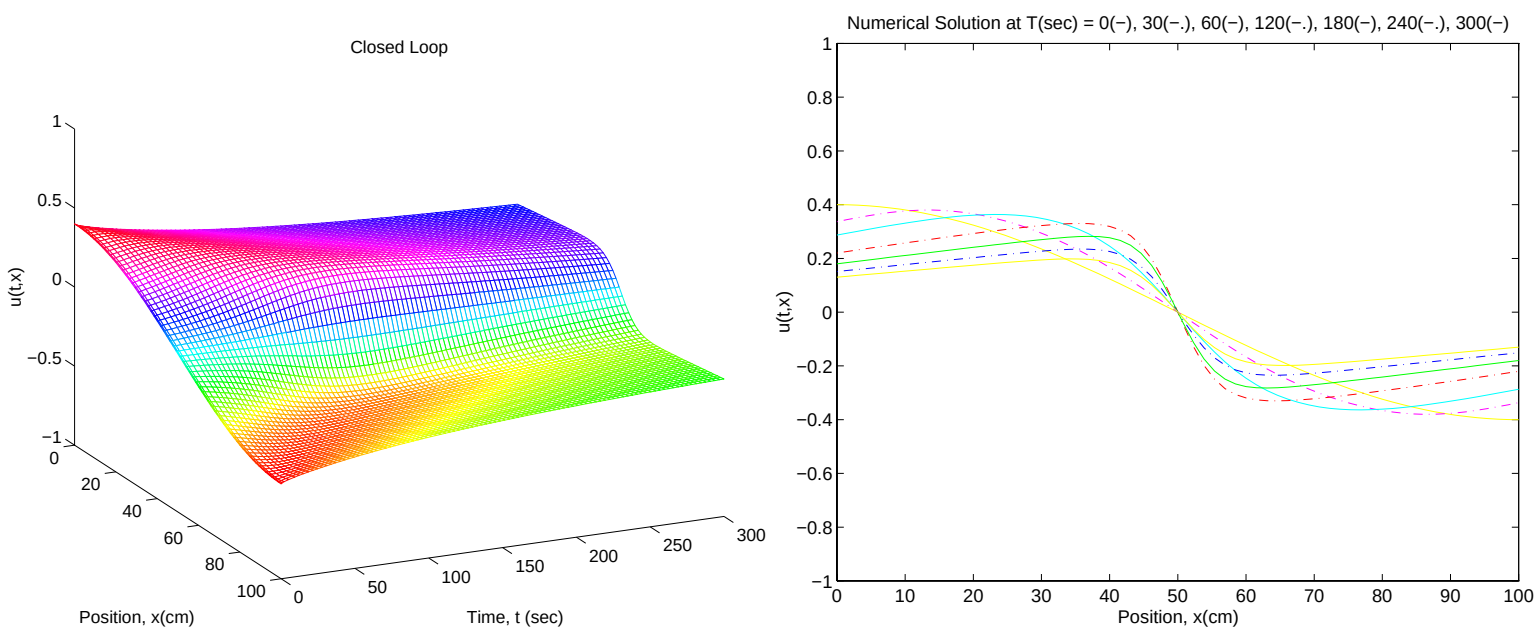


Figure 3.3.51: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = 5, $K_{.939}$, N = 64

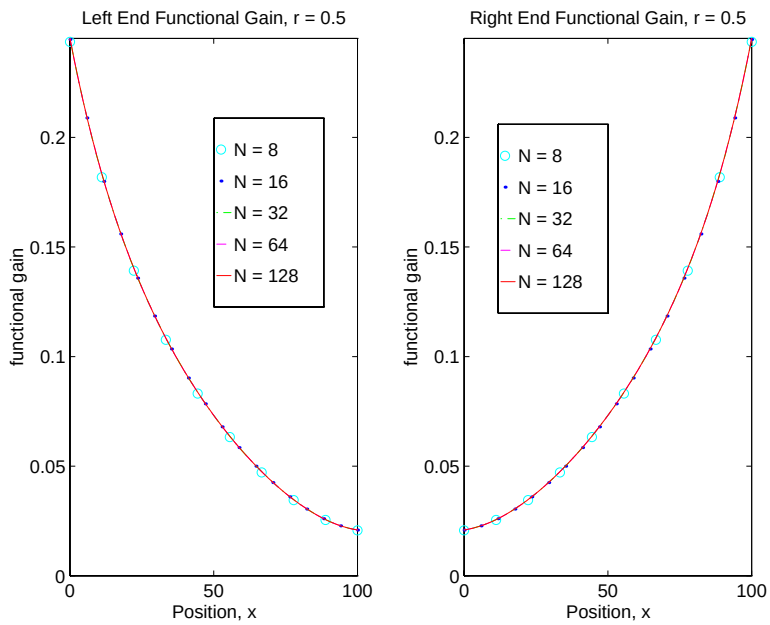


Figure 3.3.52: Functional gains, r = .5 for a 1m Al rod with 25cm Fe films

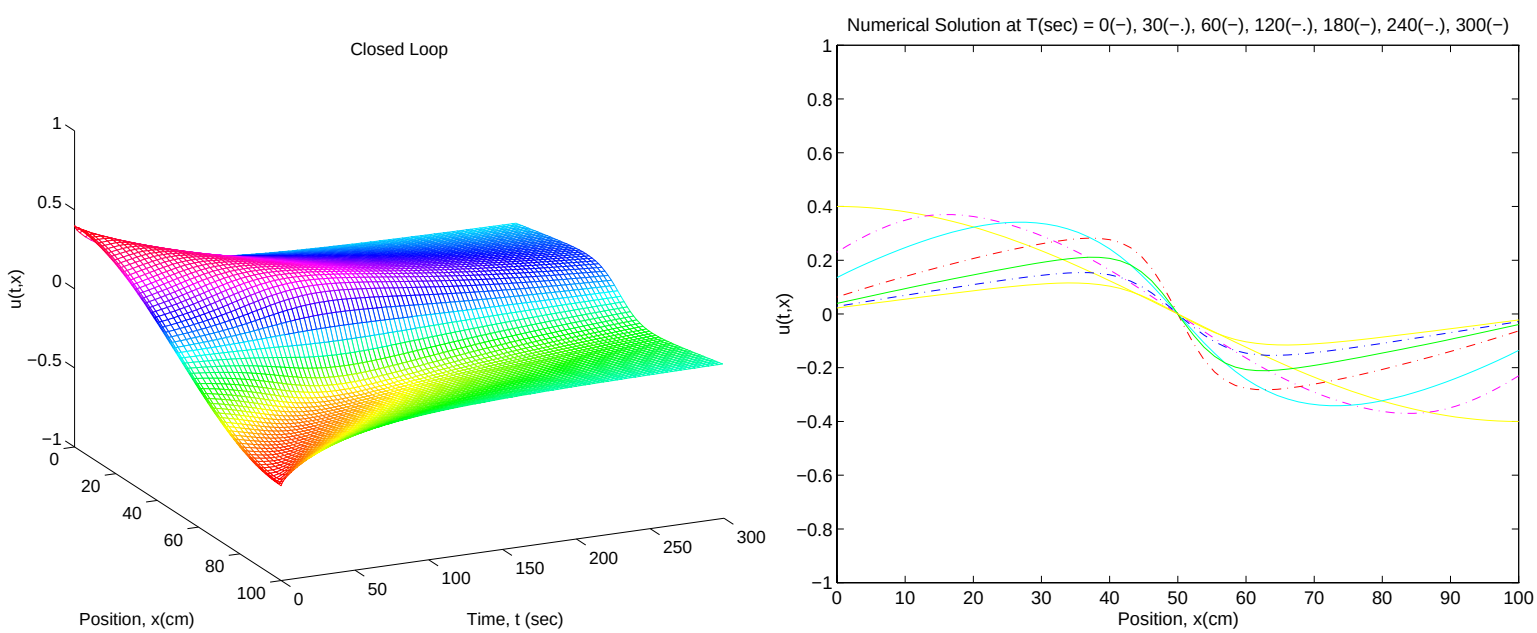


Figure 3.3.53: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .5, $K_{.939}$, N = 64

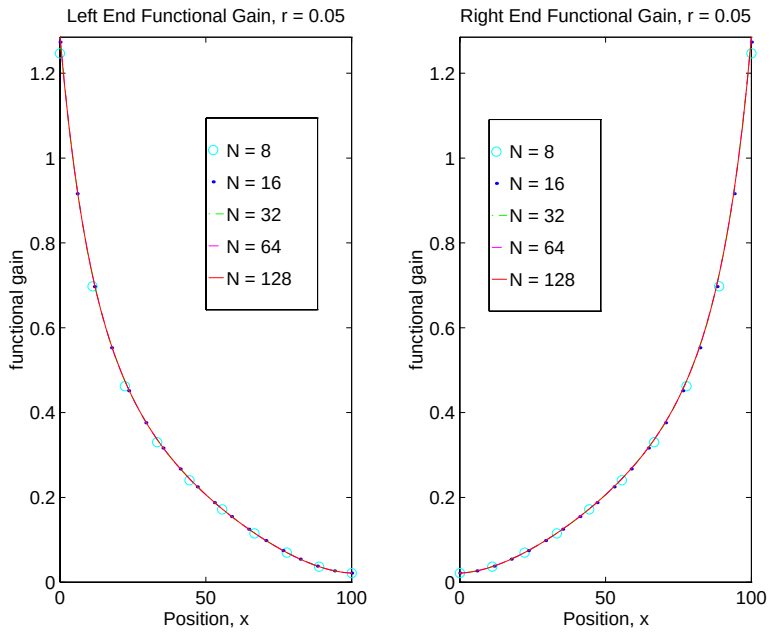


Figure 3.3.54: Functional gains, r = .05 for a 1m Al rod with 25cm Fe films

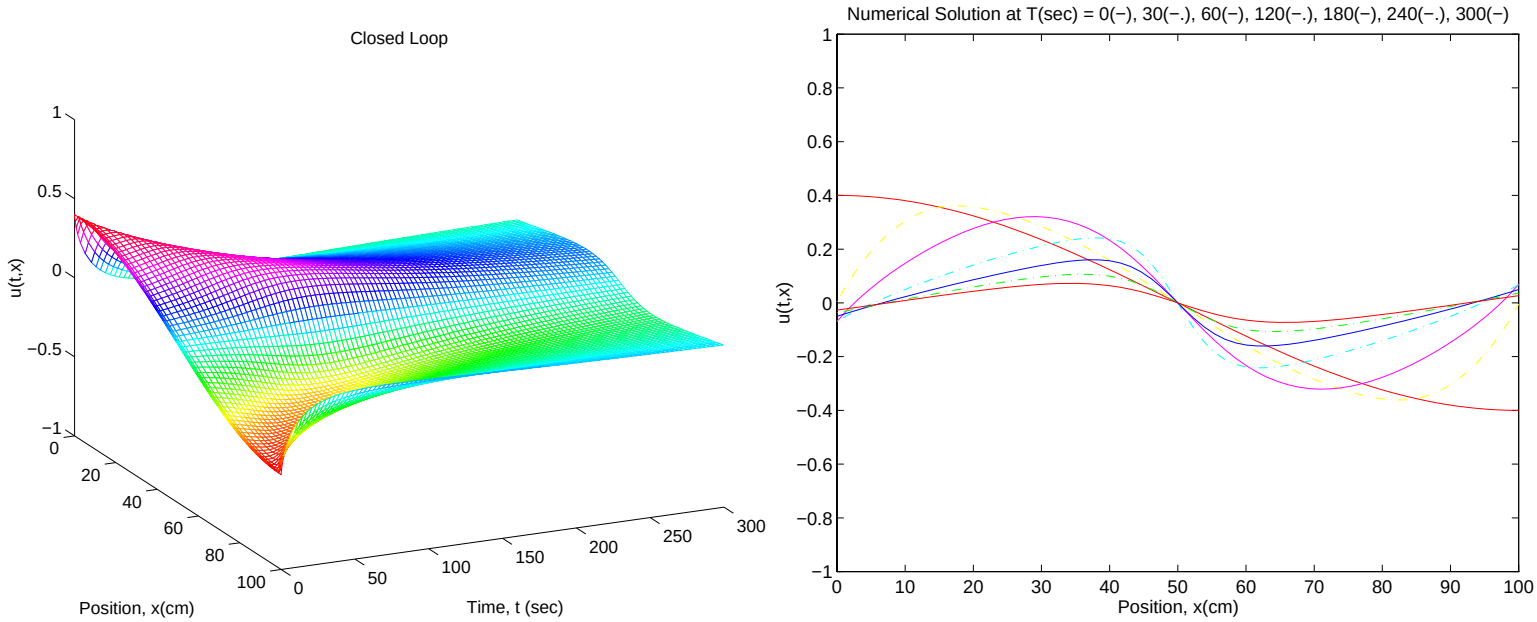


Figure 3.3.55: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .05, $K_{.939}$, N = 64

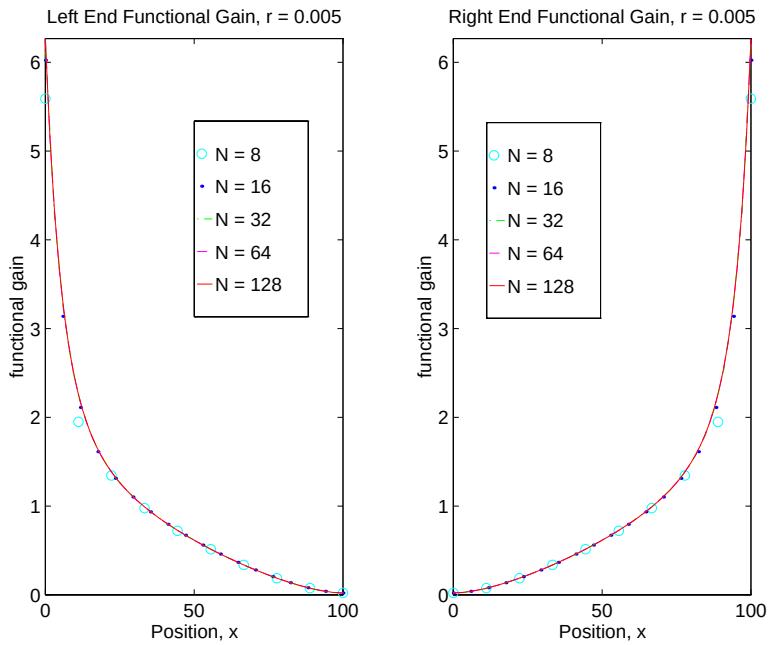


Figure 3.3.56: Functional gains, r = .005 for a 1m Al rod with 25cm Fe films

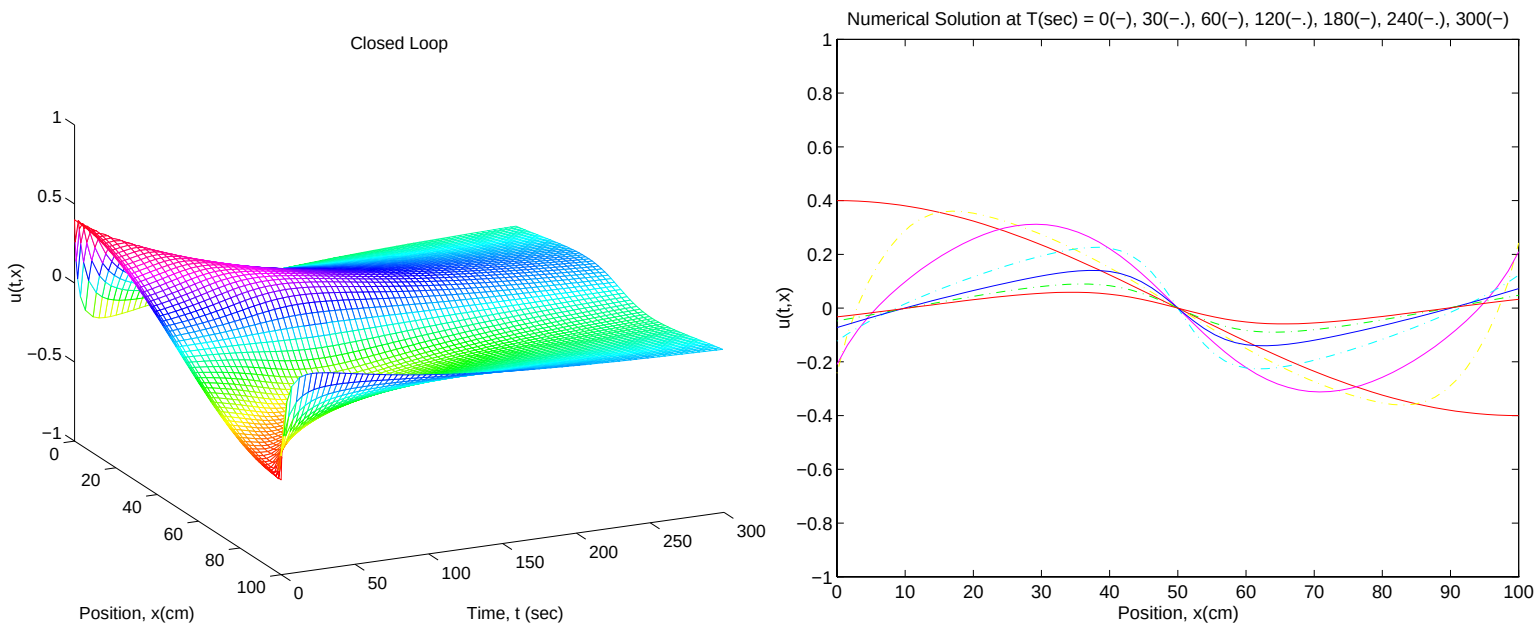


Figure 3.3.57: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .005, $K_{.939}$, N = 64

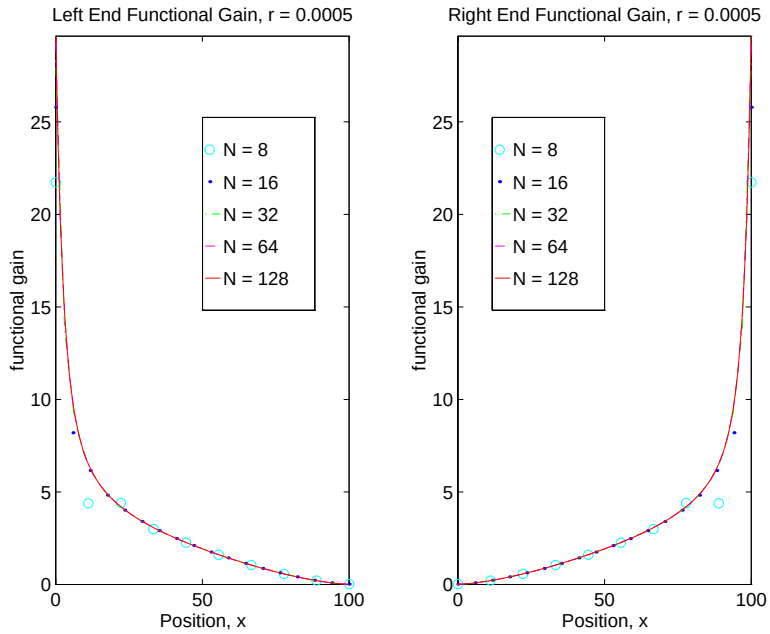


Figure 3.3.58: Functional gains, r = .0005 for a 1m Al rod with 25cm Fe films

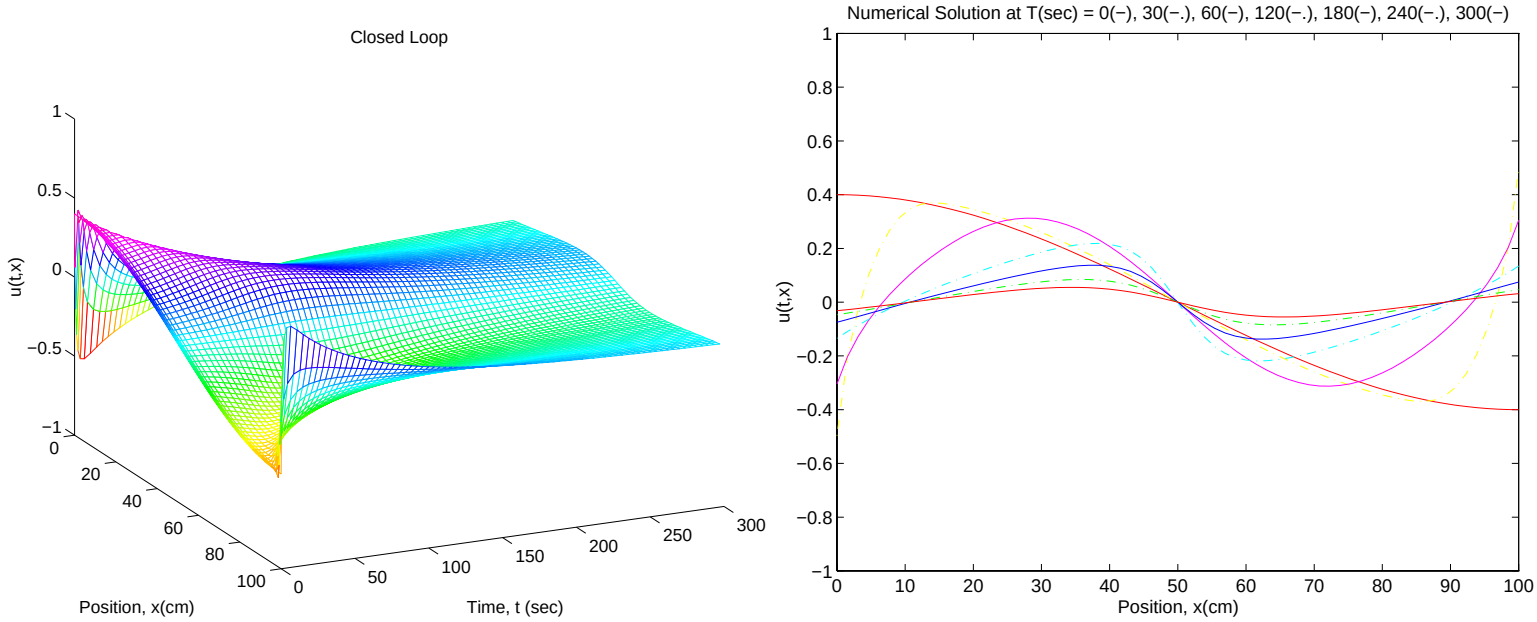


Figure 3.3.59: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .0005, $K_{.939}$, N = 64

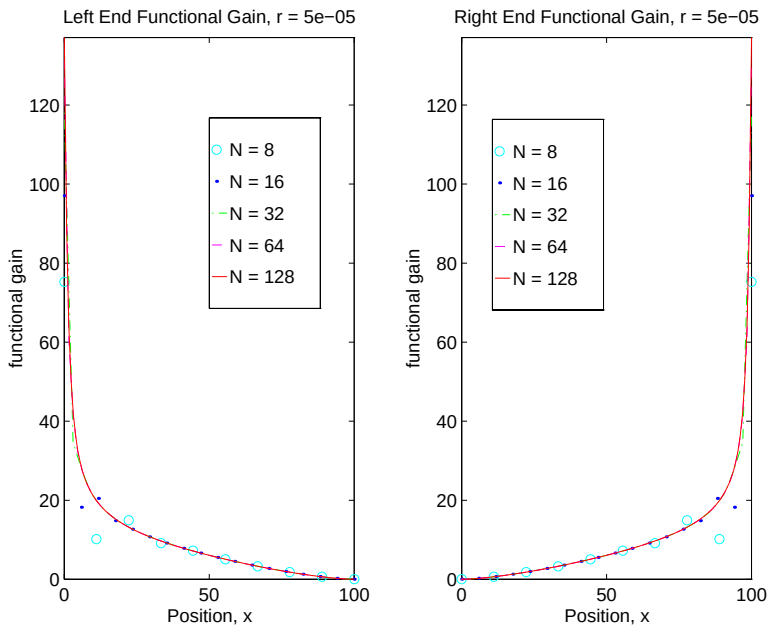


Figure 3.3.60: Functional gains, r = .00005 for a 1m Al rod with 25cm Fe films

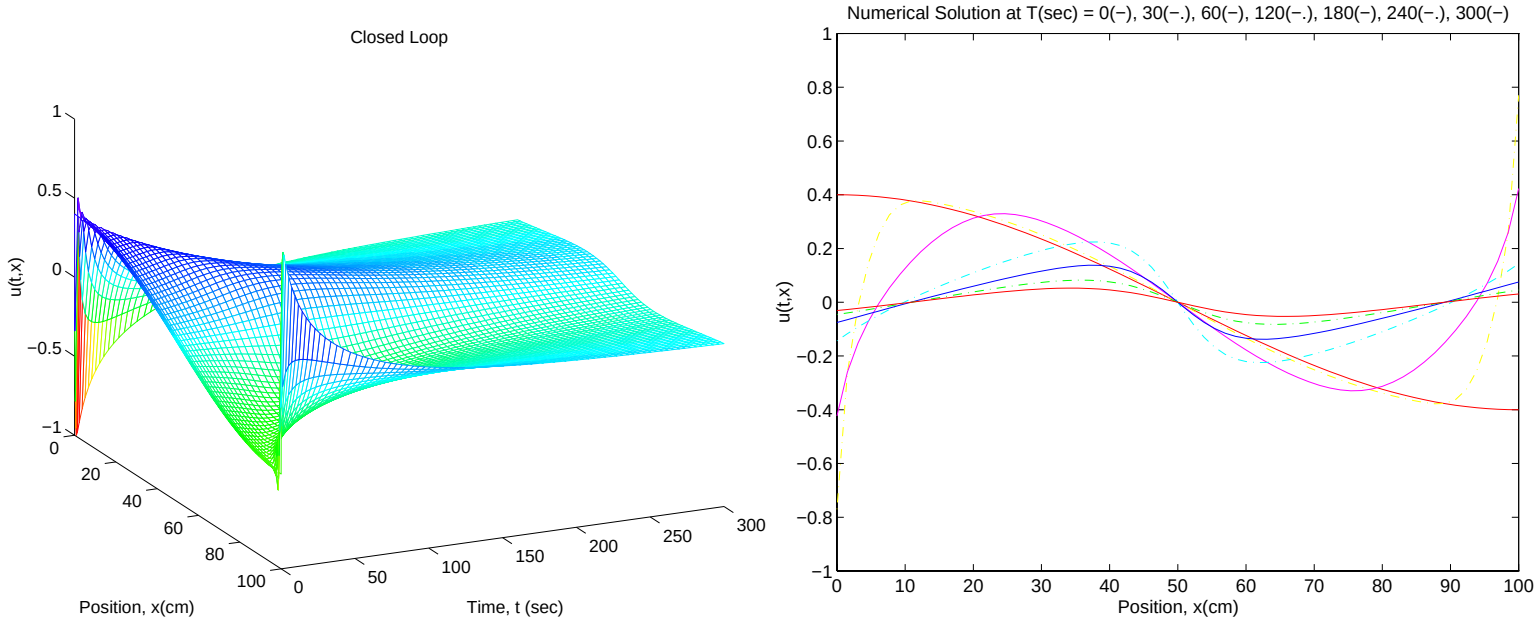


Figure 3.3.61: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .00005, $K_{.939}$, N = 64

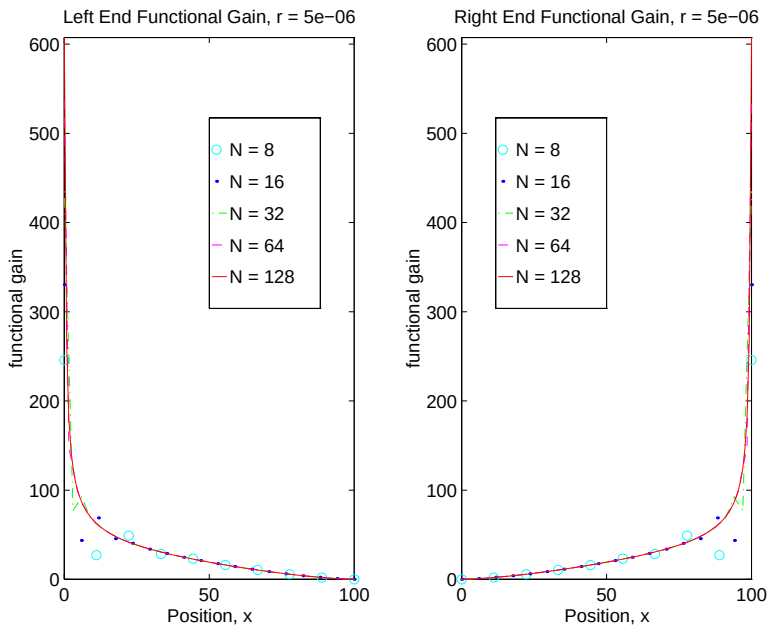


Figure 3.3.62: Functional gains, r = .000005 for a 1m Al rod with 25cm Fe films

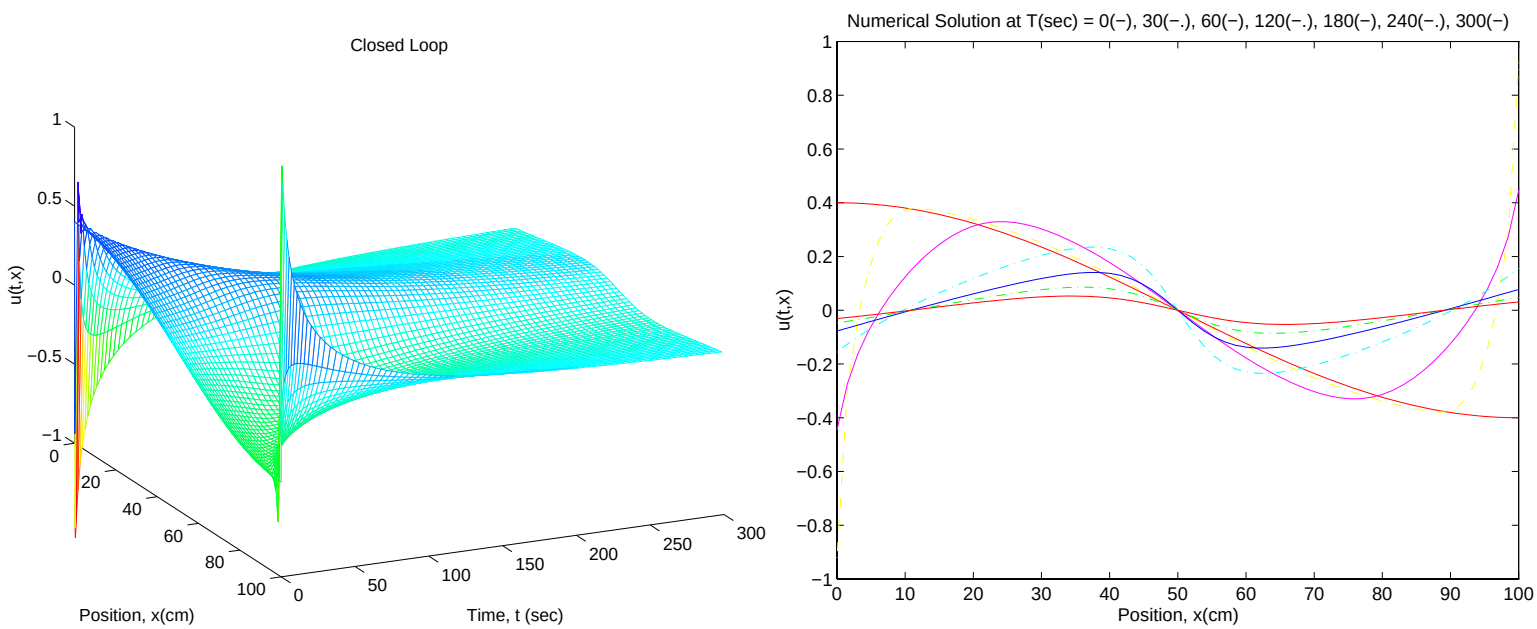


Figure 3.3.63: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, r = .000005, $K_{.939}$, N = 64

### 3.3.3 Neumann to Dirichlet Conditions

The following simulations are conducted to investigate the response and performance for the same aluminum rod with various iron films and initial function, $u_o(x) = .4\cos(\frac{\pi}{100}x)$. The weighting constant, $r$, is fixed at .0005, since it appears to be the largest value which shows significant control for the approximate Neumann and Dirichlet boundary conditions. In order to approximate Neumann boundary conditions, $\frac{\kappa_1}{L_1}$ and $\frac{\kappa_2}{L_2}$ are driven toward zero in (3.21) so that $u_x(t,0)$ and $u_x(t,L)$ become large in comparison to $u(t,0)$ and $u(t,L)$. Conversely, to approximate Dirichlet boundary conditions, $\frac{\kappa_1}{L_1}$ and $\frac{\kappa_2}{L_2}$ are driven toward infinity in (3.21) so that $u(t,0)$ and $u(t,L)$ become large in comparison to $u_x(t,0)$ and $u_x(t,L)$. In both cases, only $L_1$ and $L_2$ are varied, leaving $\kappa_1$ and $\kappa_2$ unchanged.

The functional gains corresponding to the gain matrix, $K_{.939}$, are plotted for films of length 10m (Fig. 3.3.64), 1m (Fig. 3.3.68), 50cm (Fig. 3.3.72), 25cm (Fig. 3.3.76), 10cm (Fig. 3.3.80), and 5cm (Fig. 3.3.84). The functional gain values at the boundaries increase from 8.37 to 45.75 as the film lengths are decreased from ten meters to five centimeters.

Figures 3.3.65, 3.3.69, 3.3.73, 3.3.77, 3.3.81, and 3.3.85 are the open loop simulations for film lengths of 10m, 1m, 50cm, 25cm, 10cm, and 5cm, respectively. The boundary conditions move from approximate Neumann conditions ($L_1 = L_2 = 10m$) to approximate Dirichlet conditions ($L_1 = L_2 = 5cm$). As predicted by the previous experiments, the open loop numerical solutions show an increase in the rate of decay toward the steady state solution of zero as the film lengths decrease.

Figures 3.3.66, 3.3.70, 3.3.74, 3.3.78, 3.3.82, and 3.3.86 are the corresponding closed loop simulations. The optimal gain used is $K_{.939}$. As the film lengths decrease, the closed loop simulations show an increase in the initial values at the boundaries after the start of the simulation. This is predicted by the functional gain values. However, these 'spikes' rapidly smooth out and the rate at which the numerical solutions decay toward zero is increased, as the film lengths are decreased.

Next, the gain calculated for the copper rod with aluminum films, $K_{1.14}$, is used to close the loop on these systems. Figures 3.3.67, 3.3.71, 3.3.75, 3.3.79, 3.3.83, and 3.3.87 are the closed loop simulations for film lengths of 10m, 1m, 50cm, 25cm, 10cm, and 5cm, respectively. The closed loop systems with $K_{1.14}$ are compared to their respective closed loop systems with the optimal gain $K_{.939}$. As the boundary conditions move toward the Dirichlet condition, the numerical solutions for the closed loop system with $K_{1.14}$ and numerical solutions for the closed loop system with $K_{.939}$ start to decay at nearly the same rate. For example, at five hundred seconds, when the film length is one meter (Figures: 3.3.70 and 3.3.71), the amplitude of the numerical solution with the optimal gain is .0178, while the amplitude of the numerical solution with $K_{1.14}$ is .0546. When the film length is 50cm (Figures: 3.3.74 and 3.3.75), at five hundred seconds, the amplitude of the numerical solution with $K_{.939}$ is .0165, while with $K_{1.14}$ it is .0361. For film lengths of 25cm, the corresponding amplitudes are .0157 and .0261. For film lengths of 10cm the corresponding amplitudes are .0153 and .0191. Finally, for film lengths of 5cm, they are .0153 and .0166.

Thus, the numerical solutions for both systems get closer as the film lengths decrease.

The sharp 'spikes' at the ends of the rod present in the closed loop solutions with the optimal gain are not present when $K_{1.14}$ is used to close the loop. However, as the boundary conditions approach Dirichlet conditions, the gain $K_{1.14}$ drives the numerical solutions toward zero at almost the same rate as the optimal gain, $K_{.939}$. This indicates that it can be used to form an effective control even though it is not the optimal control. A similar set of experiments with various gains is performed in the next section.

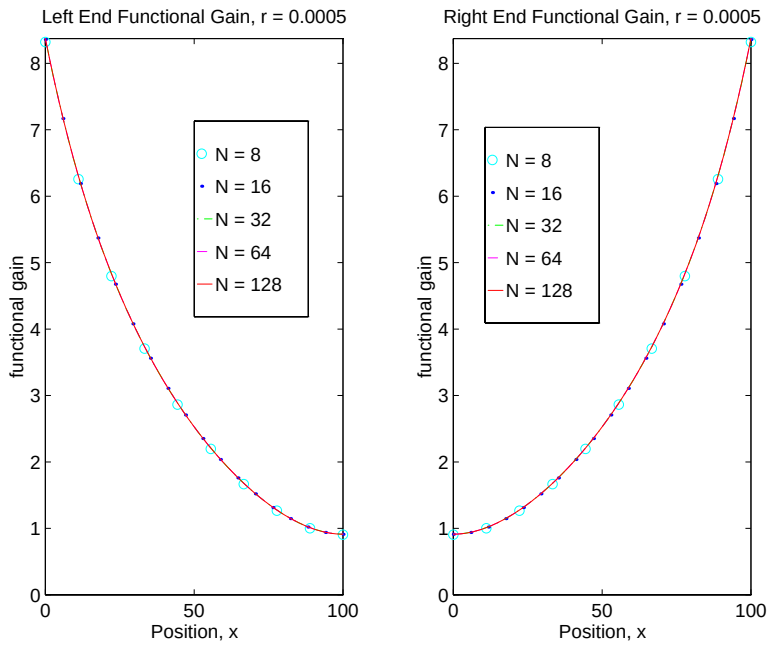


Figure 3.3.64: Functional gains, r = .0005 for a 1m Al rod with 10m Fe films

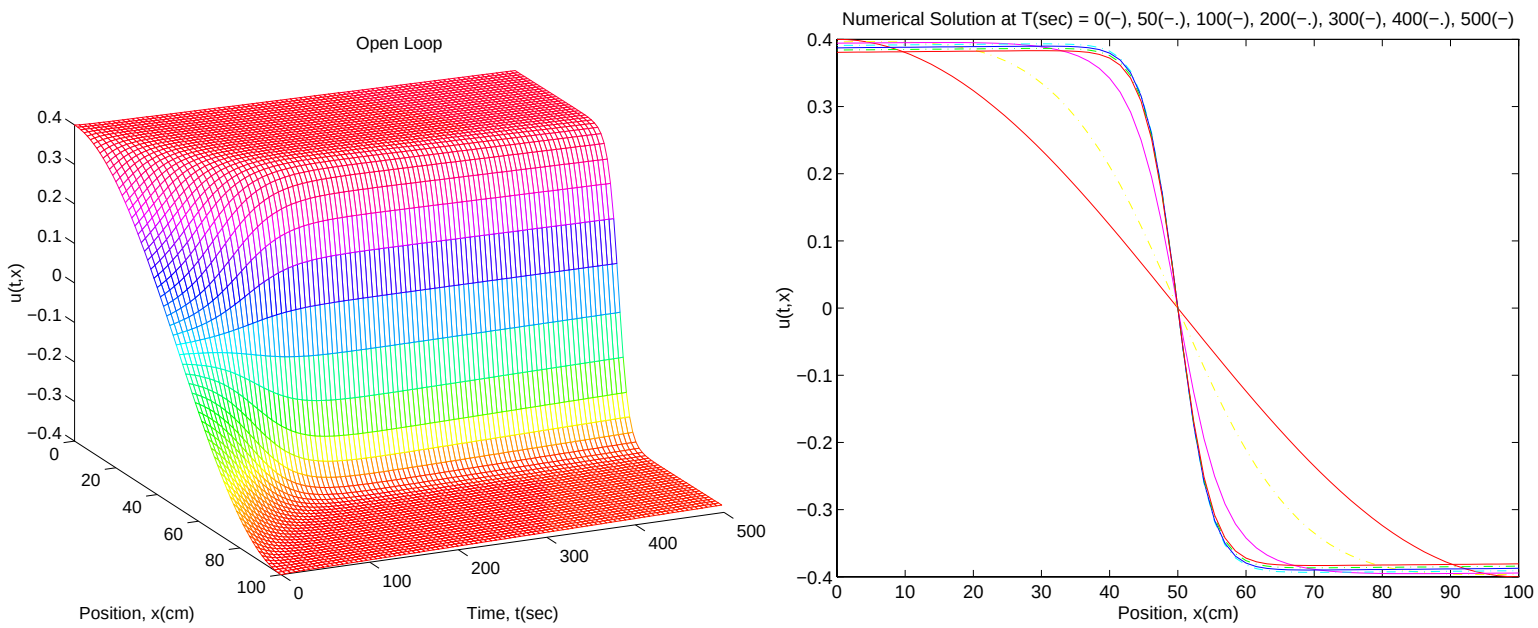


Figure 3.3.65: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 64

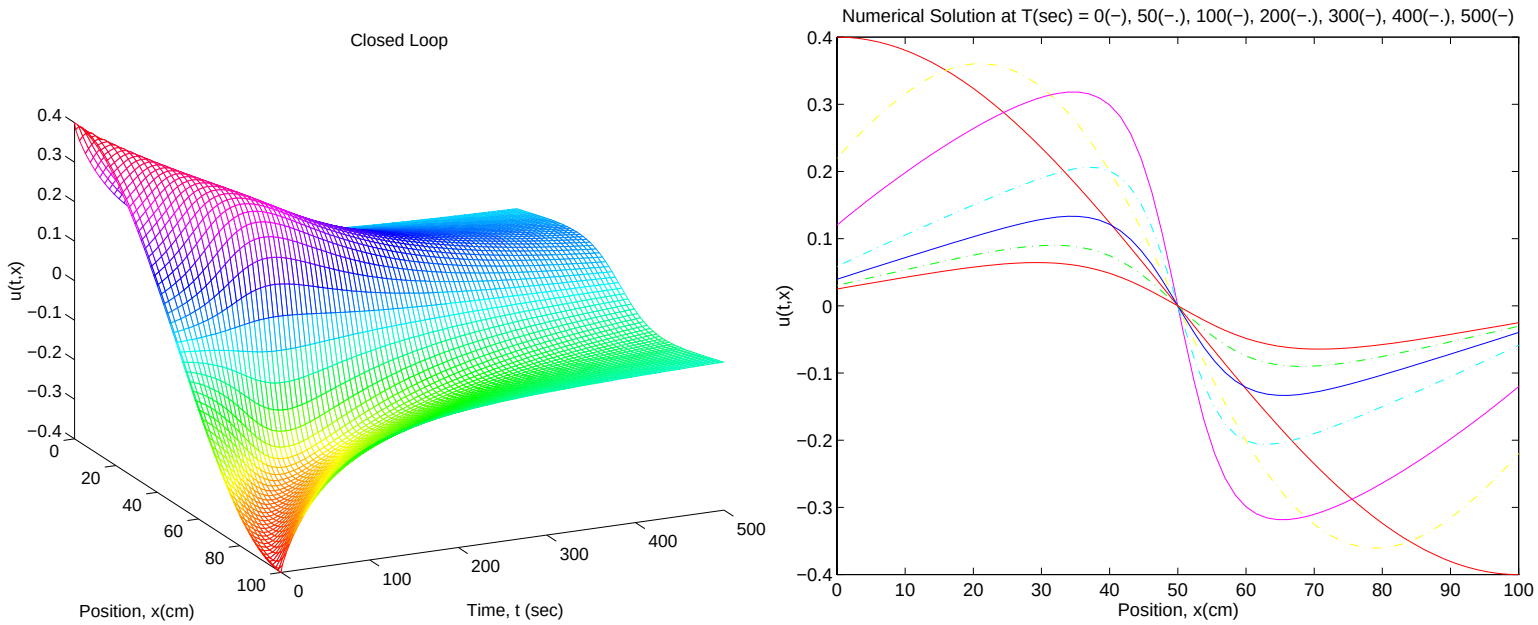


Figure 3.3.66: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 64

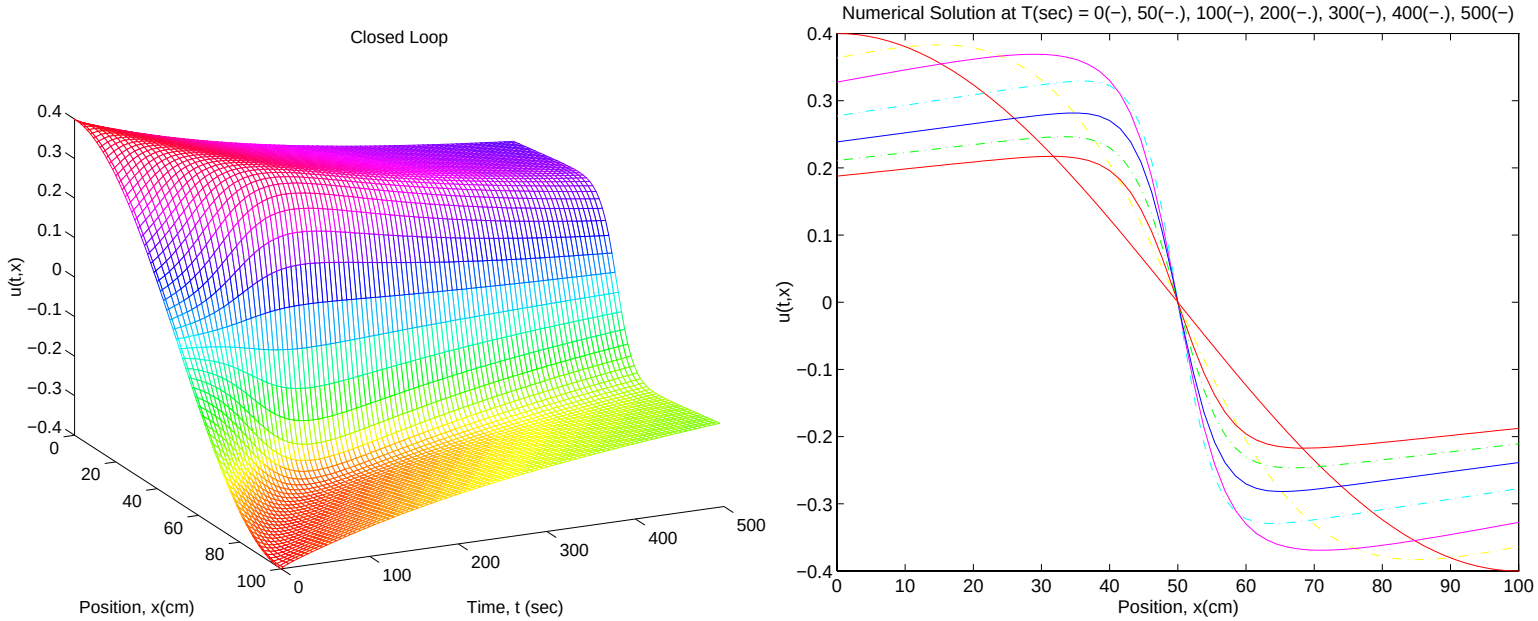


Figure 3.3.67: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 64

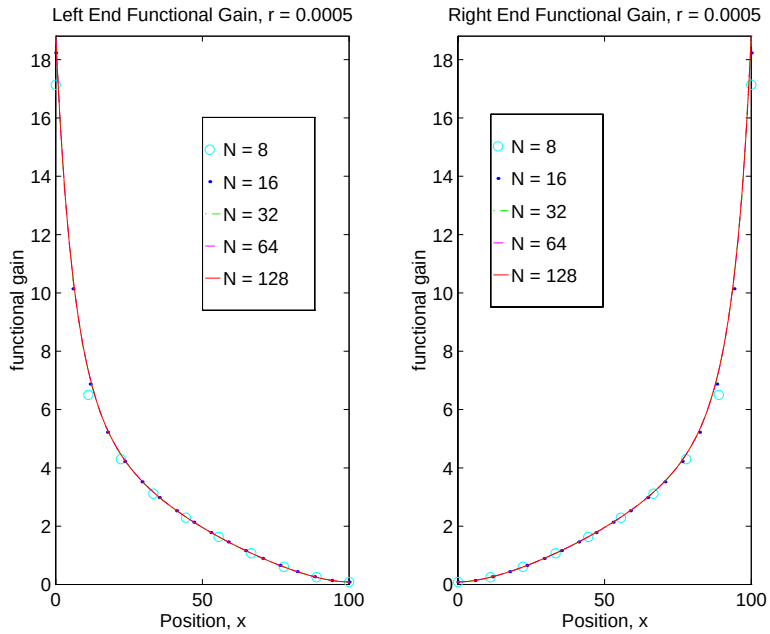


Figure 3.3.68: Functional gains, r = .0005 for a 1m Al rod with 1m Fe films

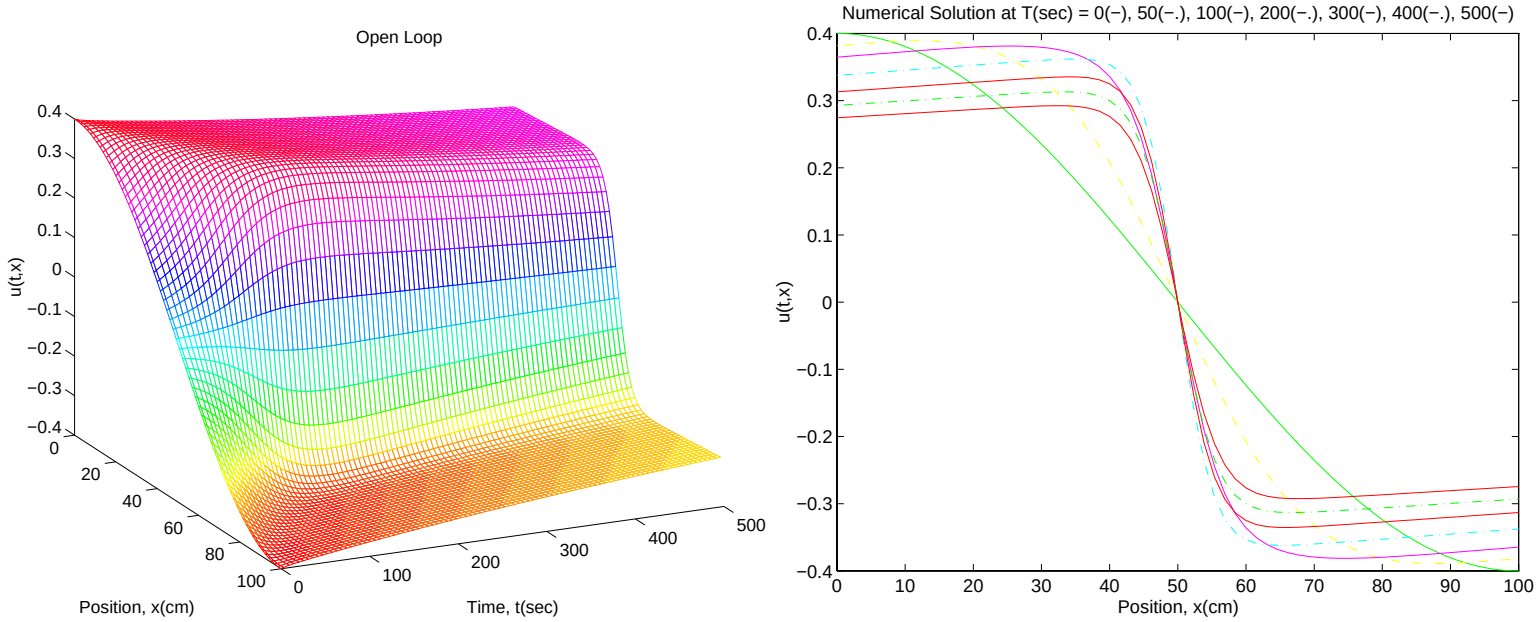


Figure 3.3.69: 1m Al rod, 1m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 64

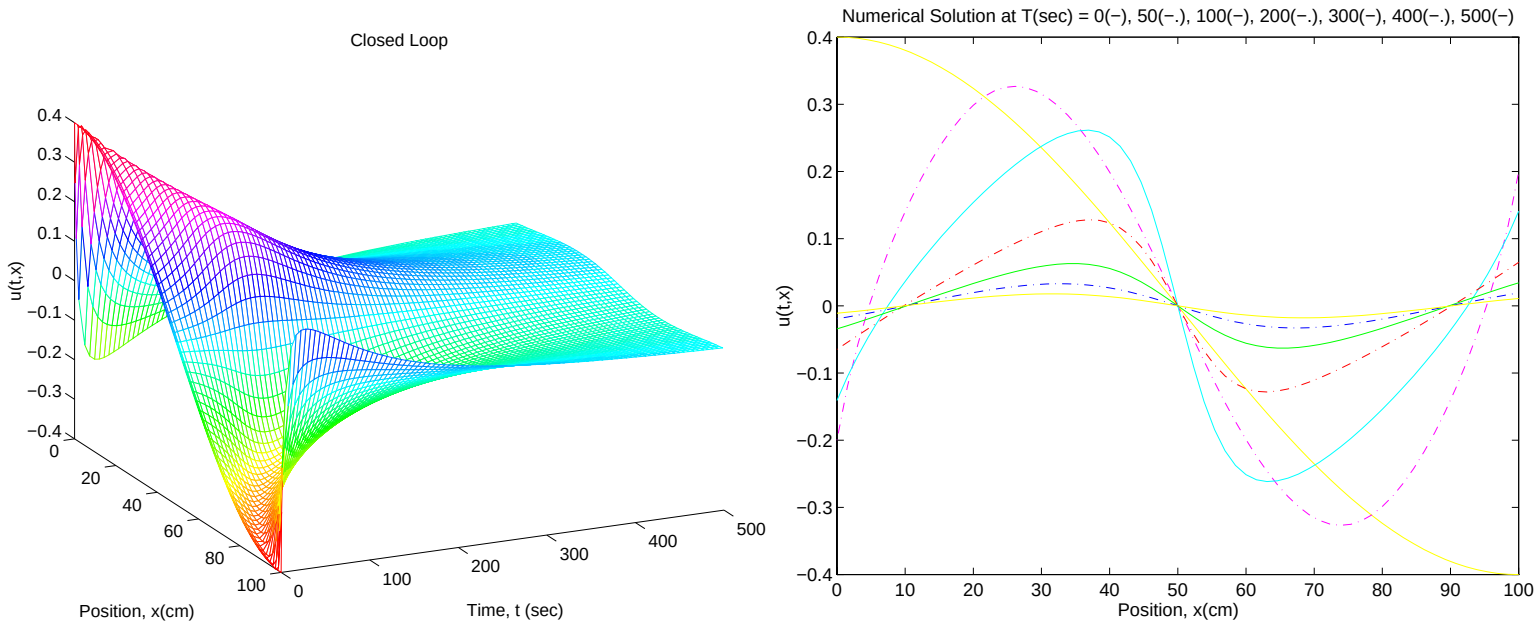


Figure 3.3.70: 1m Al rod, 1m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 64

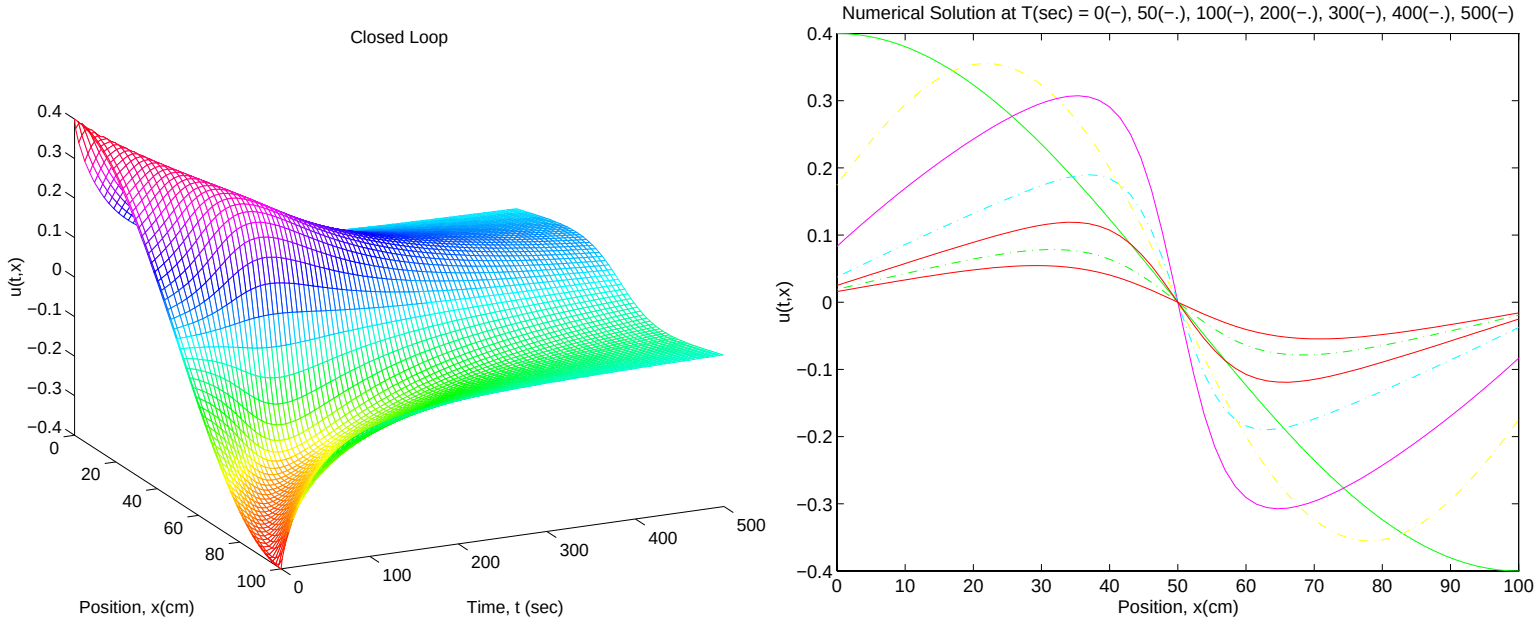


Figure 3.3.71: 1m Al rod, 1m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 64

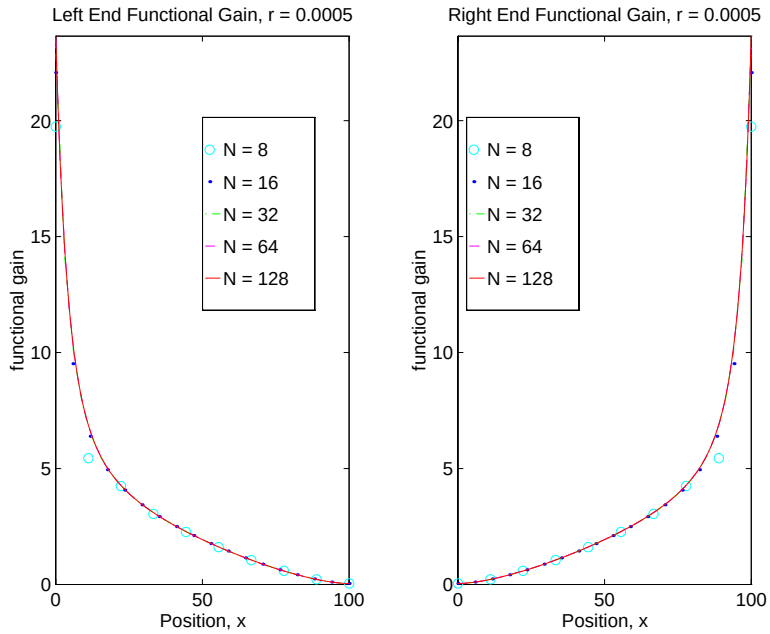


Figure 3.3.72: Functional gains, r = .0005 for a 1m Al rod with 50cm Fe films

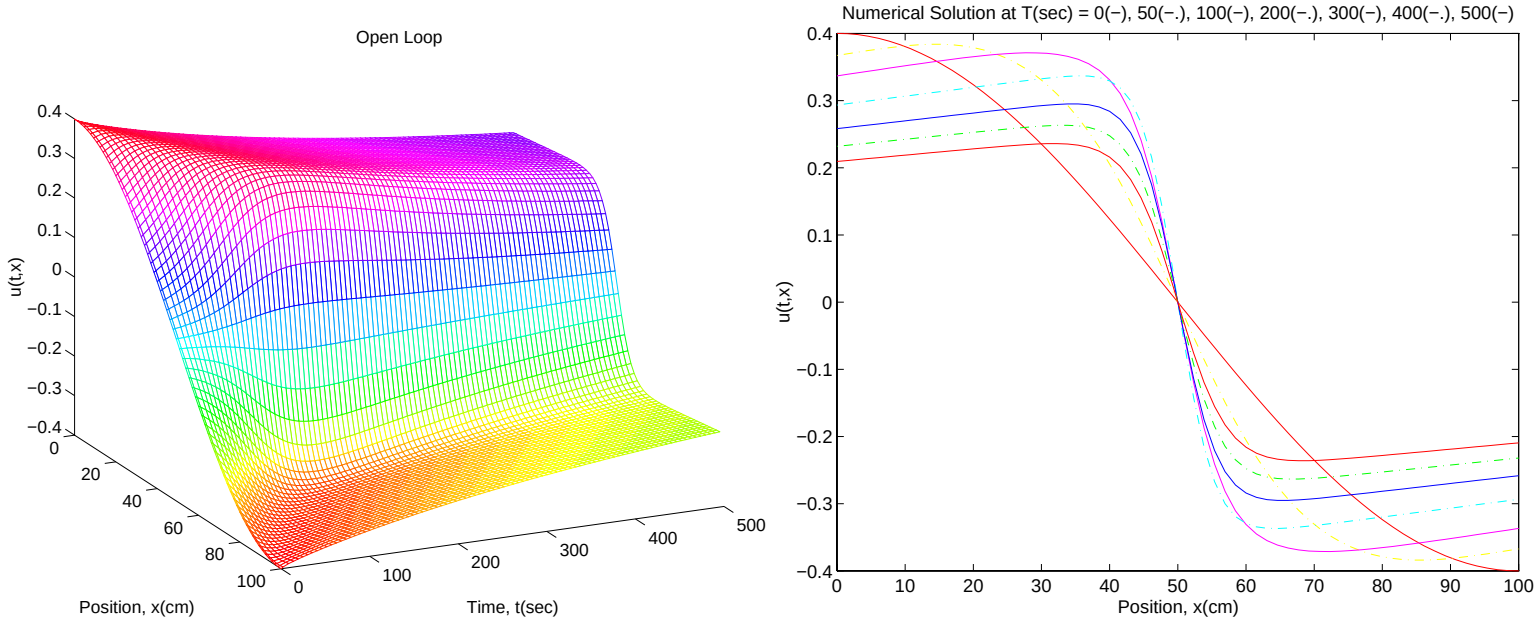


Figure 3.3.73: 1m Al rod, 50cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 64

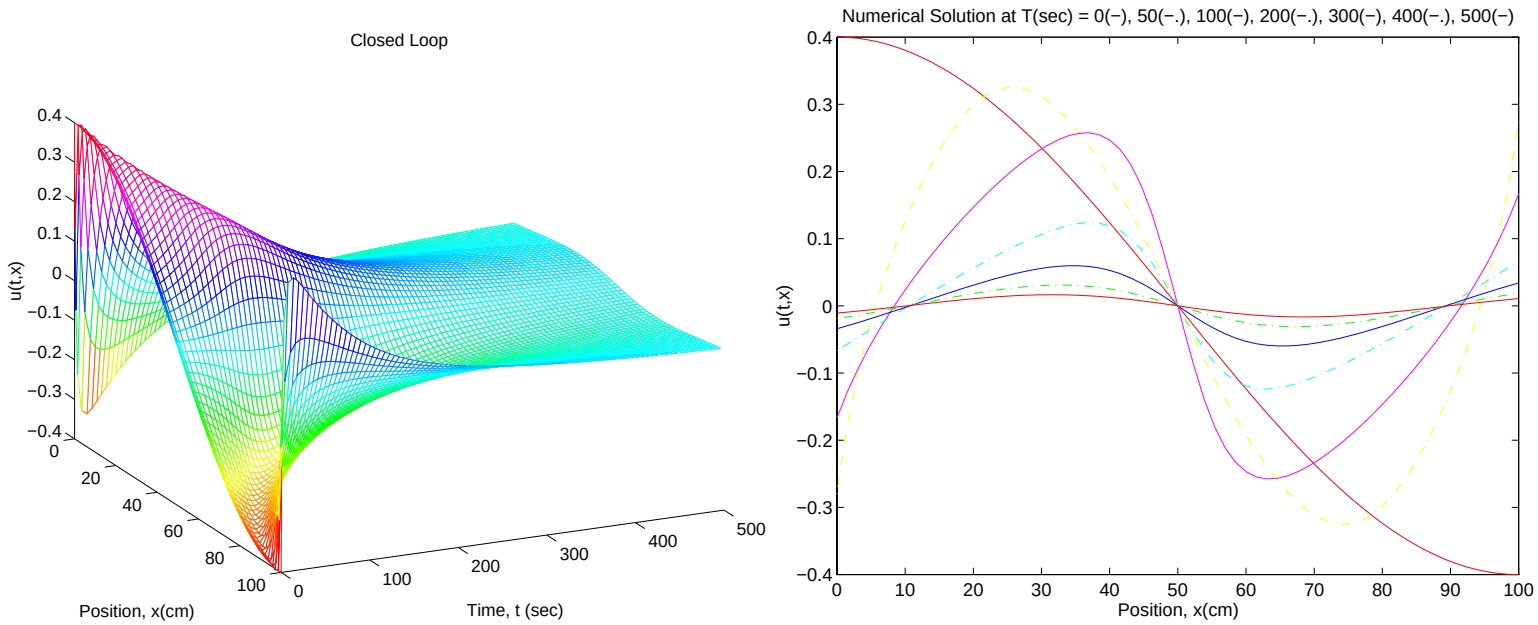


Figure 3.3.74: 1m Al rod, 50cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 64

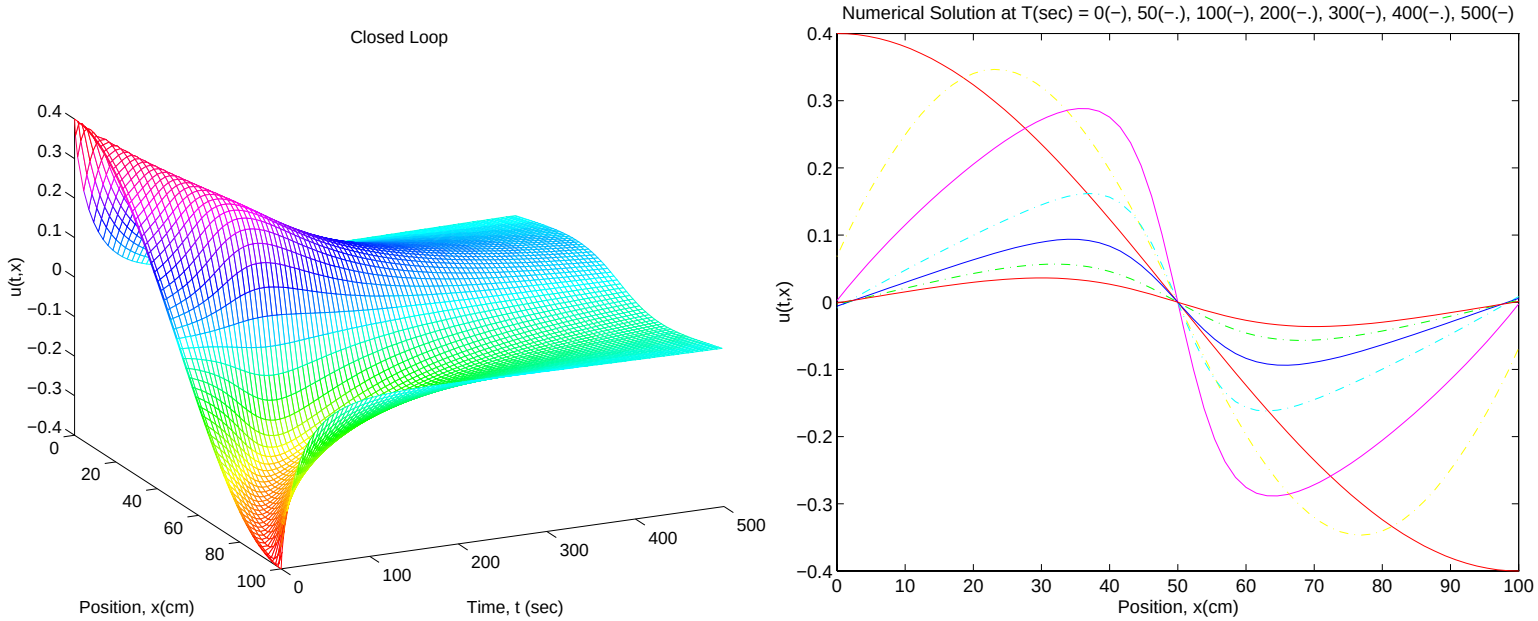


Figure 3.3.75: 1m Al rod, 50cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 64

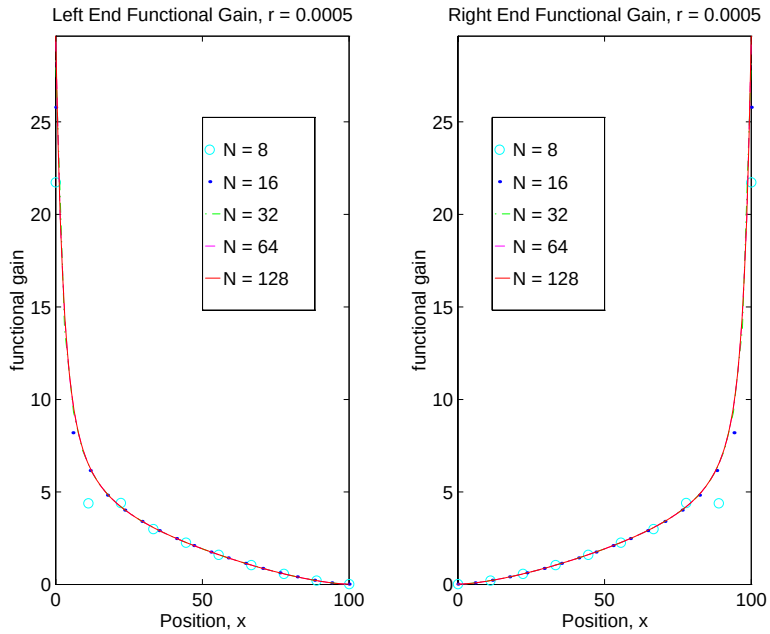


Figure 3.3.76: Functional gains, r = .0005 for a 1m Al rod with 25cm Fe films

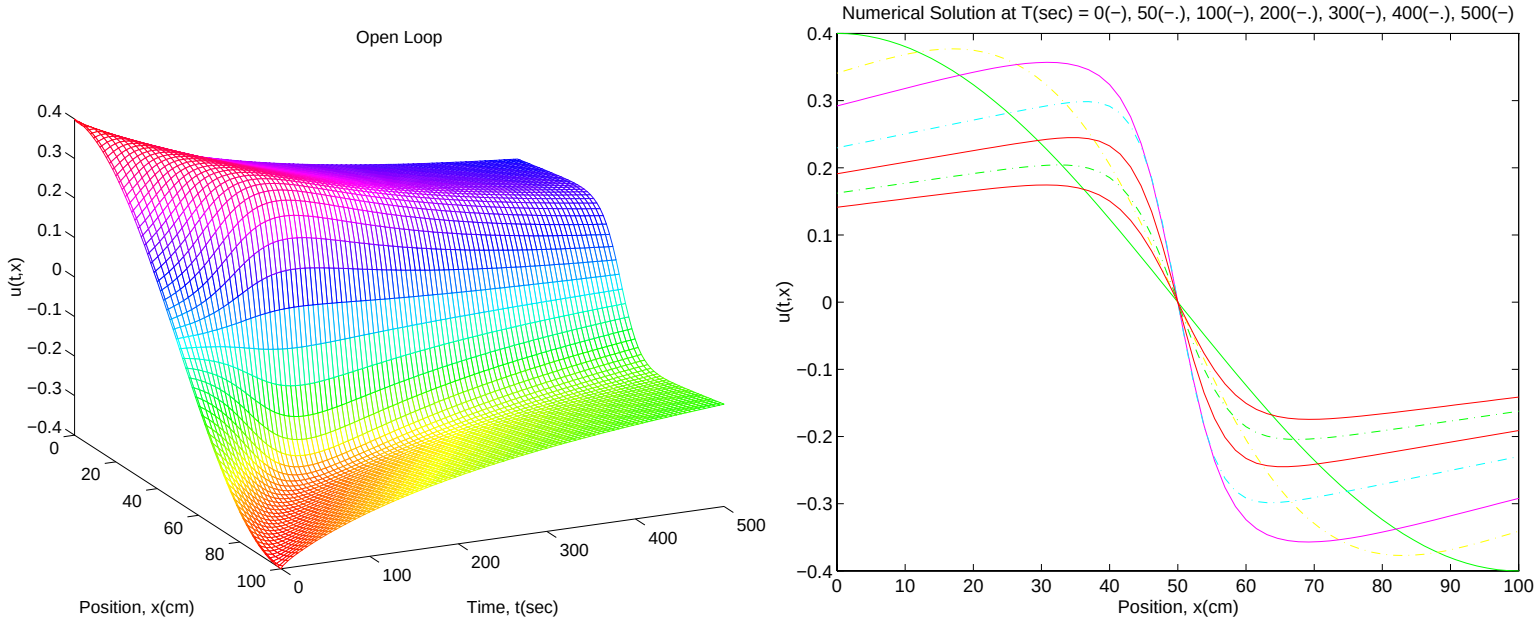


Figure 3.3.77: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 64

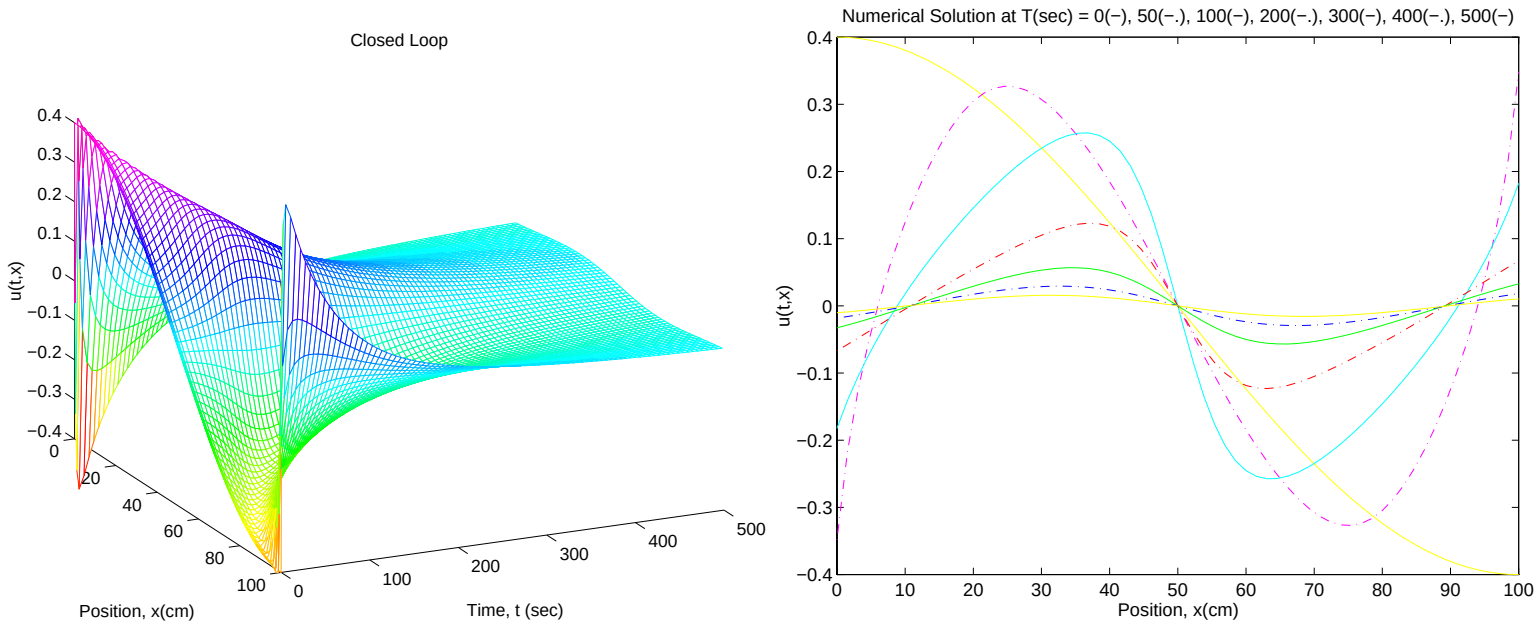


Figure 3.3.78: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 64

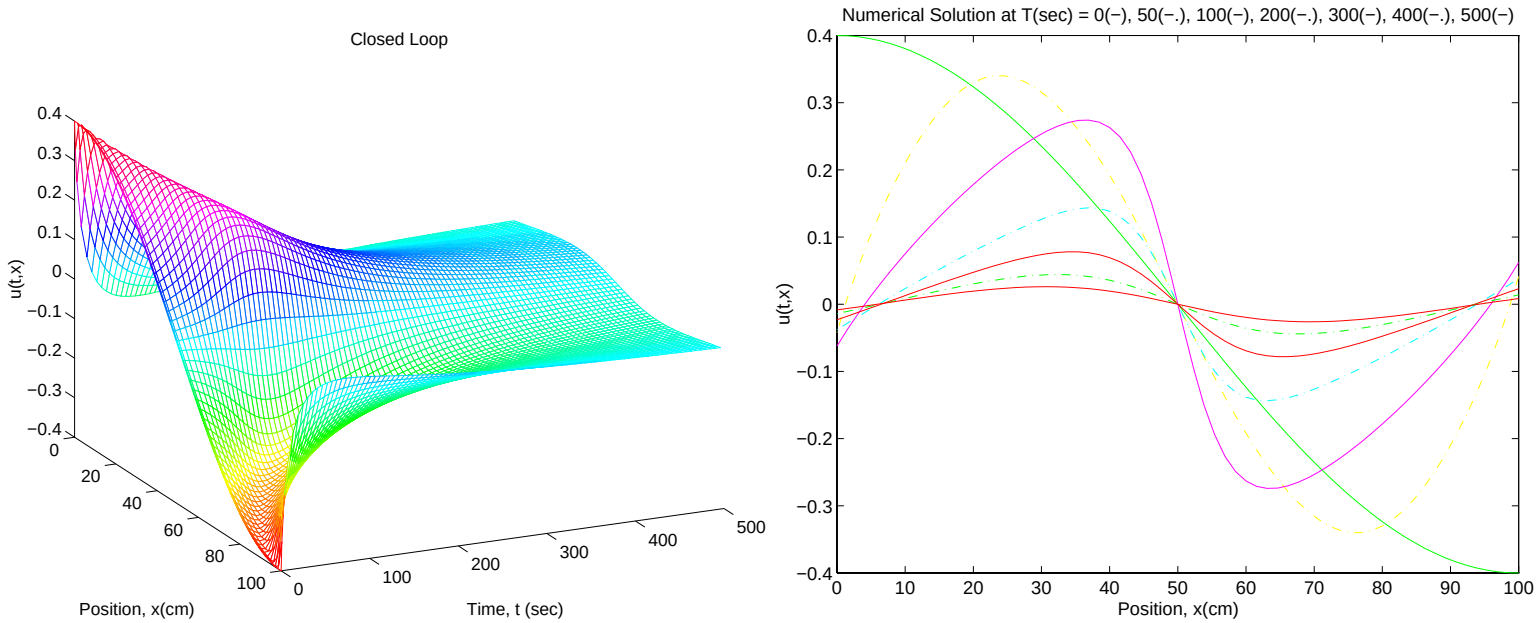


Figure 3.3.79: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 64

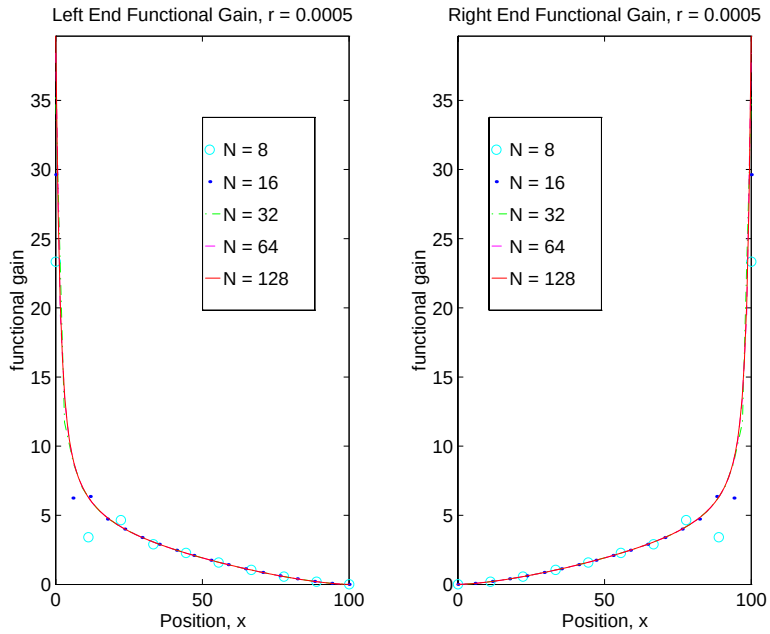


Figure 3.3.80: Functional gains, r = .0005 for a 1m Al rod with 10cm Fe films

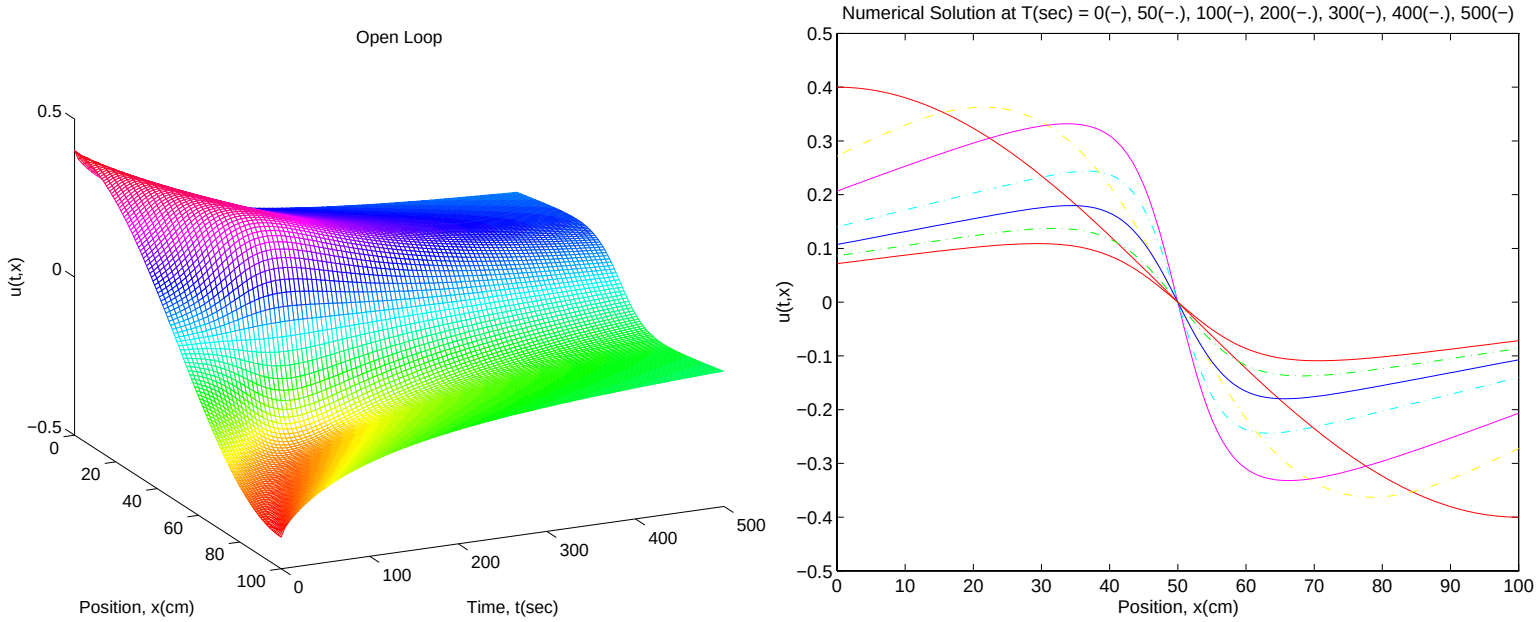


Figure 3.3.81: 1m Al rod, 10cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 128

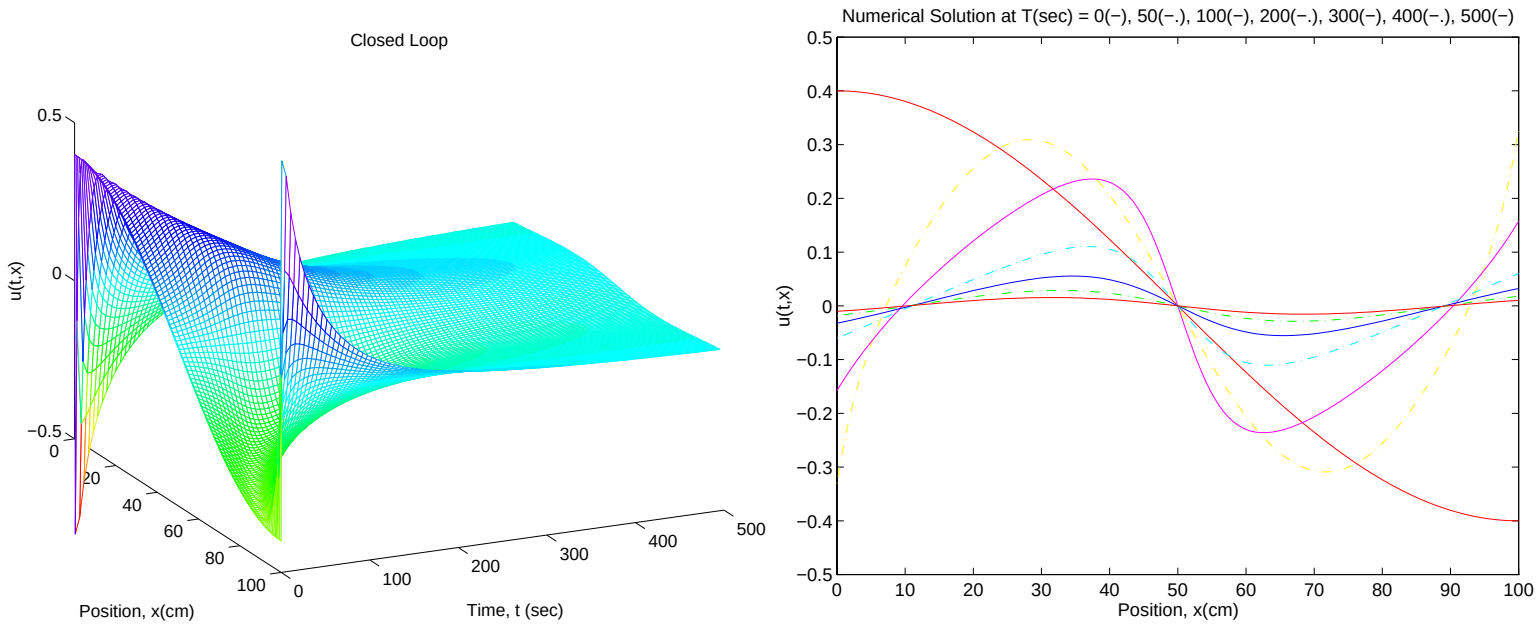


Figure 3.3.82: 1m Al rod, 10cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 128

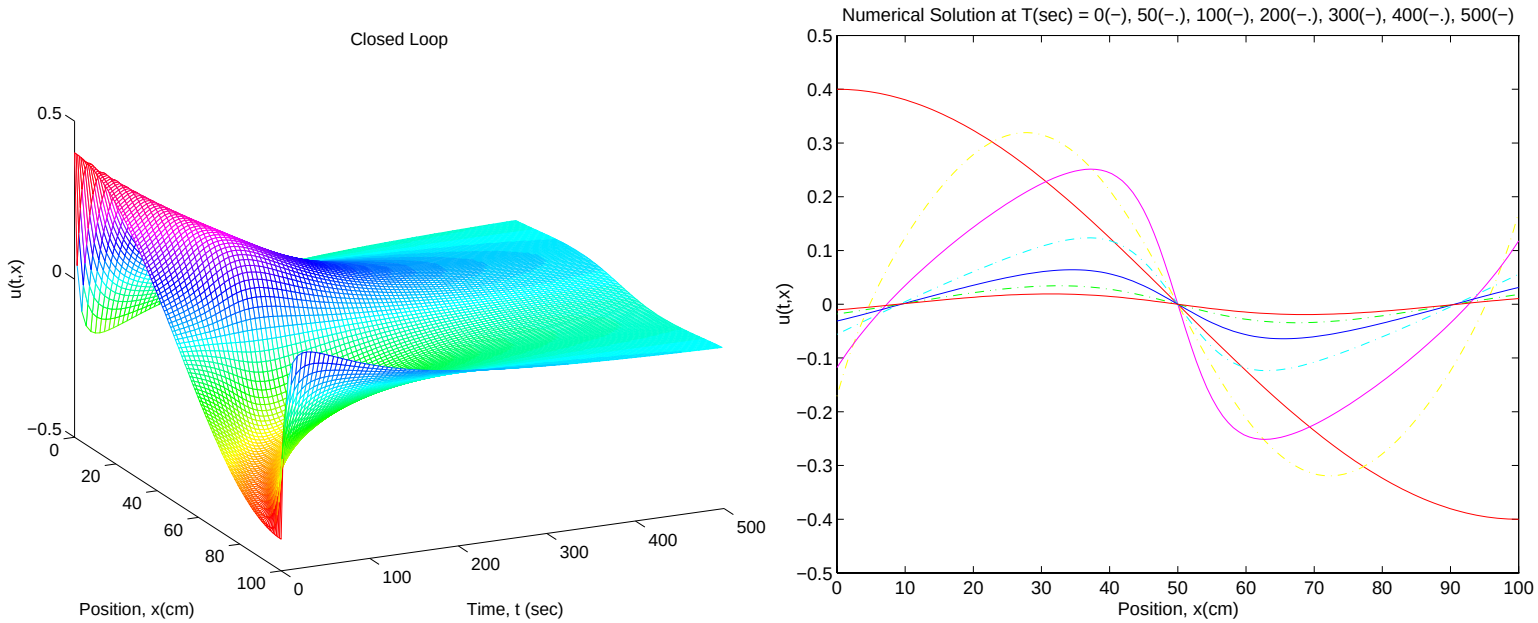


Figure 3.3.83: 1m Al rod, 10cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 128

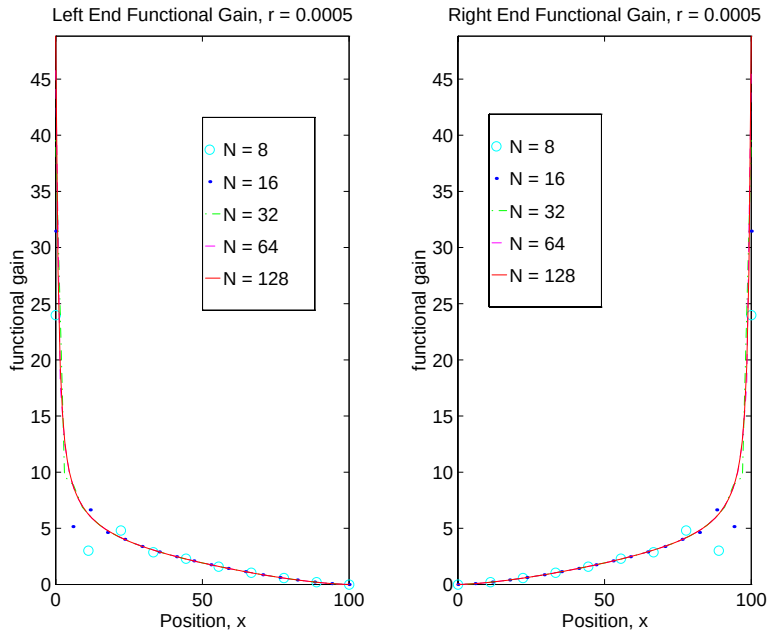


Figure 3.3.84: Functional gains, r = .0005 for a 1m Al rod with 5cm Fe films

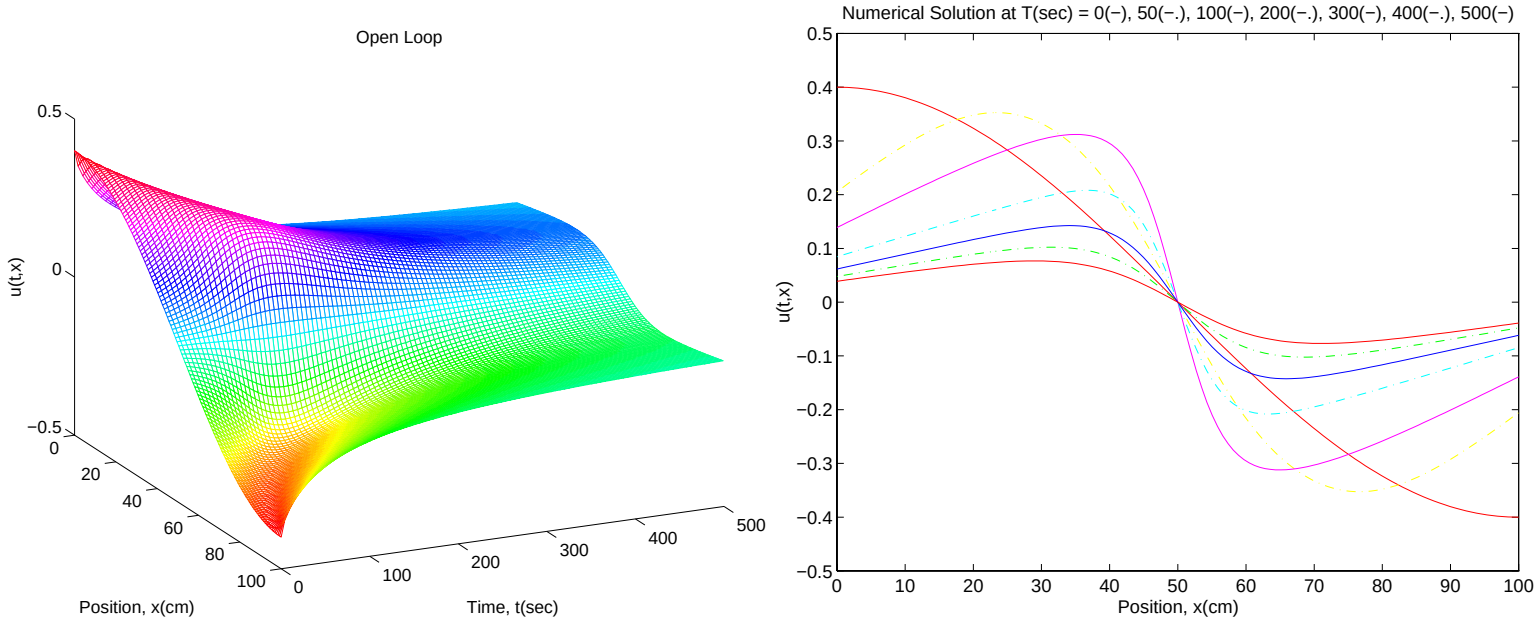


Figure 3.3.85: 1m Al rod, 5cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, N = 128

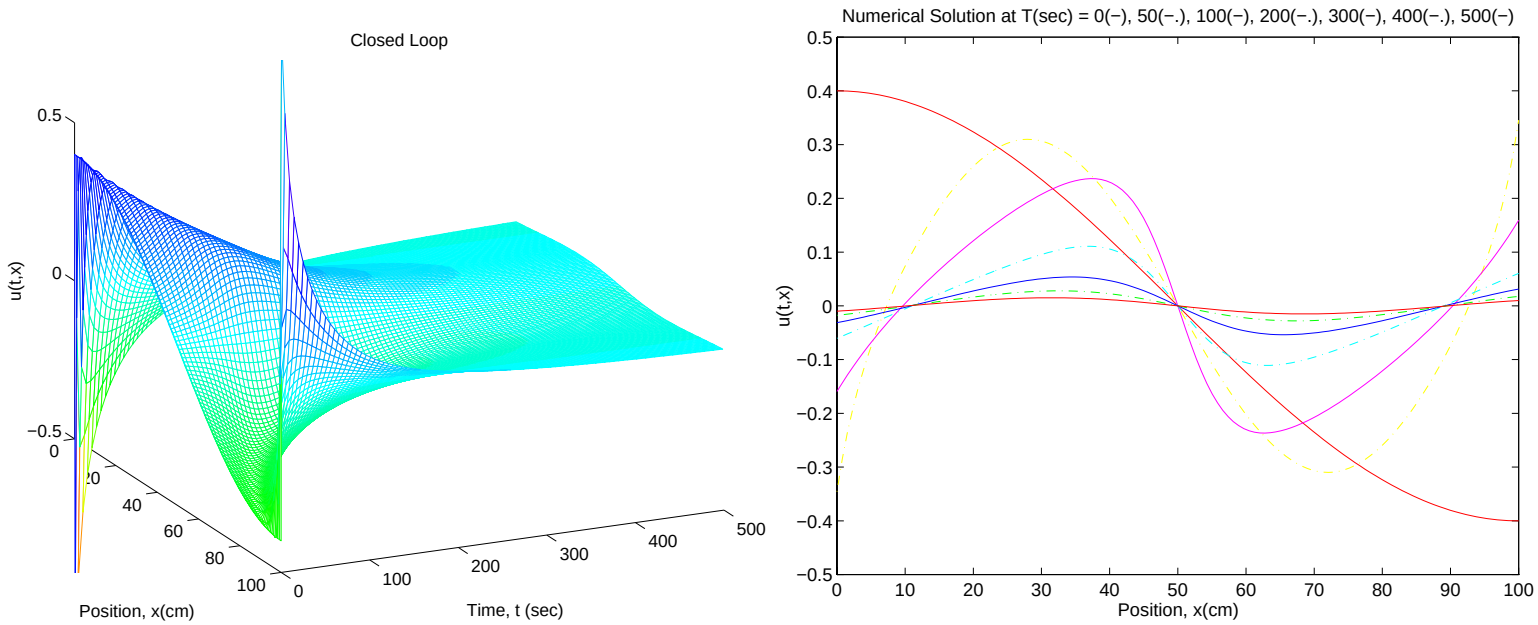


Figure 3.3.86: 1m Al rod, 5cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{.939}$, N = 128

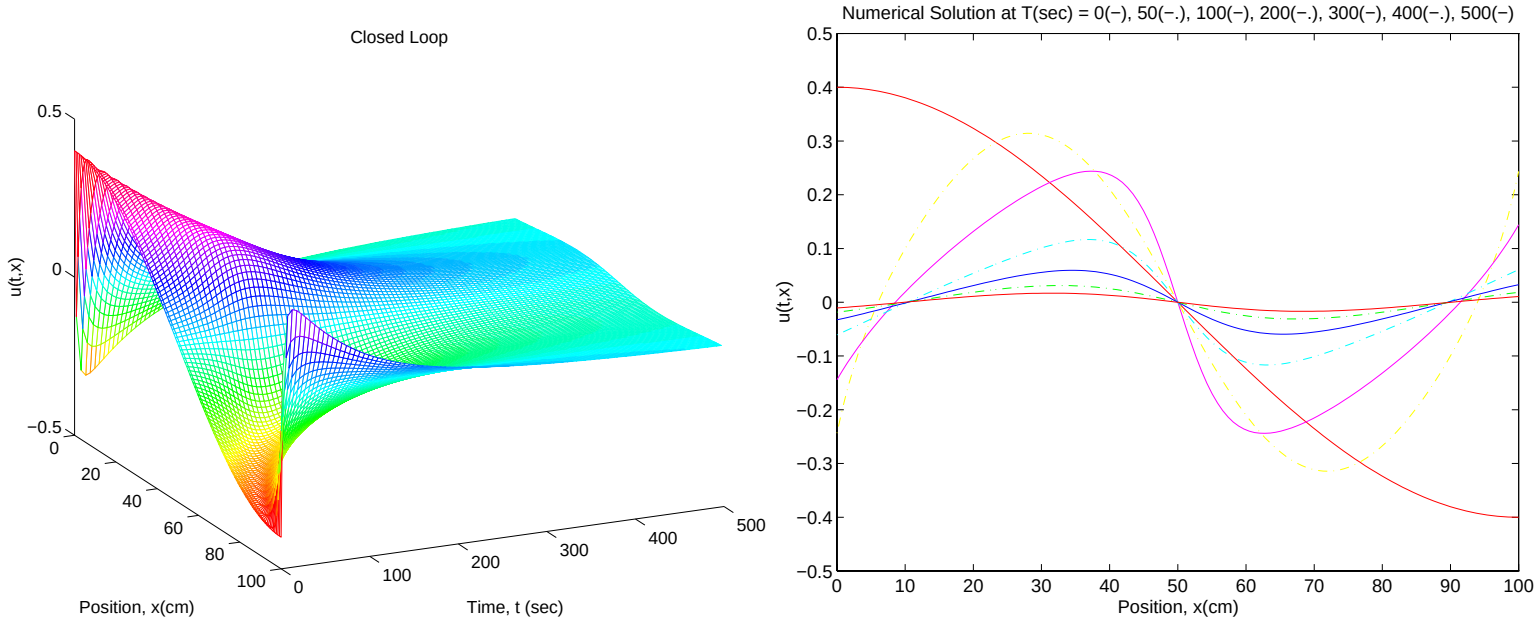


Figure 3.3.87: 1m Al rod, 5cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, $K_{1.14}$, N = 128

### 3.3.4 Reynolds Number Experiments

This section compares the numerical solutions for the same rod ($\kappa = .55$) and films ($\kappa_1 = \kappa_2 = .124$) for various Reynolds numbers. In the previous examples, $\rho$ and $c$ are the true values of density and specific heat, respectively, for the material of the rod. However, in the following examples, the factor ($\rho c$) is adjusted in order to obtain specific values of $\epsilon$. The weighting constant, $r$, is still fixed at .0005. The Reynolds number, denoted by $Re$, is related to $\epsilon$ by $Re = \frac{1}{\epsilon}$. In order to test the ability of the control function, $u_{1.14}(t) = -K_{1.14}\alpha(t)$, to control systems with $\epsilon < 1.14$, the following closed loop systems are compared:

$$\begin{aligned}\dot{\alpha}(t)^N(t) &= (A^N_{\frac{1}{Re}} - B^N_{\frac{1}{Re}} K_{\frac{1}{Re}})\alpha(t) - \mathcal{F}^N_D(\alpha(t)) \\ &= \mathcal{A}^N_{\frac{1}{Re}}\alpha(t) - \mathcal{F}^N_D(\alpha(t))\end{aligned} \tag{3.22}$$

$$\begin{aligned}\dot{\alpha}(t)^N(t) &= (A^N_{\frac{1}{Re}} - B^N_{\frac{1}{Re}} K_{1.14})\alpha(t) - \mathcal{F}^N_D(\alpha(t)) \\ &= \mathcal{A}^N_{1.14}\alpha(t) - \mathcal{F}^N_D(\alpha(t))\end{aligned} \tag{3.23}$$

First, numerical solutions are generated for the one meter aluminum rod with ten meter iron films ($L_1 = L_2 = 1000cm$), which approximates Neumann boundary conditions. Figures 3.3.88, 3.3.91, 3.3.94, 3.3.97, 3.3.100, and 3.3.103 are the numerical solutions for the open loop systems with Reynolds numbers of 10, 20, 40, 80, 160, and 320, respectively. After $Re = 40$, there is a significant increase in the rate at which the open loop solution decays toward zero. At 3000 seconds, the amplitude of the open loop numerical solution

for $Re = 40$ is .2309. The amplitude of the open loop numerical solution at 3000 seconds drops to .0447 at $Re = 80$. This rate increases slightly more as the Reynolds numbers increase up to 320.

The numerical solutions to the closed loop systems with the optimal gain, $K_{\frac{1}{Re}}$, applied are shown in Figures 3.3.89, 3.3.92, 3.3.95, 3.3.98, 3.3.101, and 3.3.104. As the Reynolds numbers increase, the performance of the optimal control is degraded. The numerical solutions approach a step function rather than the zero steady state function.

Figures 3.3.90, 3.3.93, 3.3.96, 3.3.99, 3.3.102, and 3.3.105 are the corresponding closed loop systems with the control, $u_{1.14}(t) = -K_{1.14}\alpha(t)$. The performance of the control function, $u_{1.14}(t)$, in (3.23) approaches the performance of the optimal control, $u_{\frac{1}{Re}}(t)$, in (3.22) as the Reynolds numbers increase.

The next experiment follows the same procedure as before, except, $L_1 = L_2 = 25cm$ in order to approximate Dirichlet boundary conditions. Figures 3.3.106, 3.3.109, 3.3.112, 3.3.115, 3.3.118, and 3.3.121 are the numerical solutions for the open loop systems with Reynolds numbers of 10, 20, 40, 80, 160, and 320, respectively. As the Reynolds numbers increase, the numerical solutions steepen with time and converge toward a step function. Small perturbations in the numerical solutions, near the center of the rod, are present.

Figures 3.3.107, 3.3.110, 3.3.113, 3.3.116, 3.3.119, and 3.3.122 show the closed loop systems with optimal control. The solutions are approaching the steady state solution as time increases. However, the numerical solutions steepen with an increase in Reynolds

number.

Similarly, the closed loop solutions for the systems with control $u_{1.14}(t) = -K_{1.14}\alpha(t)$ steepen with an increasing Reynolds numbers. Figures 3.3.108, 3.3.111, 3.3.114, 3.3.117, 3.3.120, and 3.3.123 show that the performance of this control is very similar to the performance of the optimal control.

Comparing the performance of each controller at the boundaries, we see the numerical solutions for (3.23) move to zero and remain there after the initial time, while the numerical solutions with the optimal control (3.22) move through zero then relax back toward zero at a later time. This characteristic of the control can be seen by comparing Figures 3.3.107, 3.3.110, and 3.3.113 to Figures 3.3.108, 3.3.111, and 3.3.114, respectively.

In all, it appears that as the Reynolds number for a system increases there is less of a difference in the performance of the control $u_{1.14}(t)$ and performance of the optimal control, $u^*_{\frac{1}{Re}}(t)$. Also, the numerical solutions steepen toward a step function with an increasing Reynolds number.

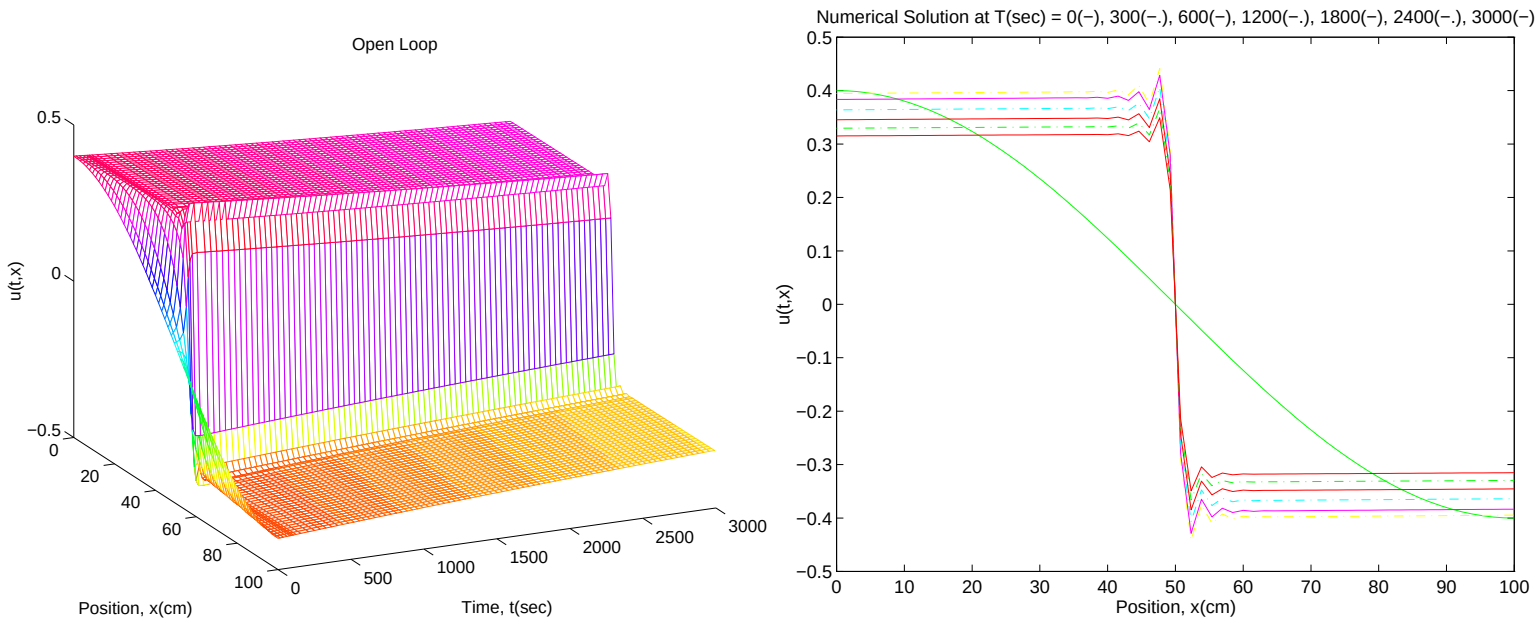


Figure 3.3.88: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, N = 64

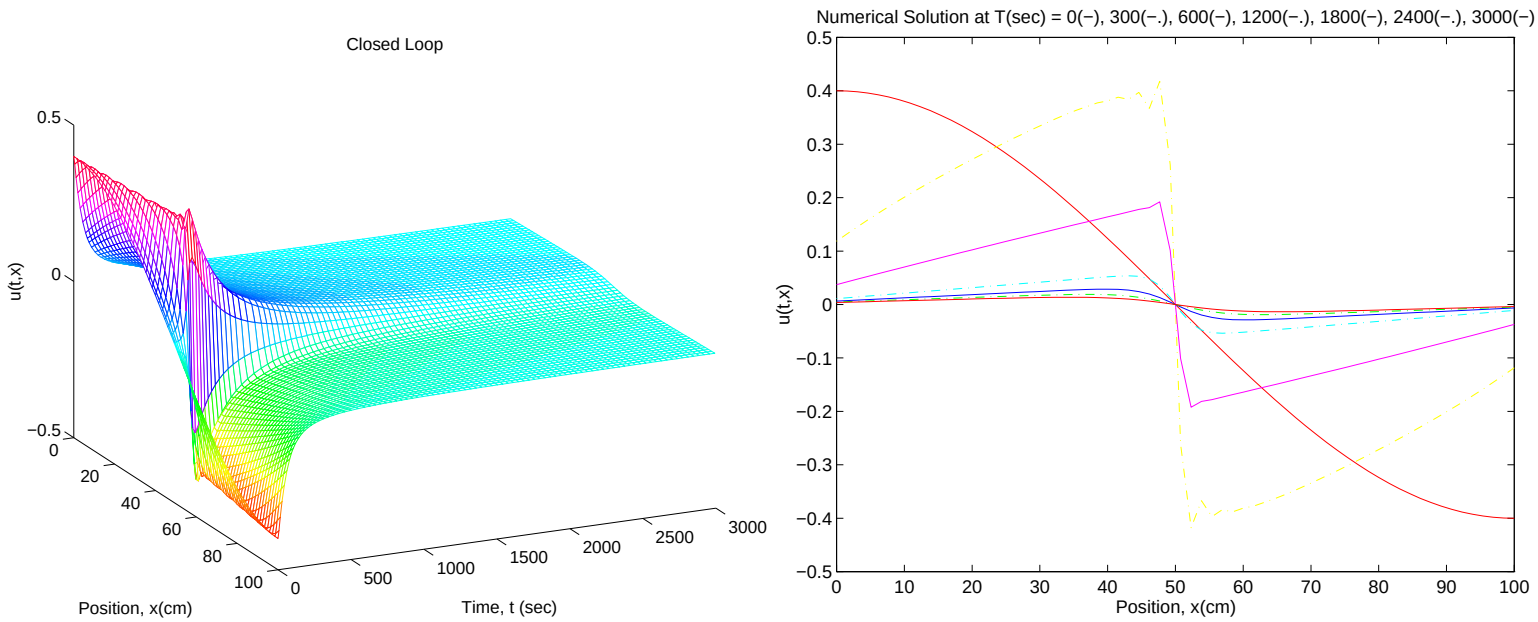


Figure 3.3.89: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{\frac{1}{10}}$, N = 64

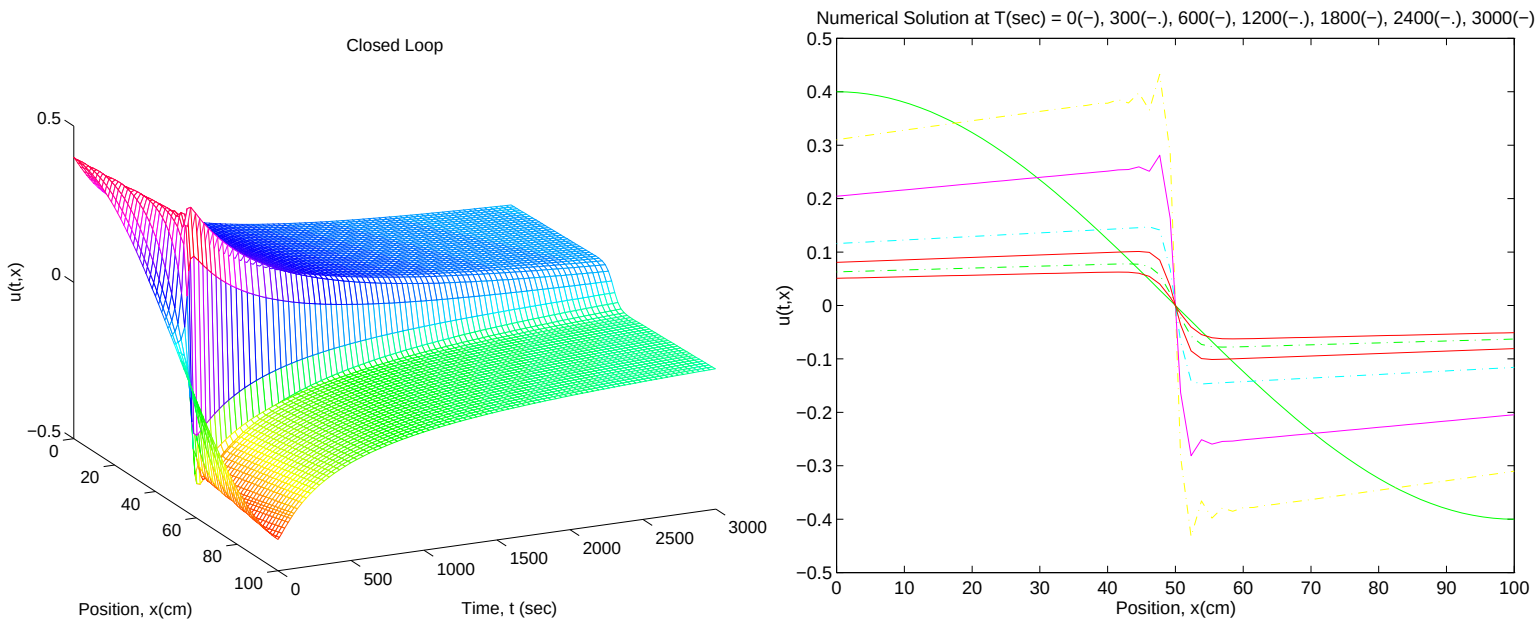


Figure 3.3.90: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{1.14}$, N = 64

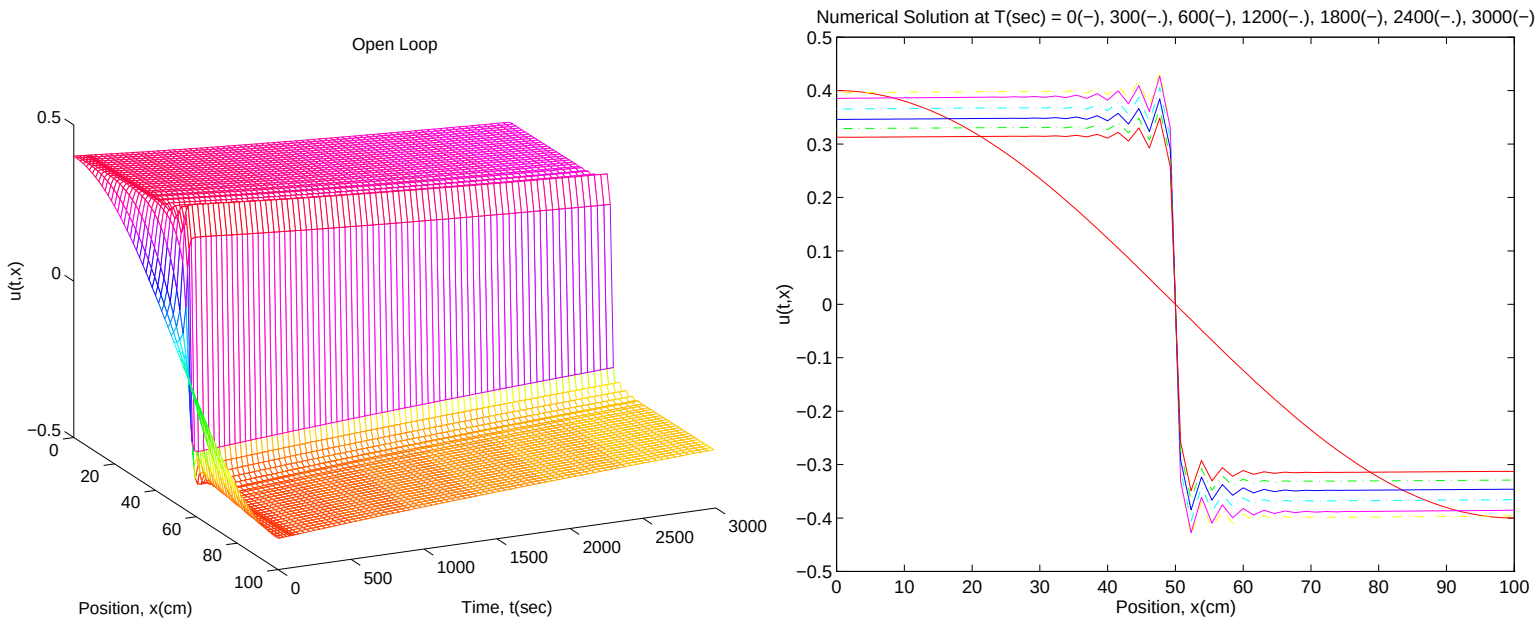


Figure 3.3.91: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, N = 64

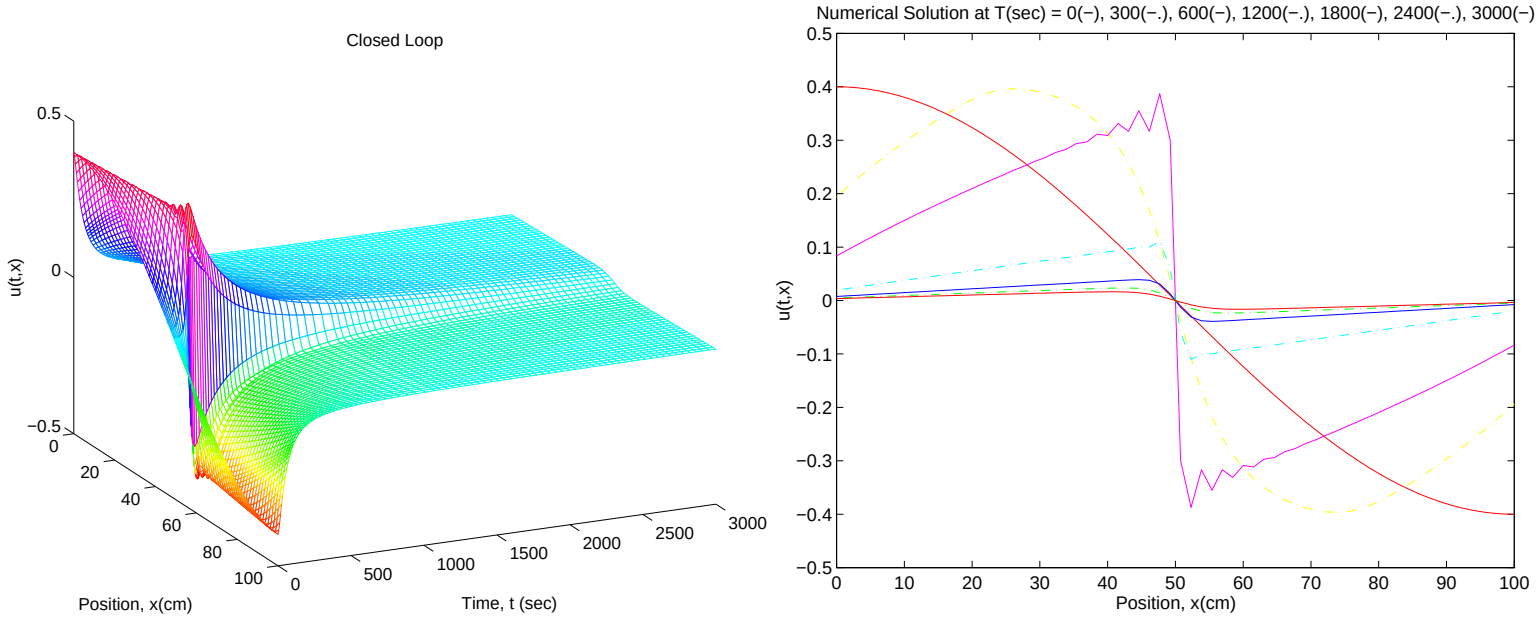


Figure 3.3.92: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{\frac{1}{20}}$, N = 64

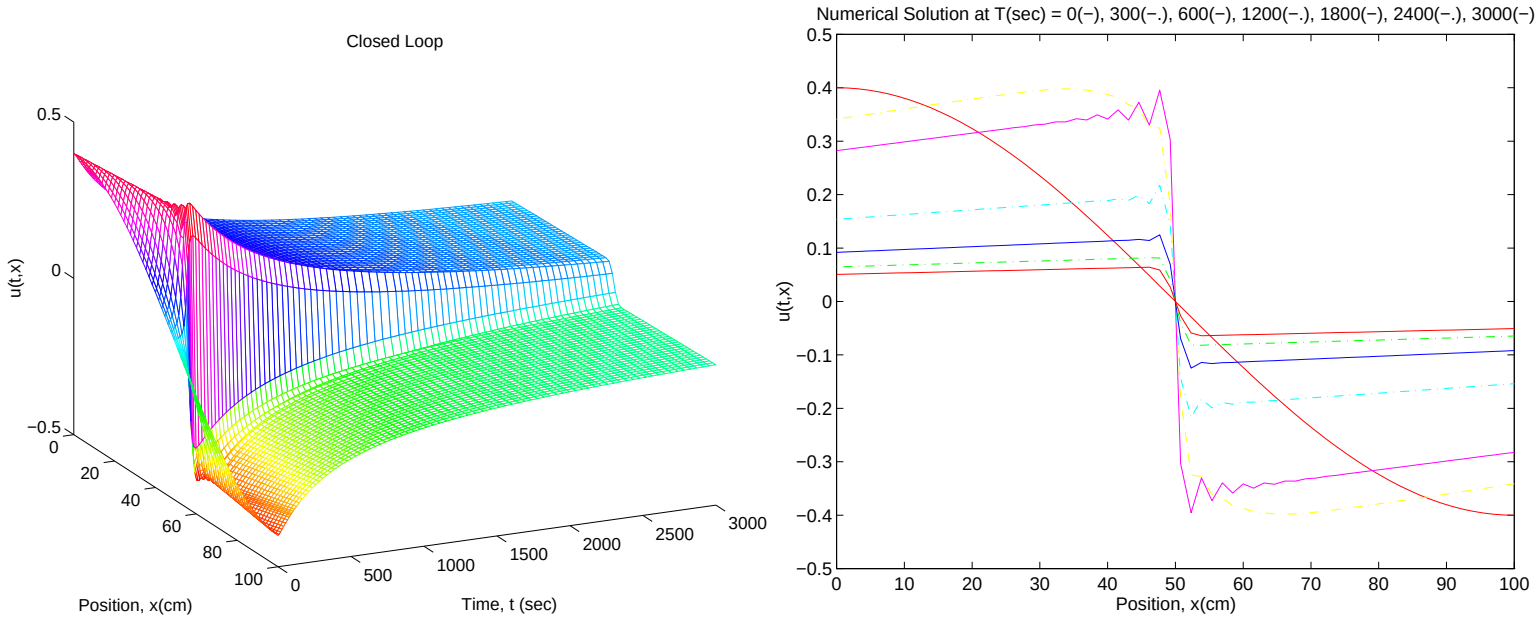


Figure 3.3.93: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{1.14}$, N = 64

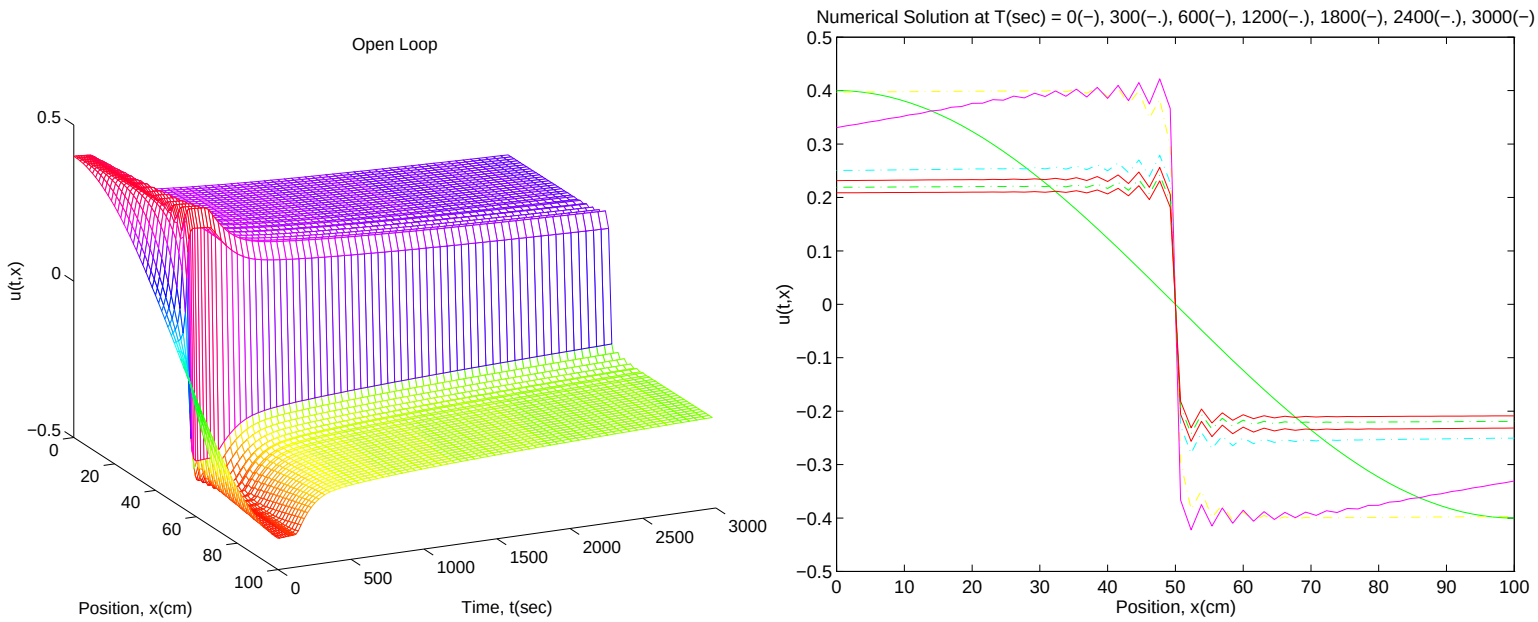


Figure 3.3.94: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, N = 64

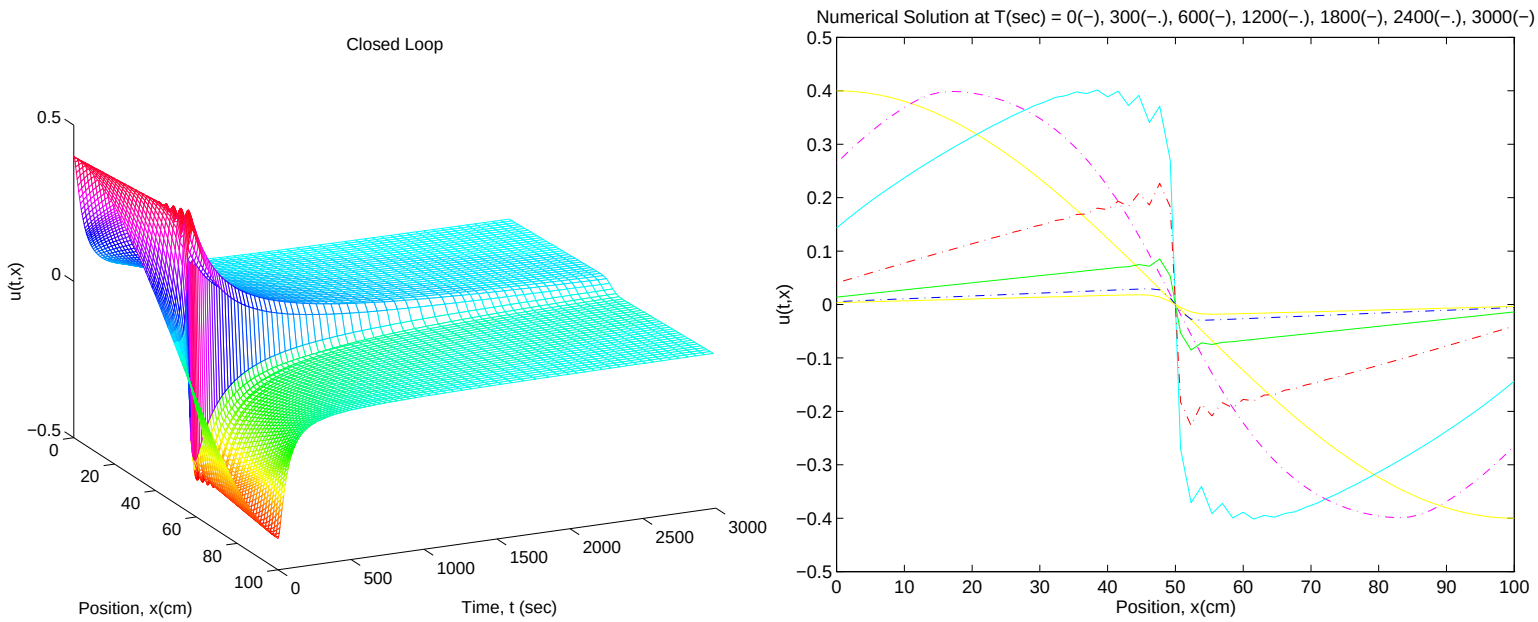


Figure 3.3.95: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{\frac{1}{40}}$, N = 64

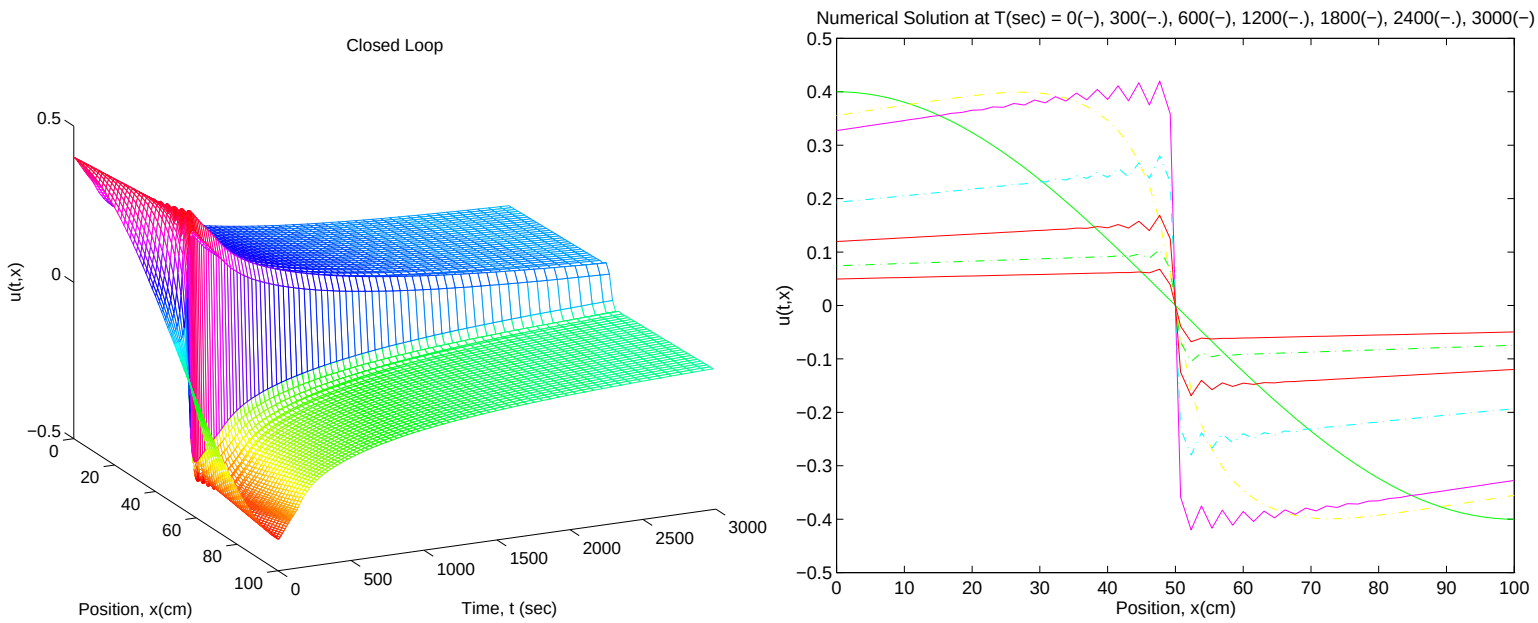


Figure 3.3.96: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{1.14}$, N = 64

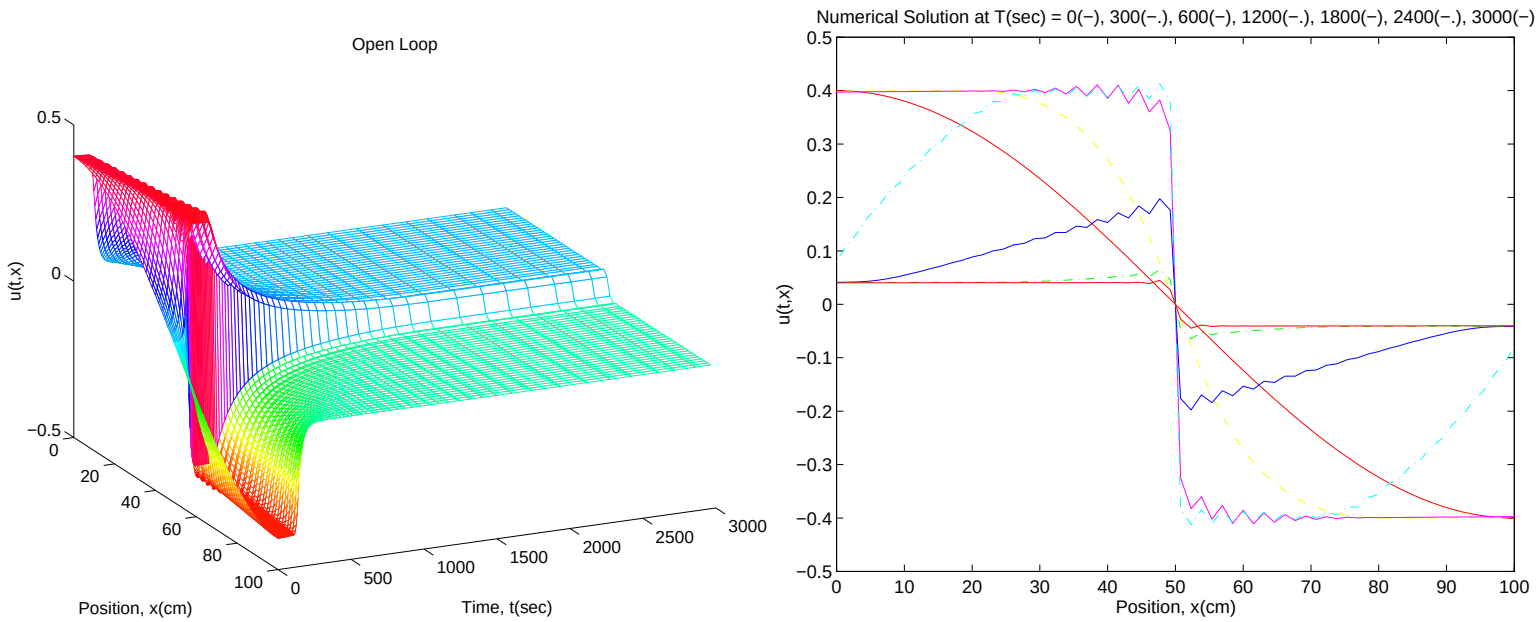


Figure 3.3.97: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, N = 64

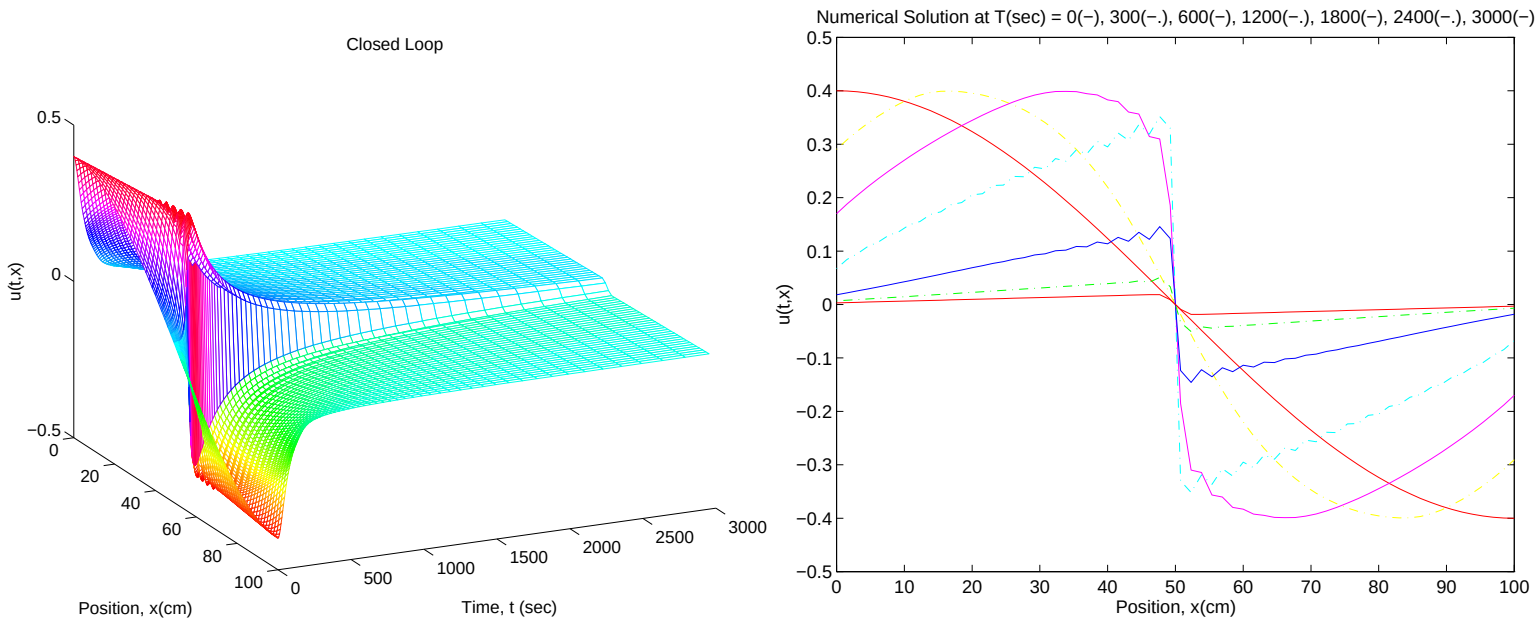


Figure 3.3.98: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{\frac{1}{80}}$, N = 64

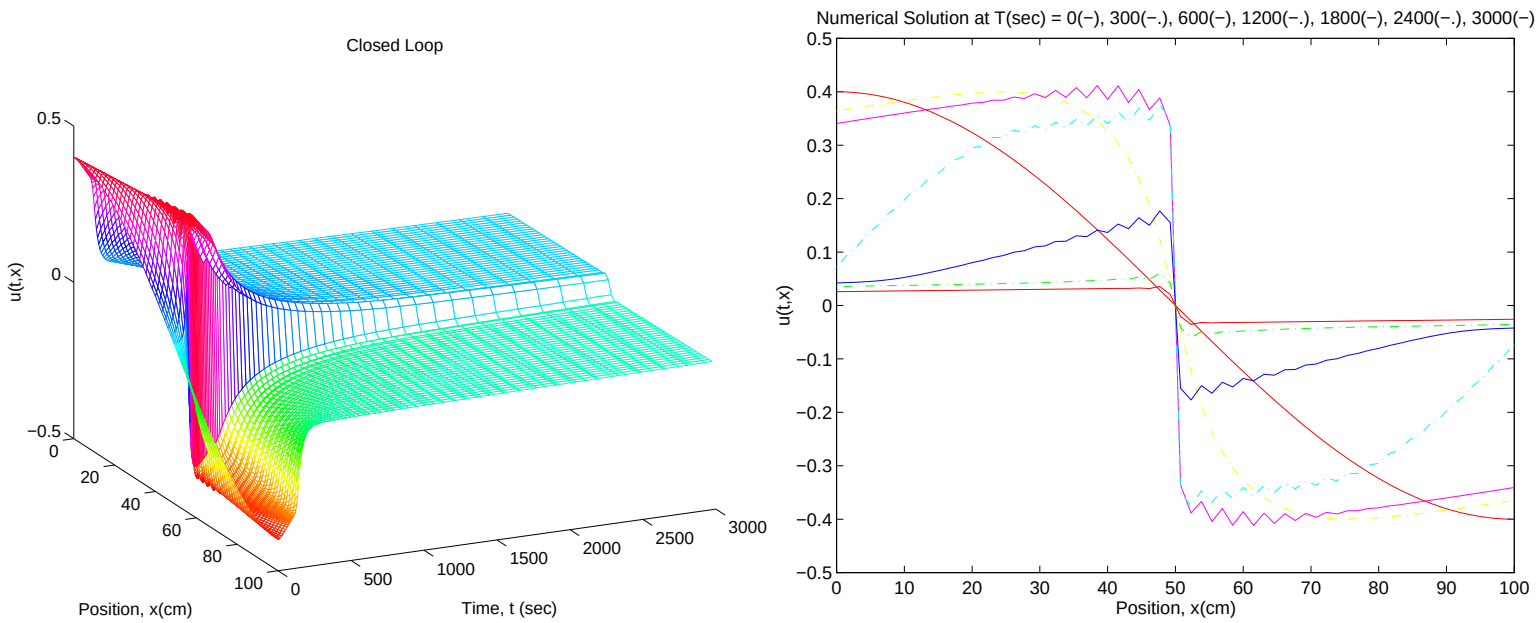


Figure 3.3.99: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{1.14}$, N = 64

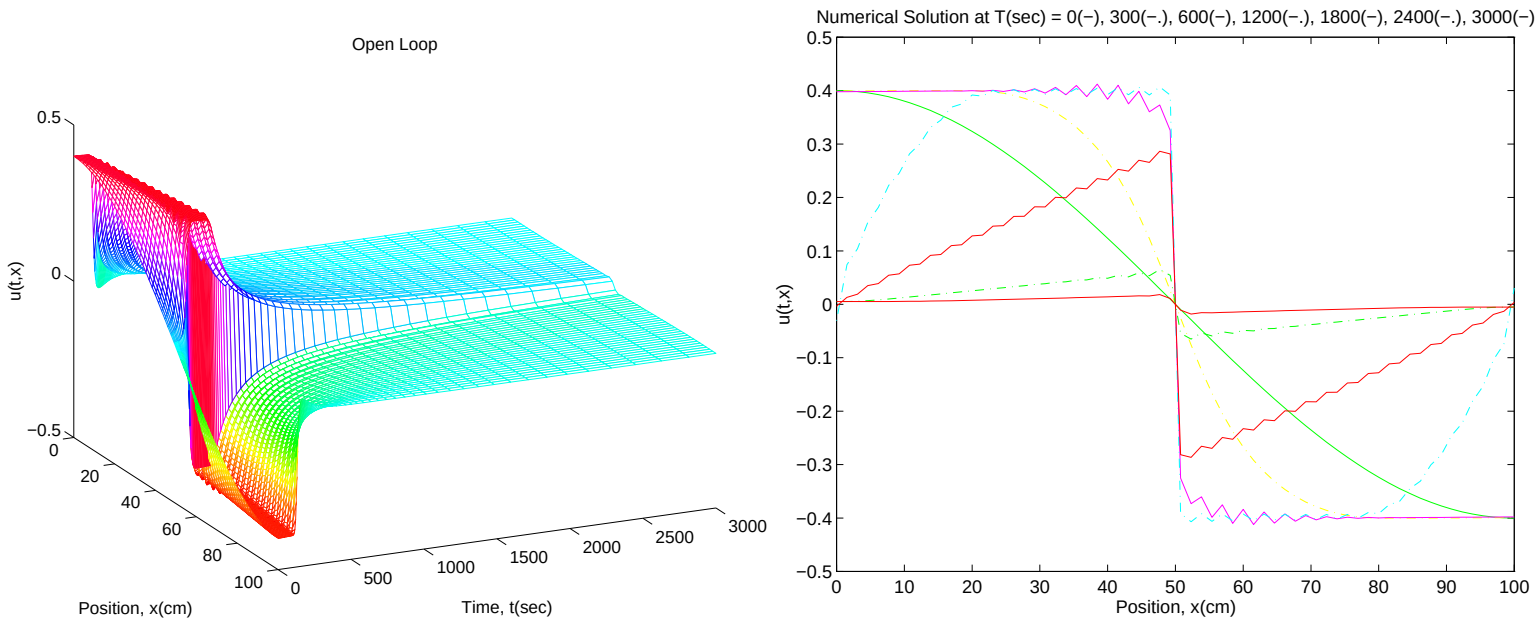


Figure 3.3.100: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, N = 64

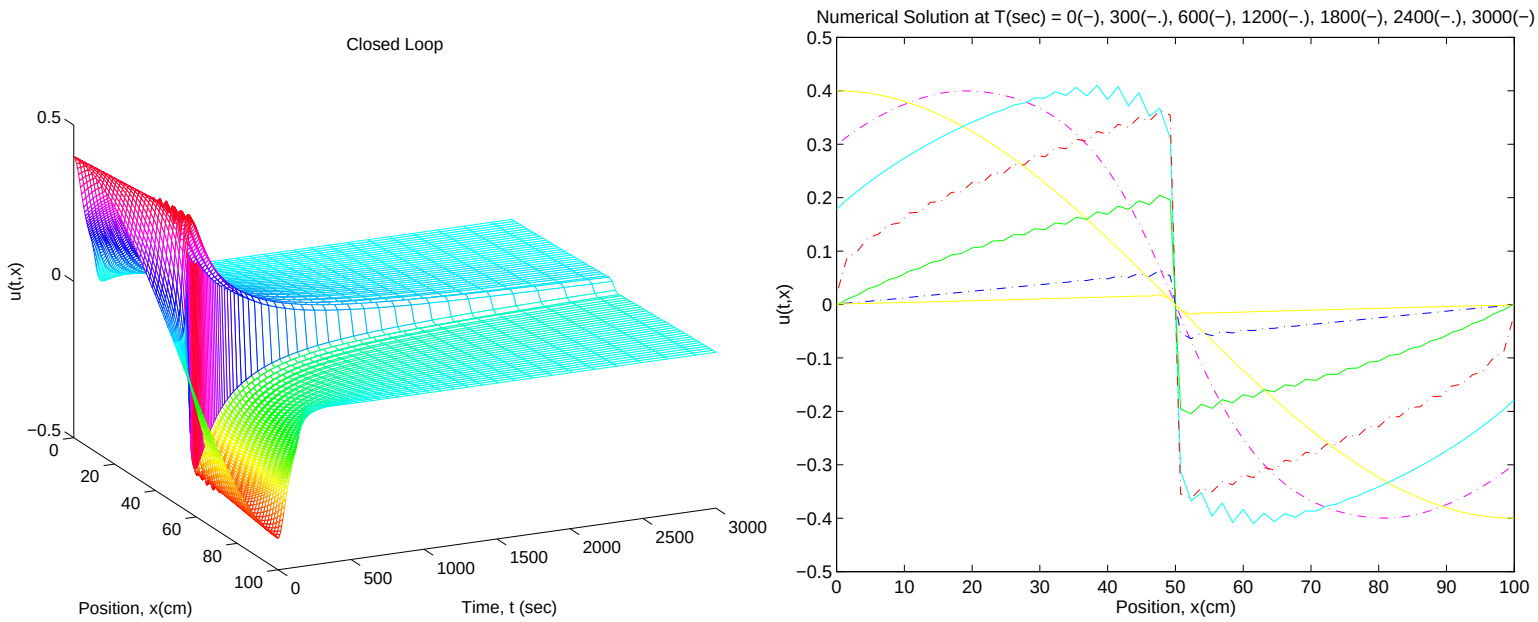


Figure 3.3.101: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{\frac{1}{160}}$, N = 64

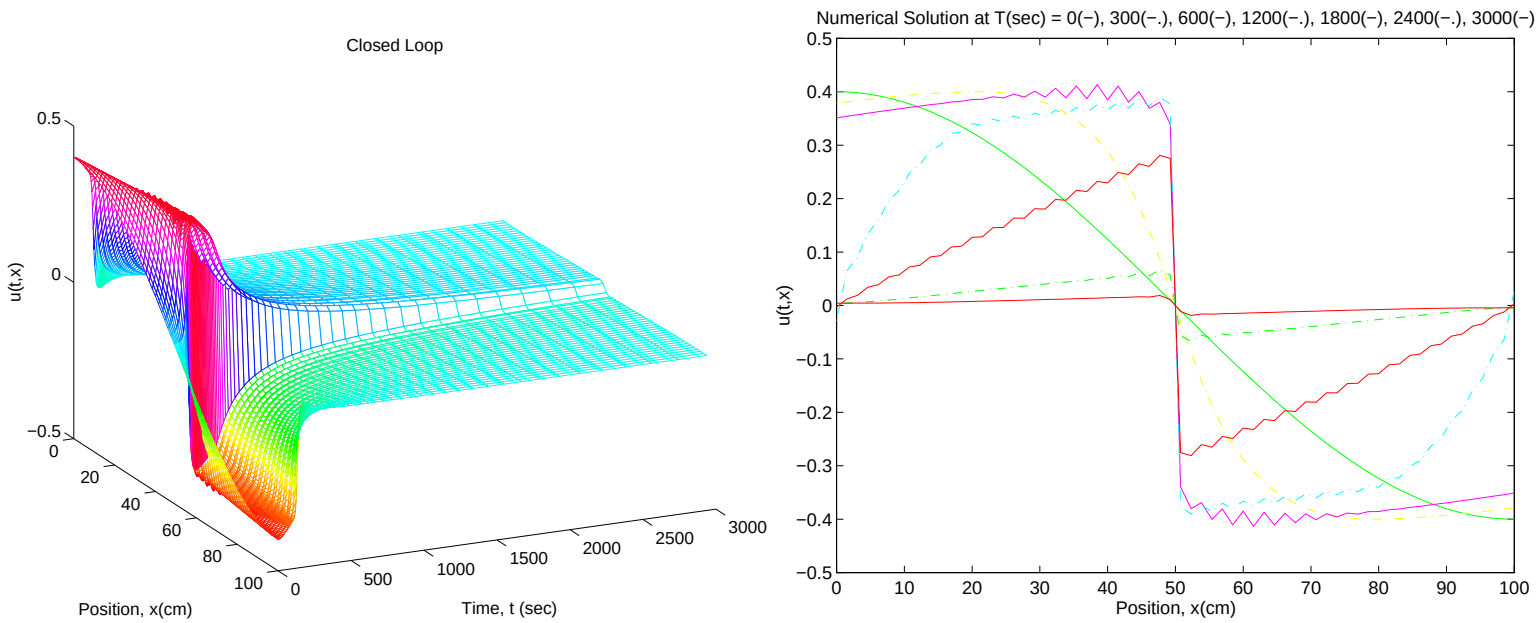


Figure 3.3.102: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{1.14}$, N = 64

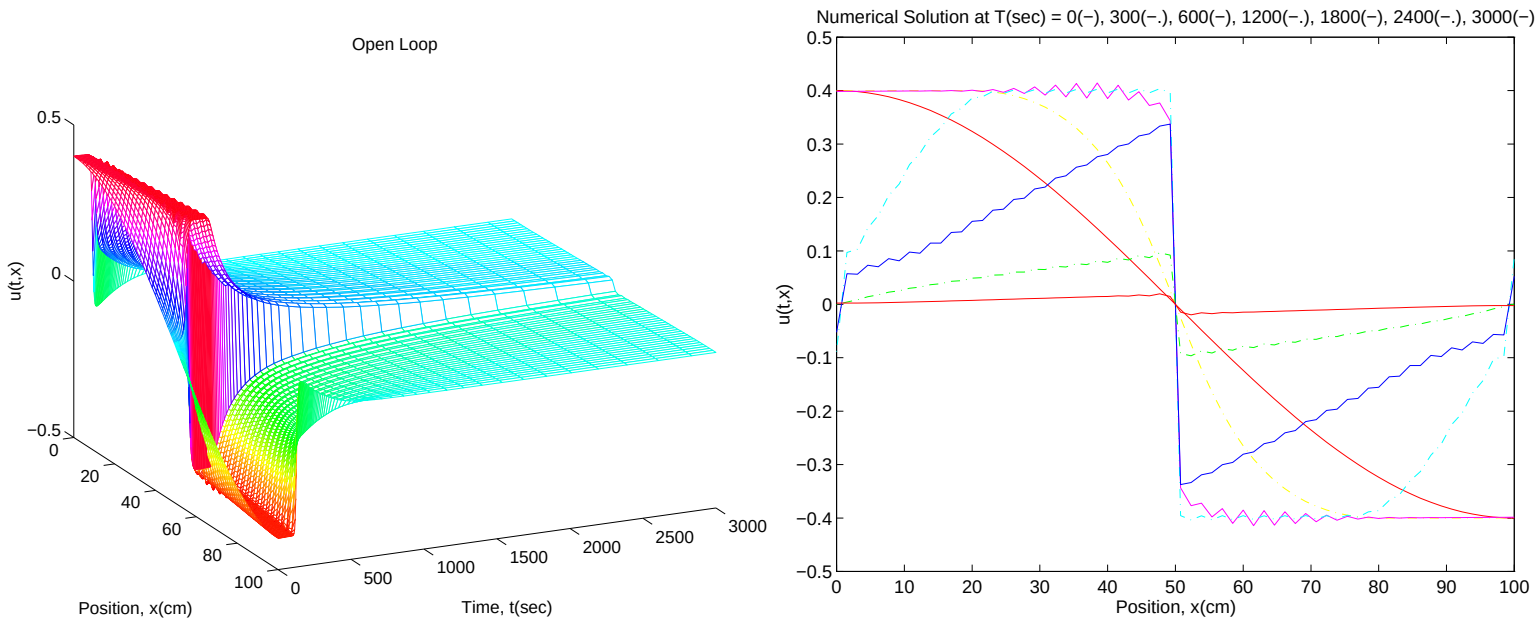


Figure 3.3.103: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, N = 64

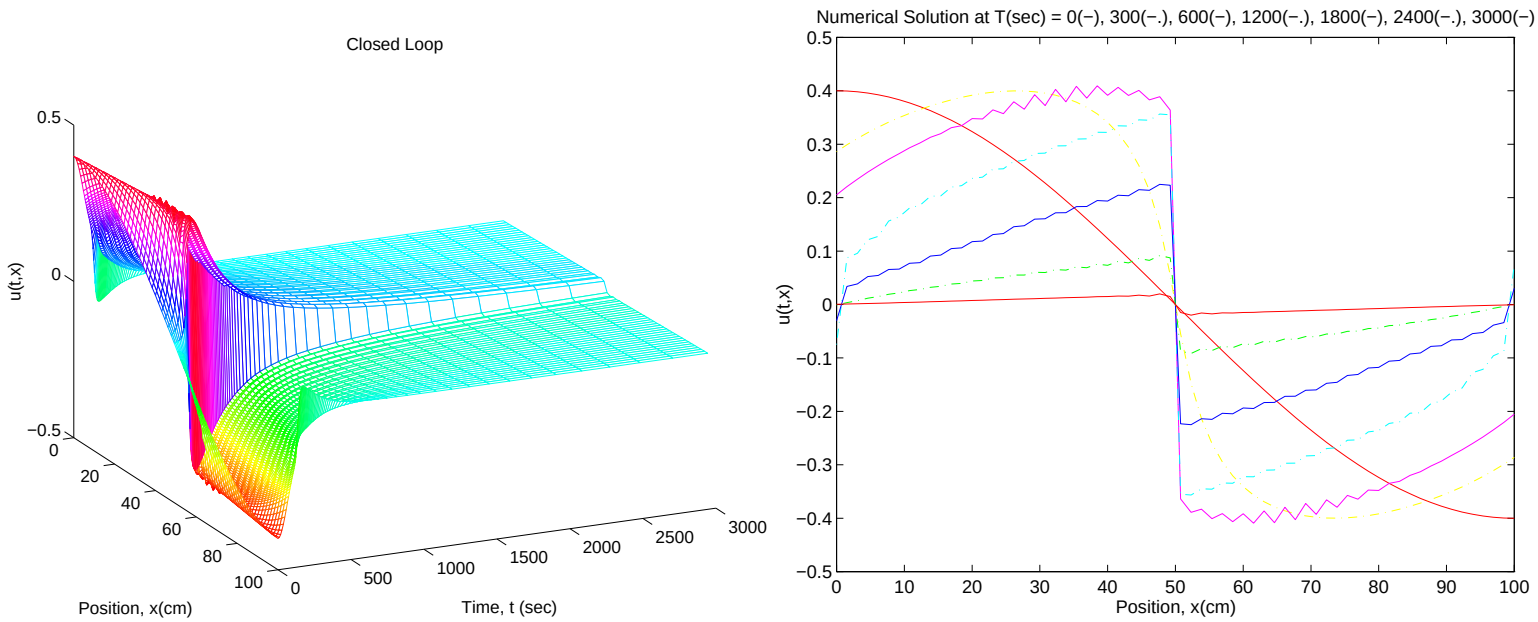


Figure 3.3.104: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{\frac{1}{320}}$, N = 64

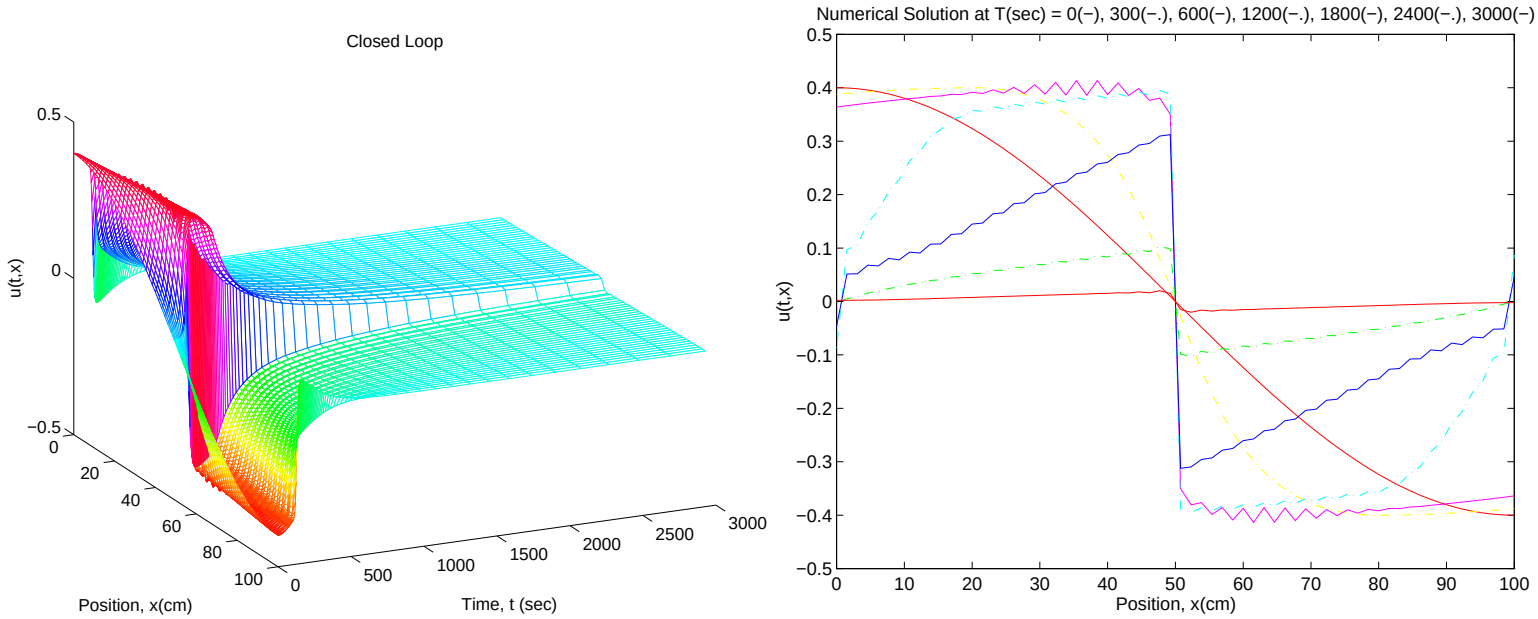


Figure 3.3.105: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{1.14}$, N = 64

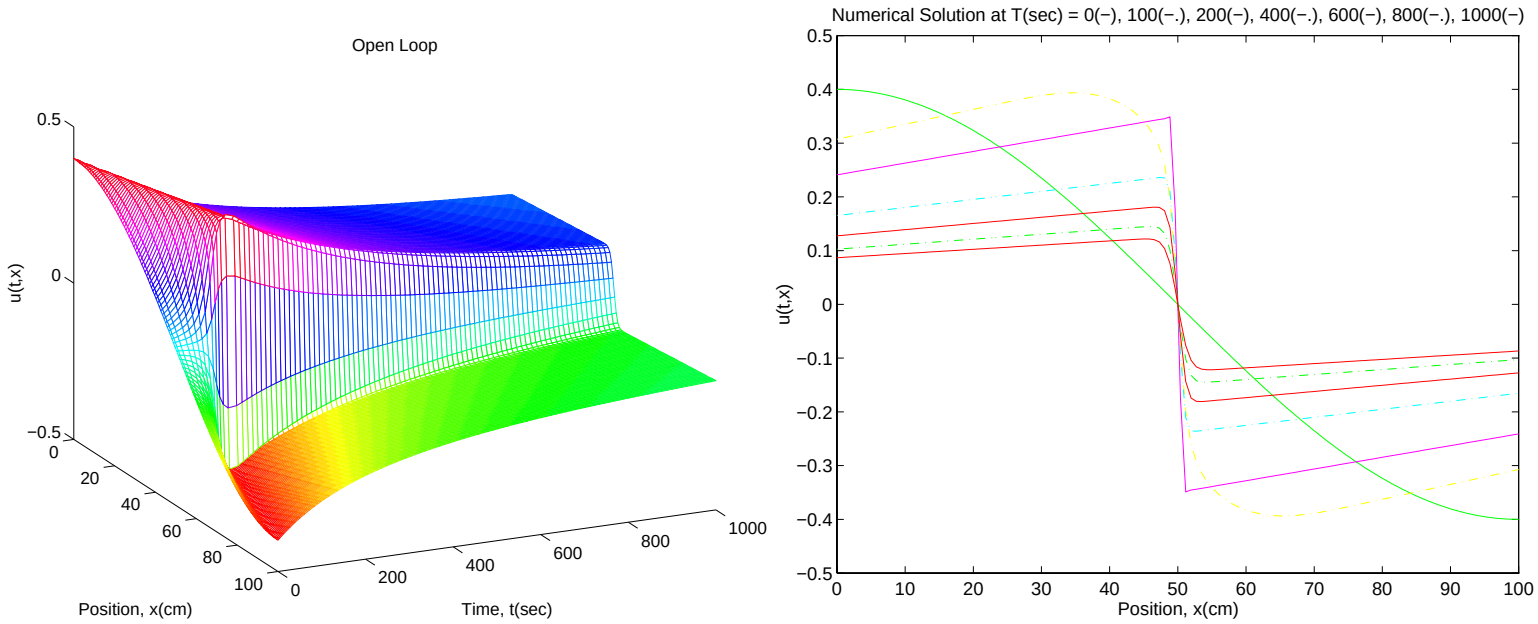


Figure 3.3.106: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, N = 128

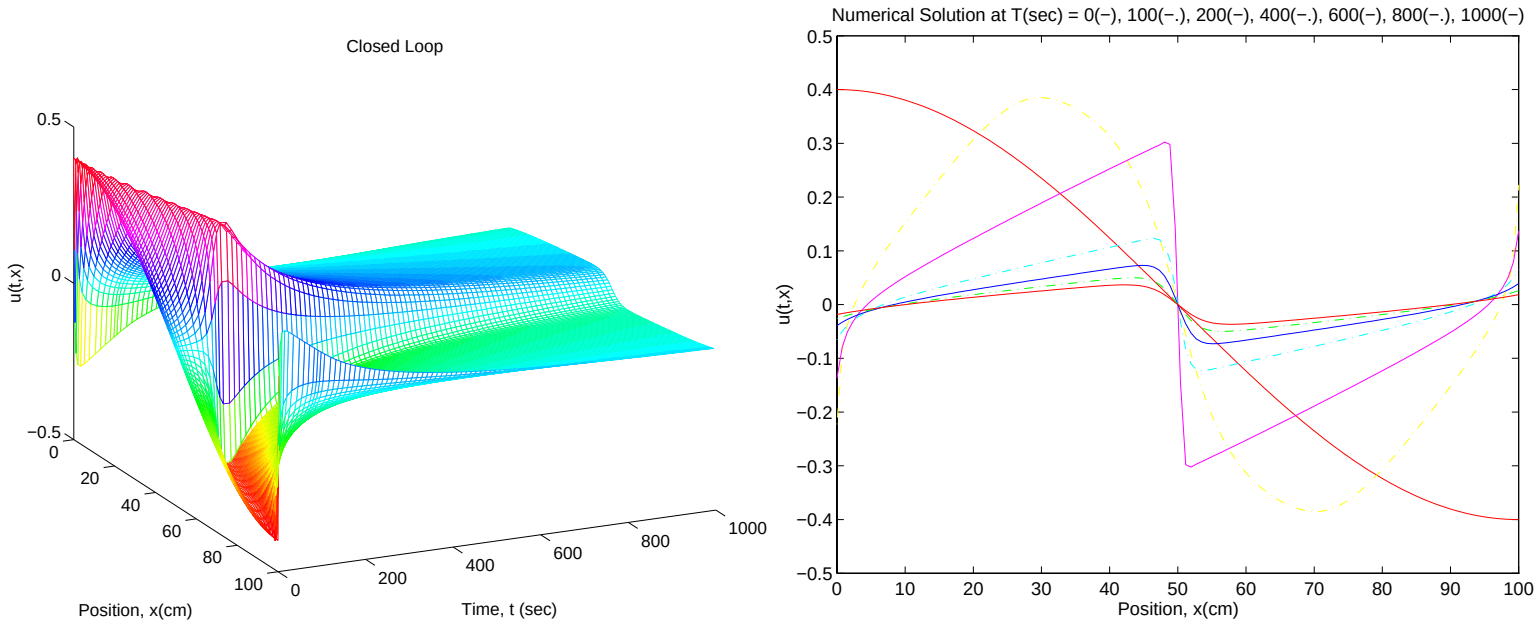


Figure 3.3.107: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{\frac{1}{10}}$, N = 128

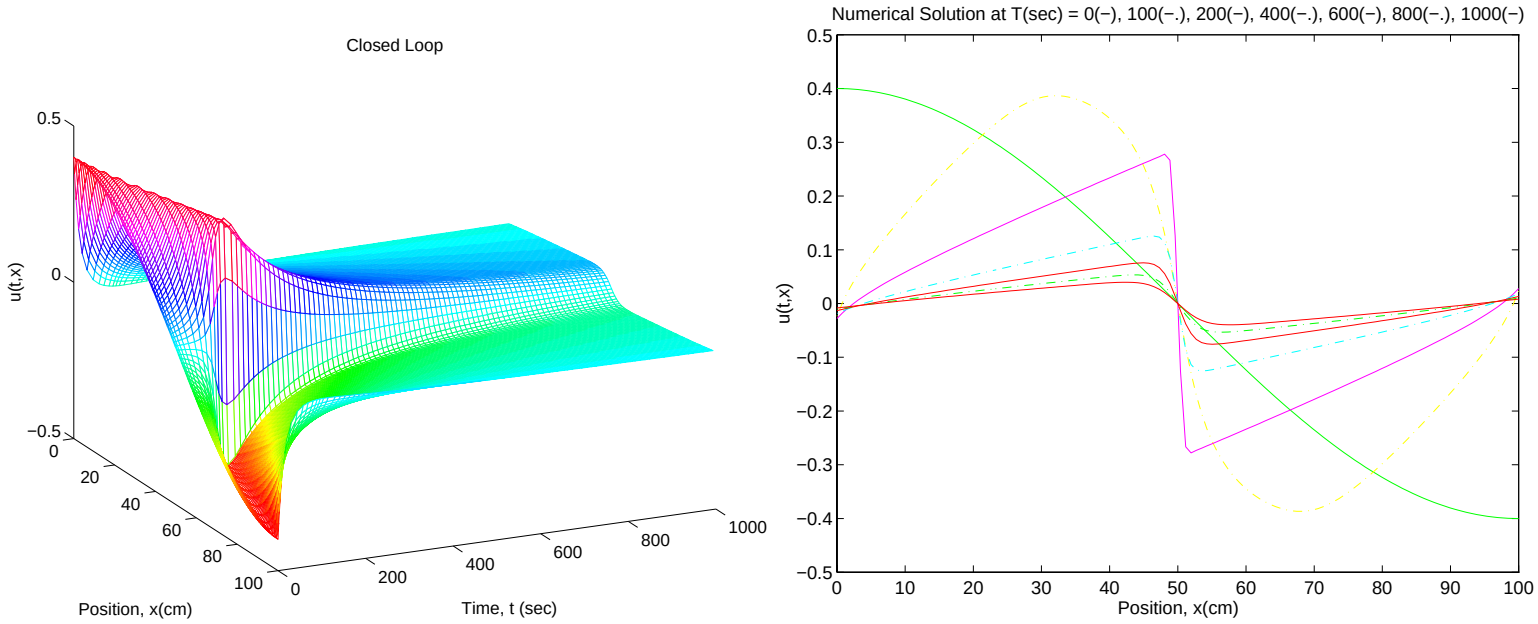


Figure 3.3.108: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{1.14}$, N = 128

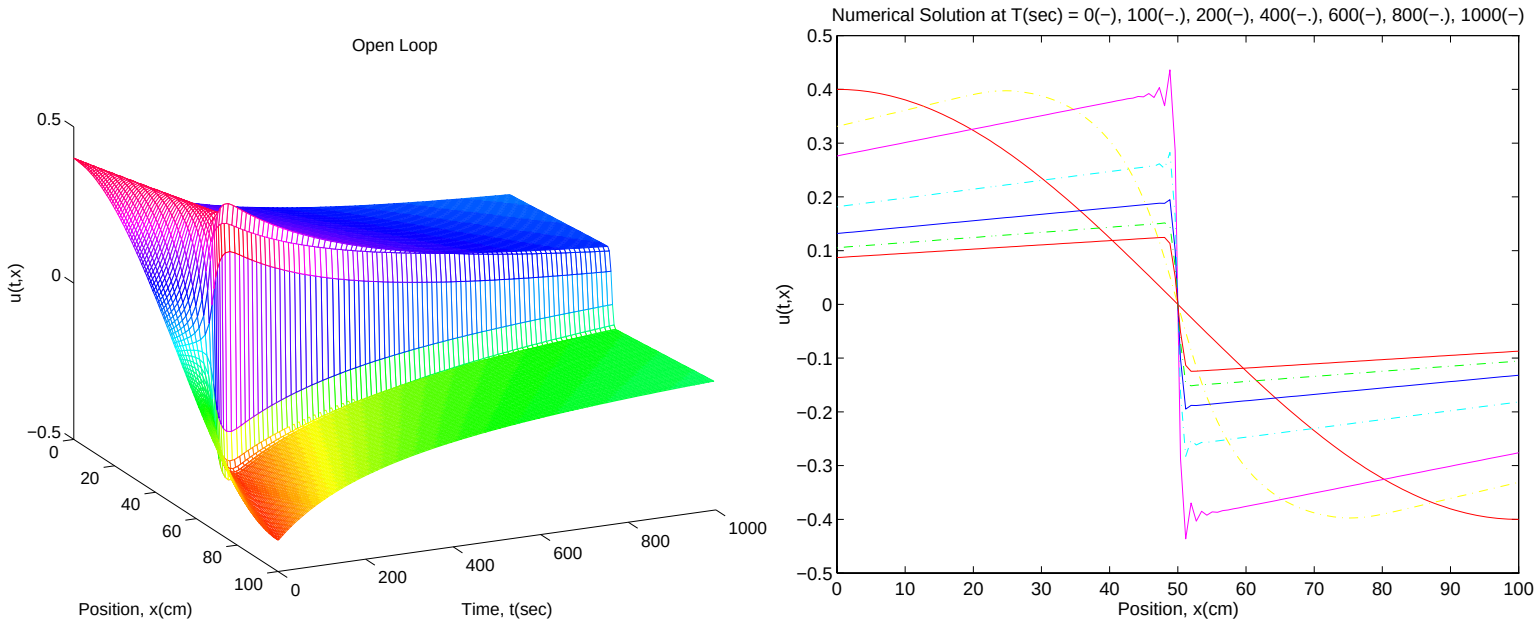


Figure 3.3.109: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, N = 128

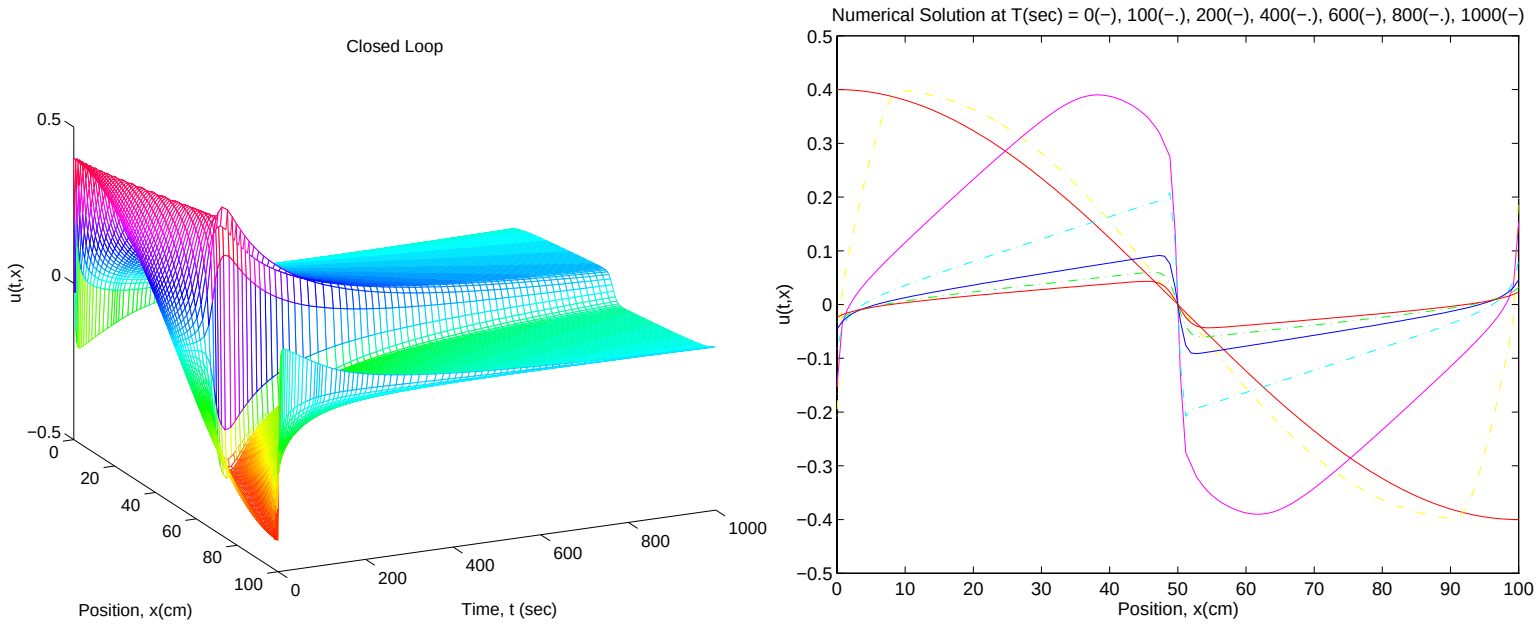


Figure 3.3.110: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{\frac{1}{20}}$, N = 128

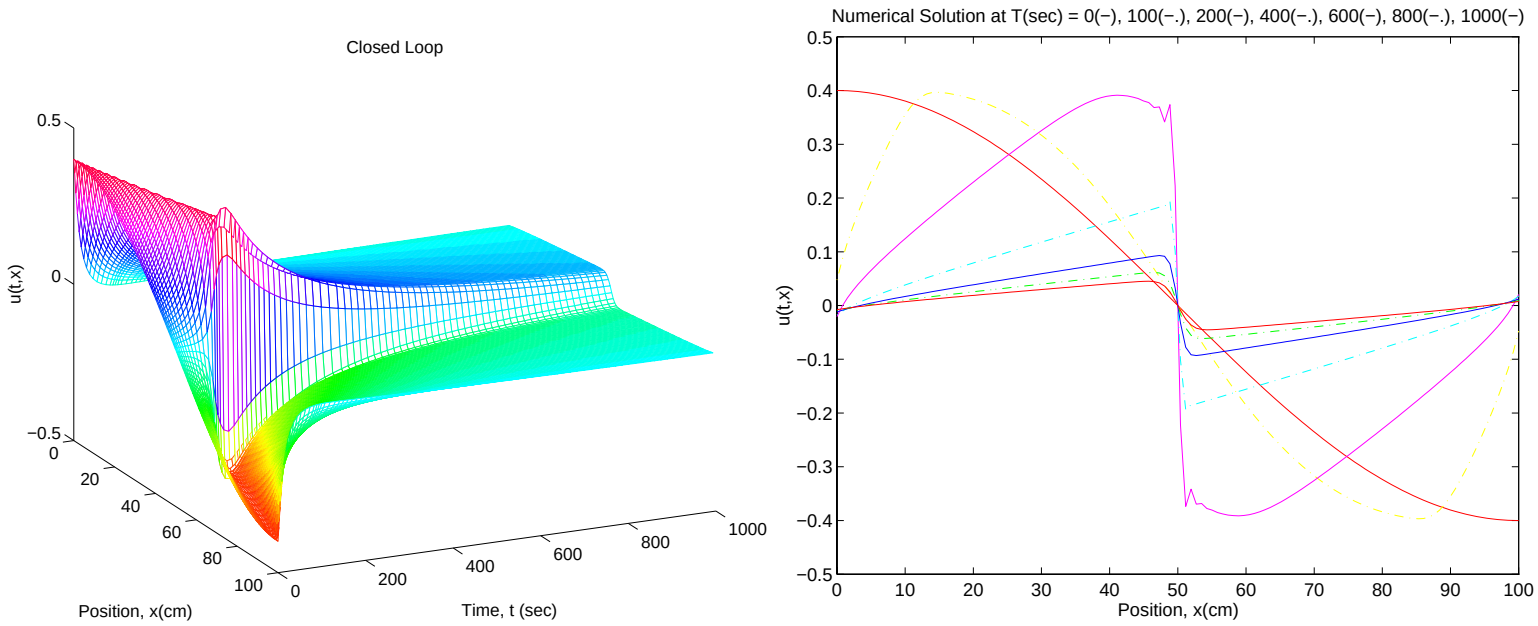


Figure 3.3.111: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{1.14}$, N = 128

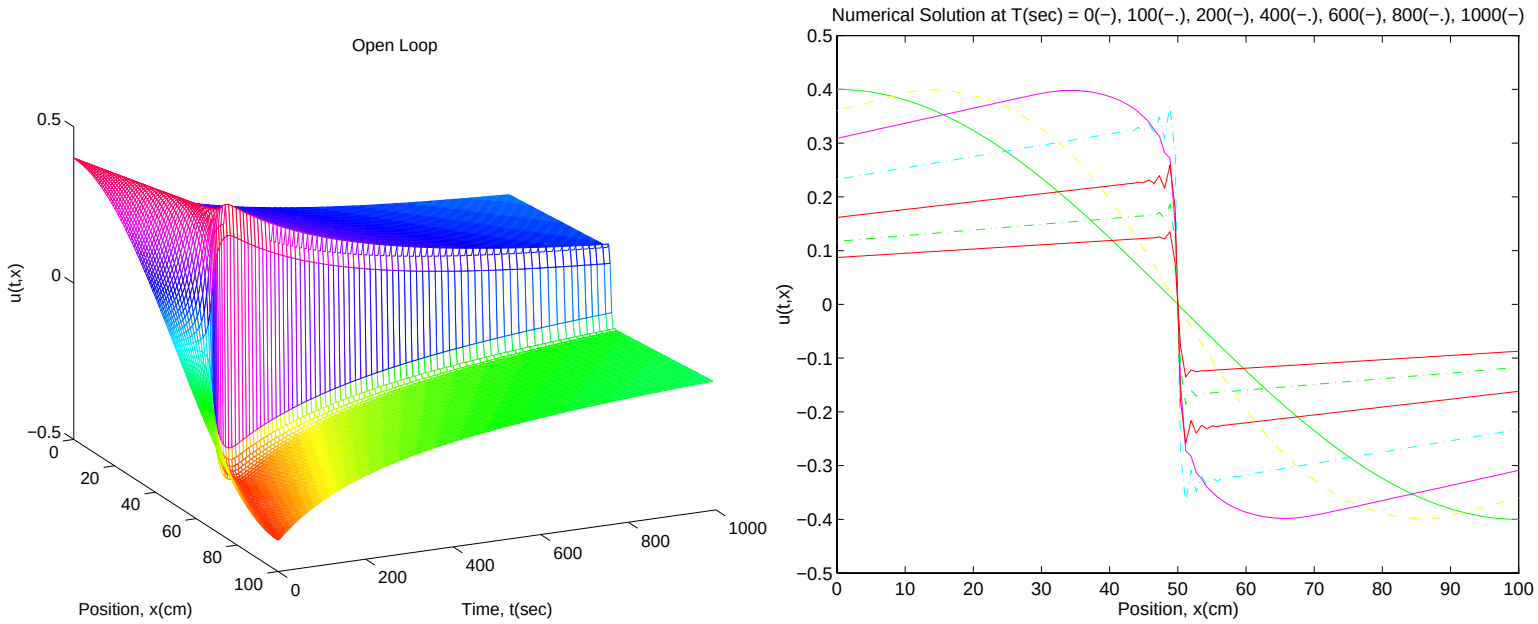


Figure 3.3.112: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, N = 128

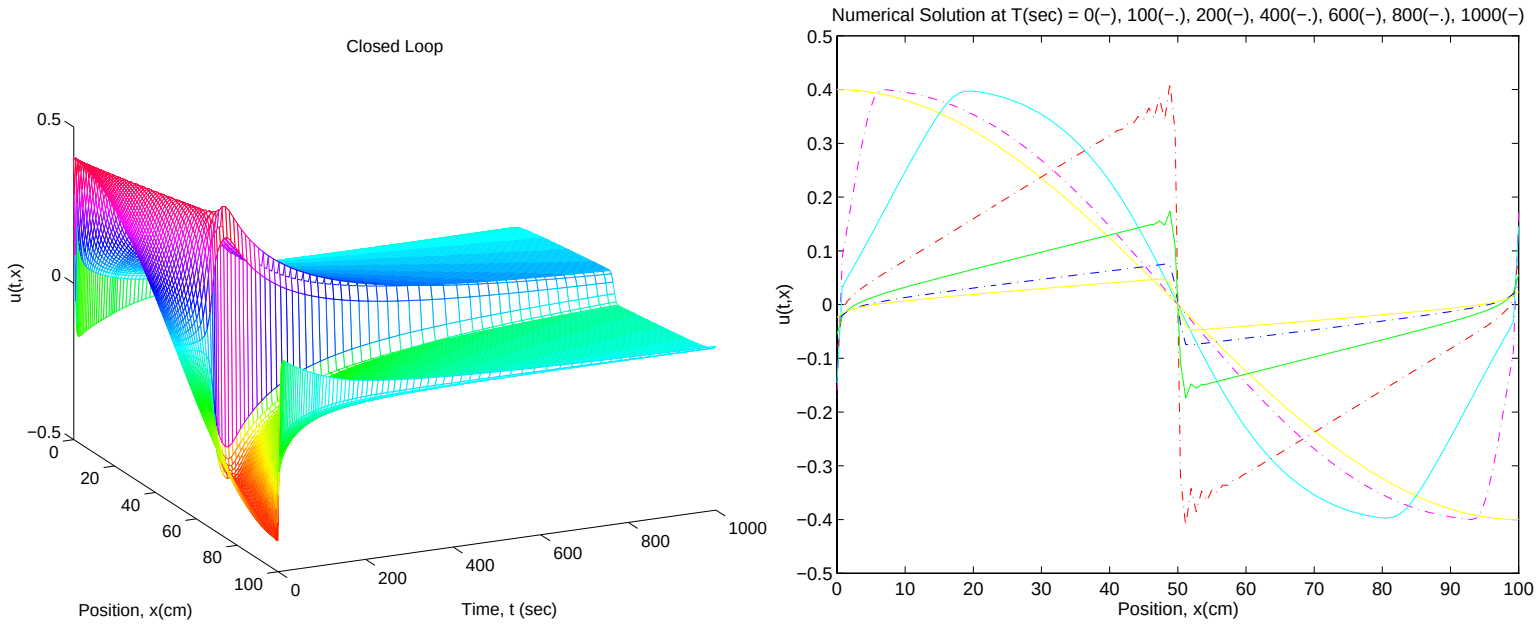


Figure 3.3.113: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{\frac{1}{40}}$, N = 128

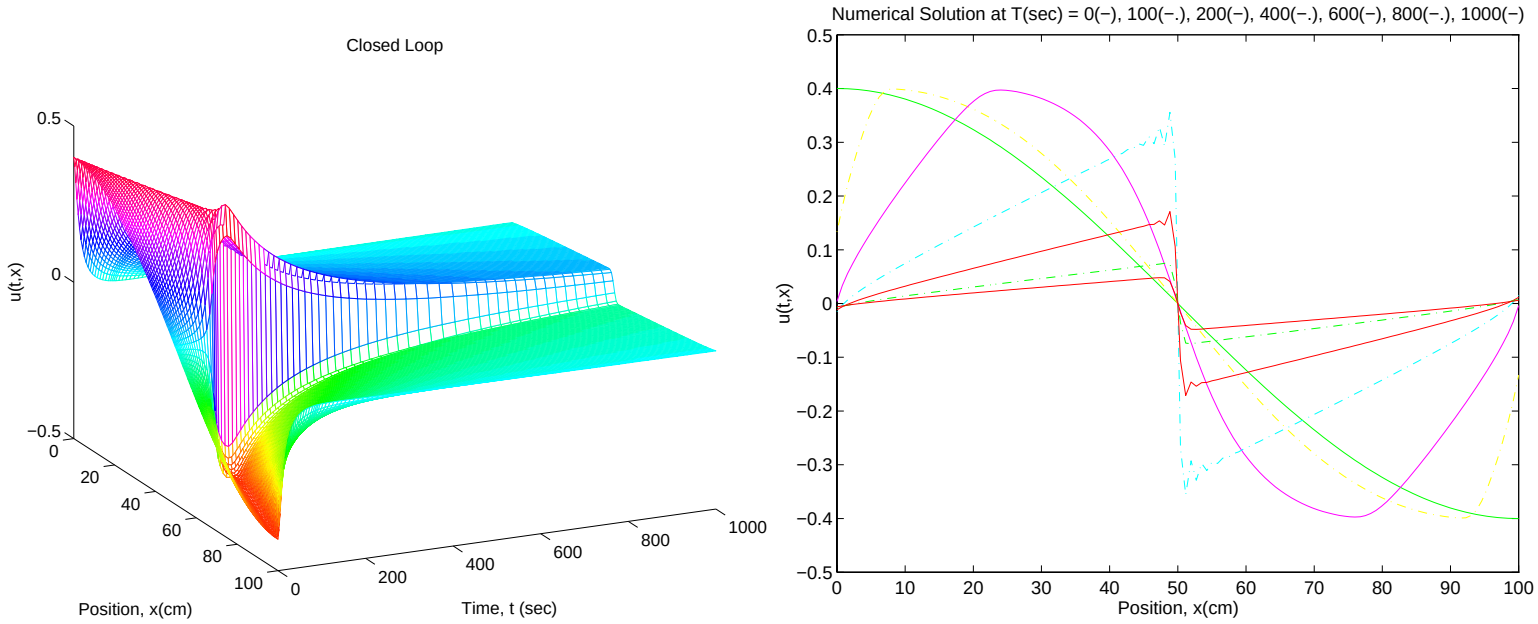


Figure 3.3.114: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{1.14}$, N = 128

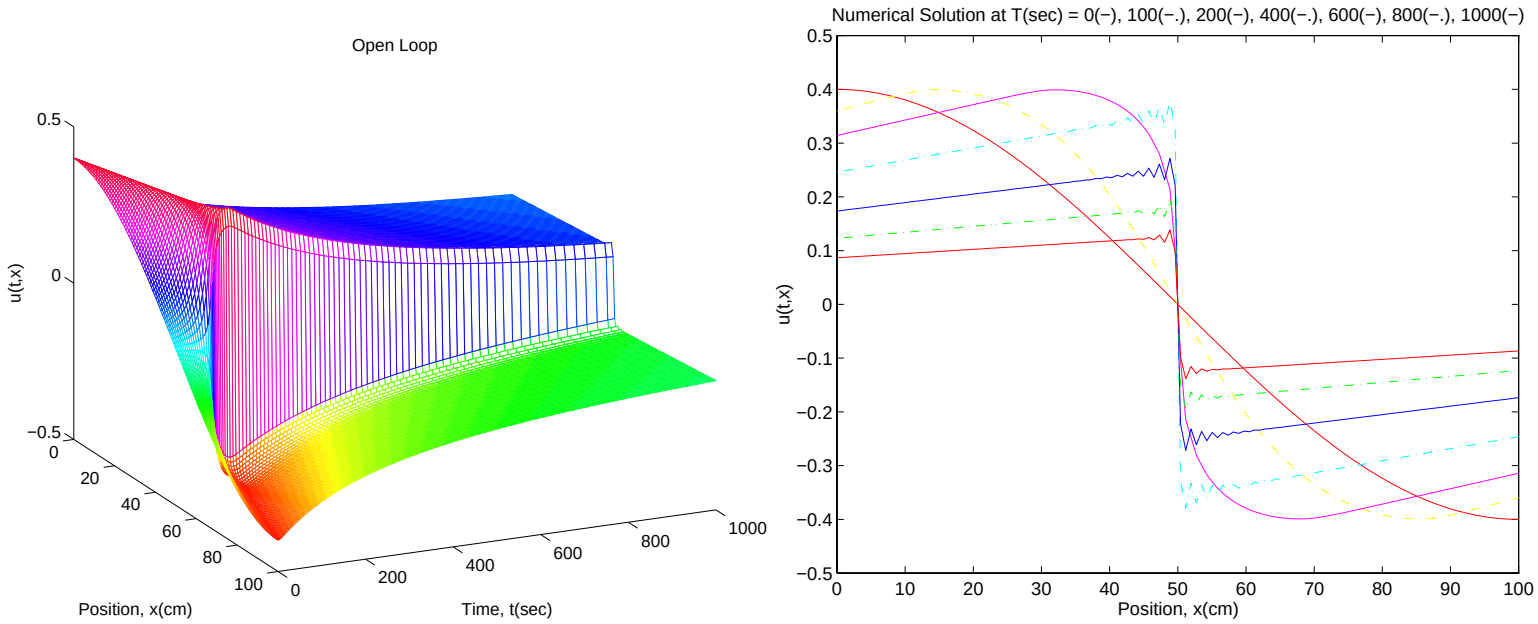


Figure 3.3.115: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, N = 128

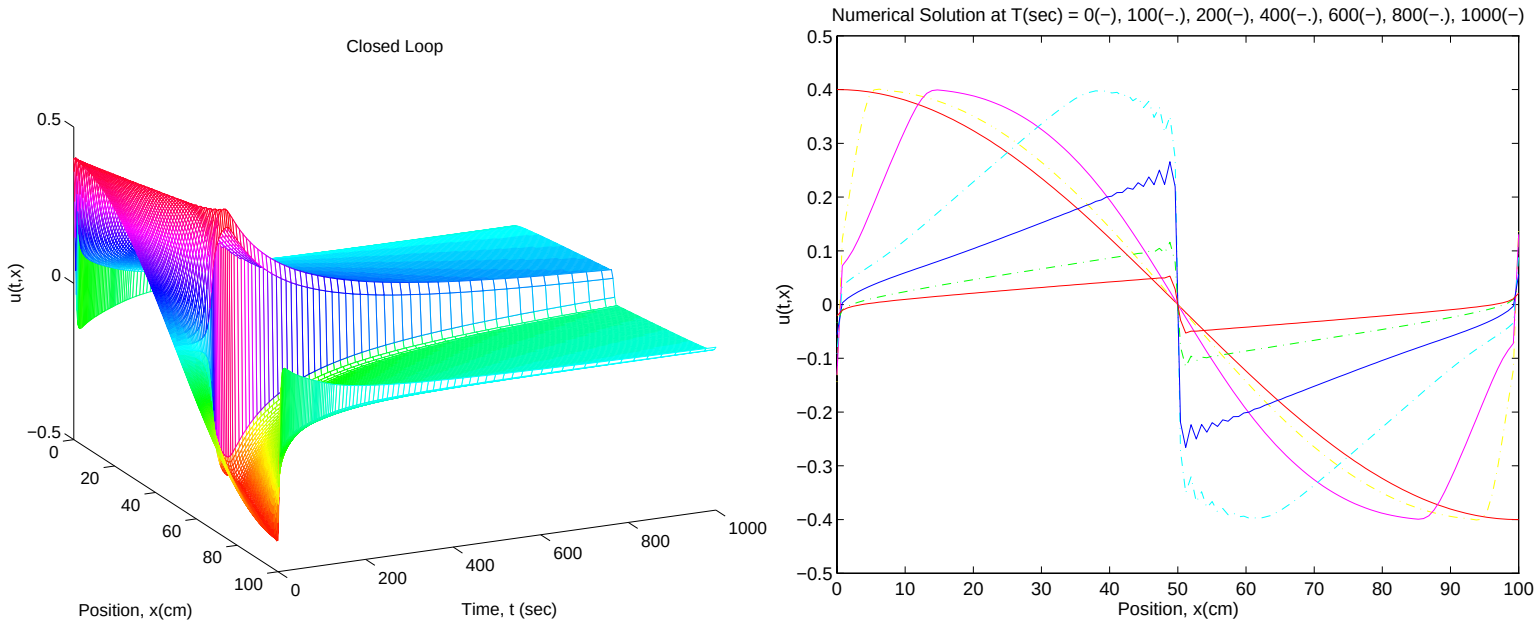


Figure 3.3.116: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{\frac{1}{80}}$, N = 128

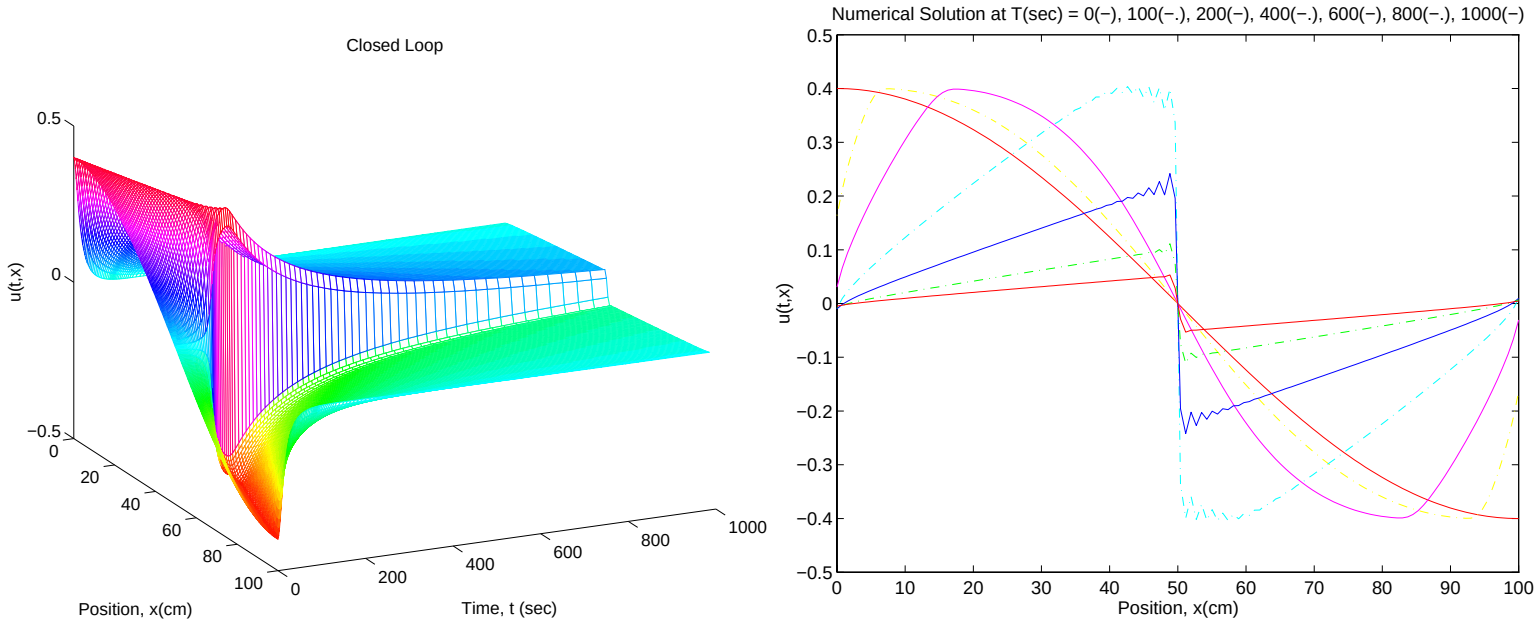


Figure 3.3.117: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{1.14}$, N = 128

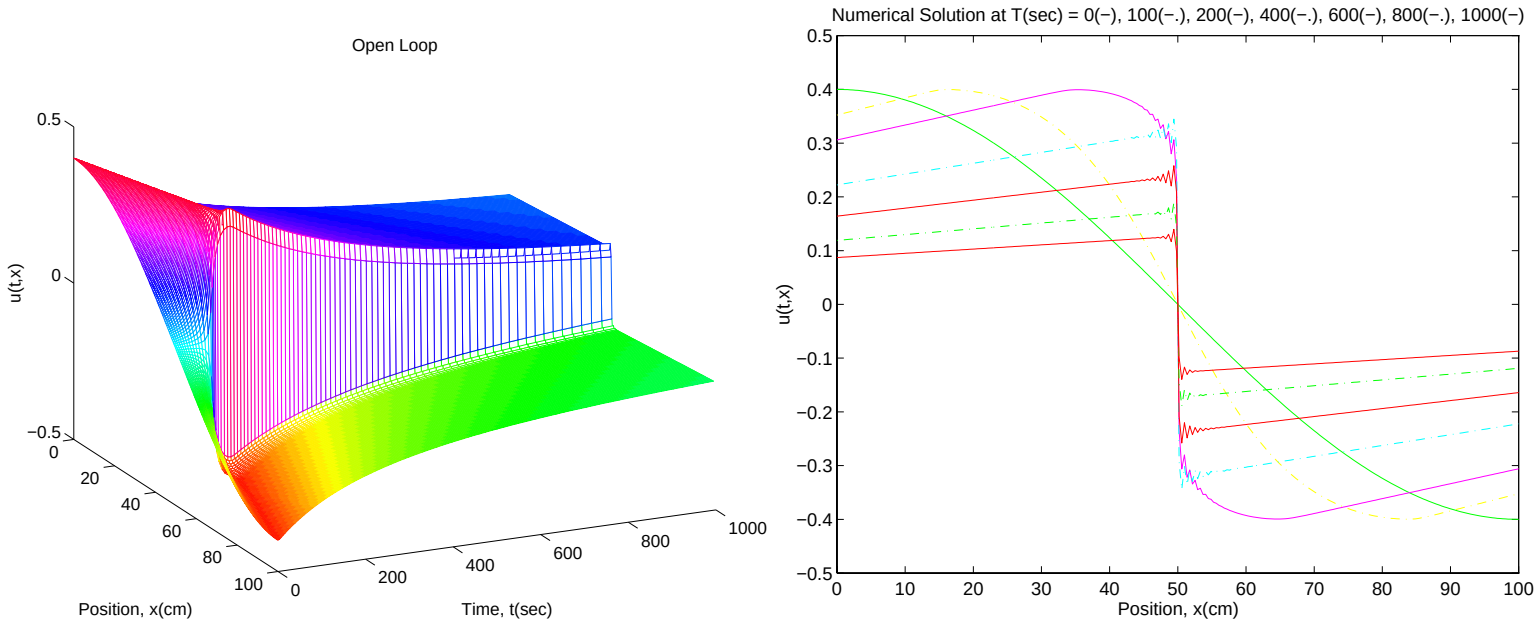


Figure 3.3.118: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, N = 256

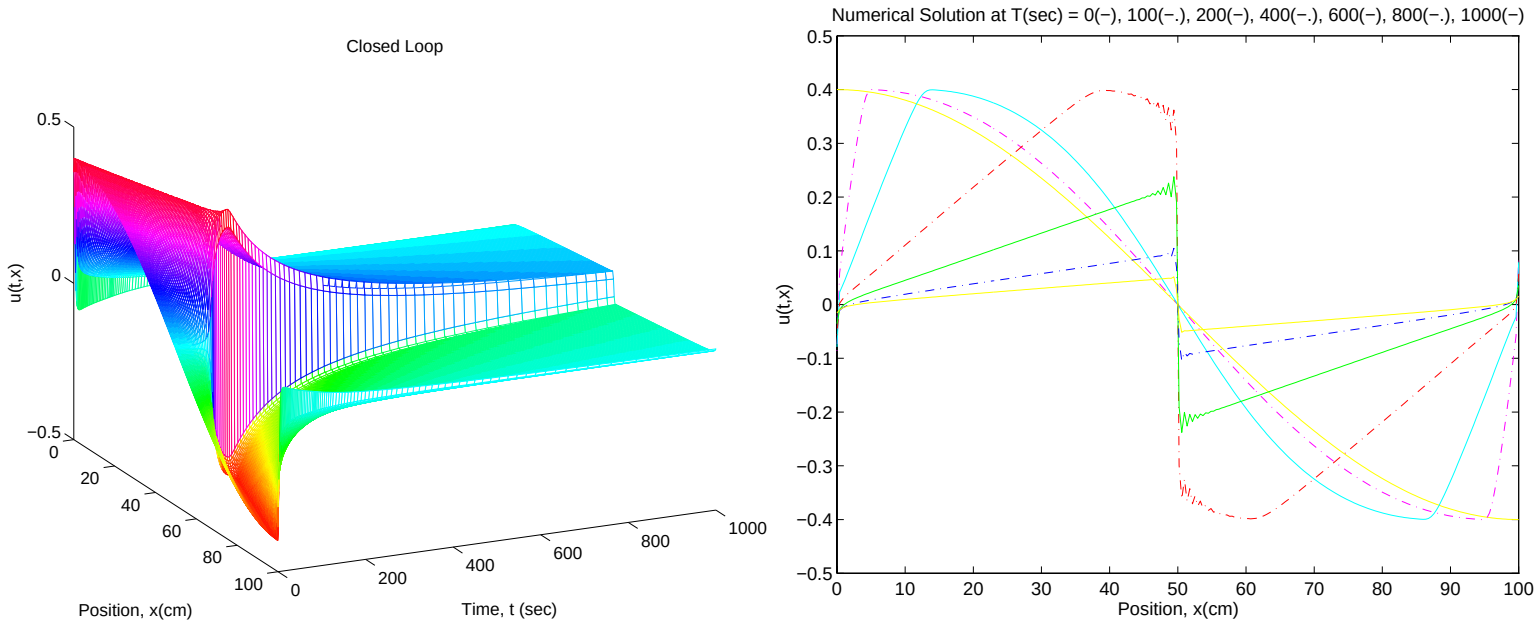


Figure 3.3.119: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{\frac{1}{160}}$, N = 256

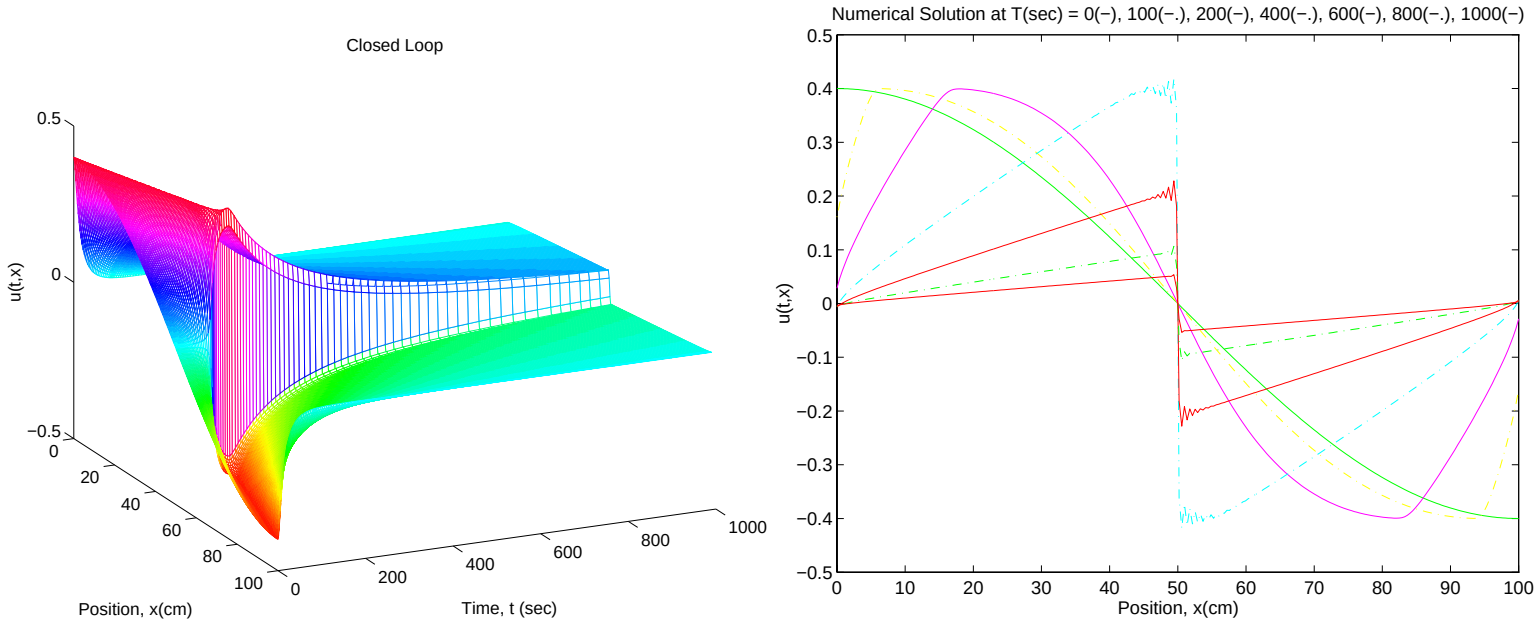


Figure 3.3.120: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{1.14}$, N = 256

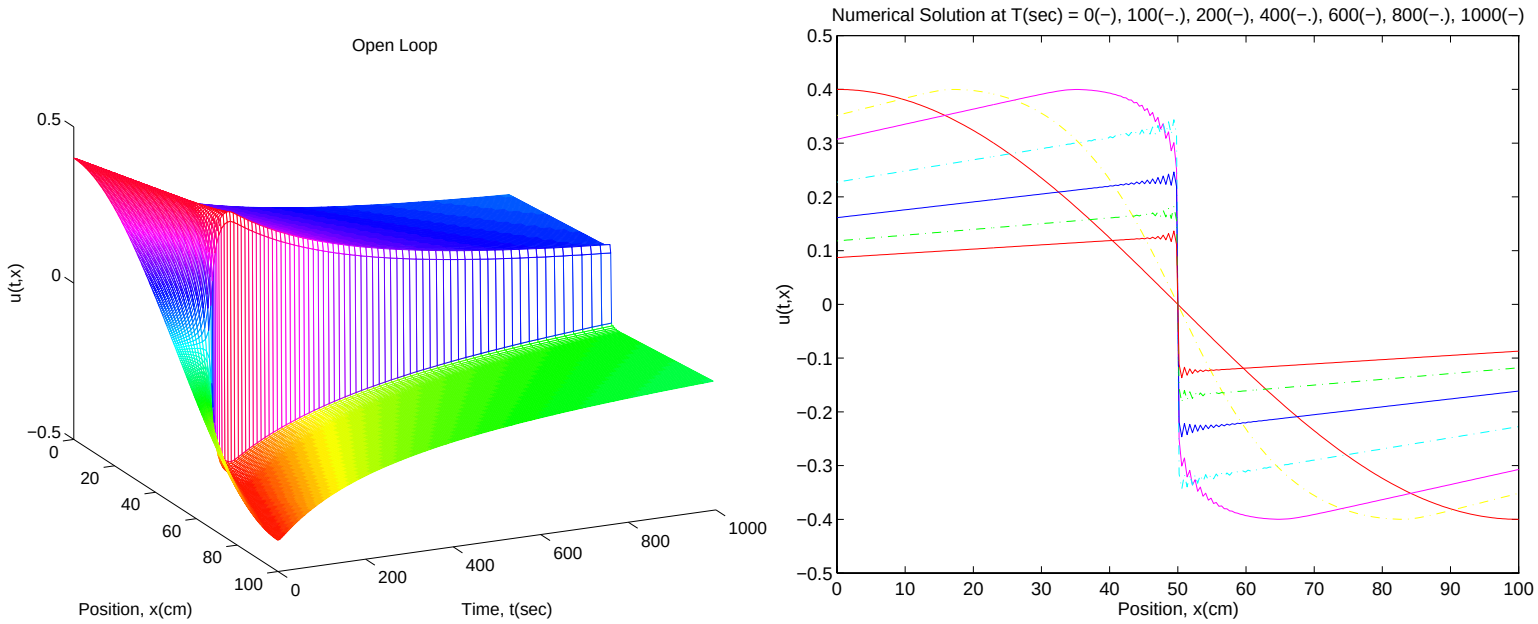


Figure 3.3.121: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, N = 256

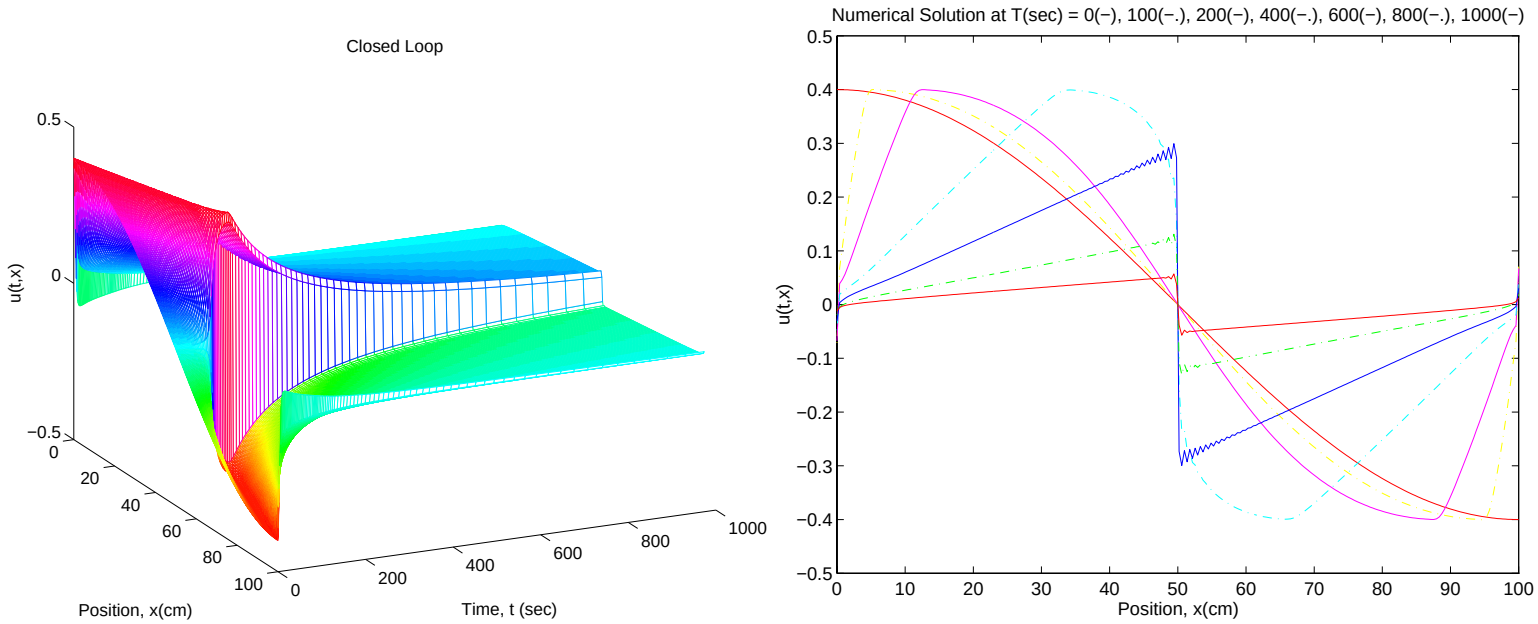


Figure 3.3.122: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{\frac{1}{320}}$, N = 256

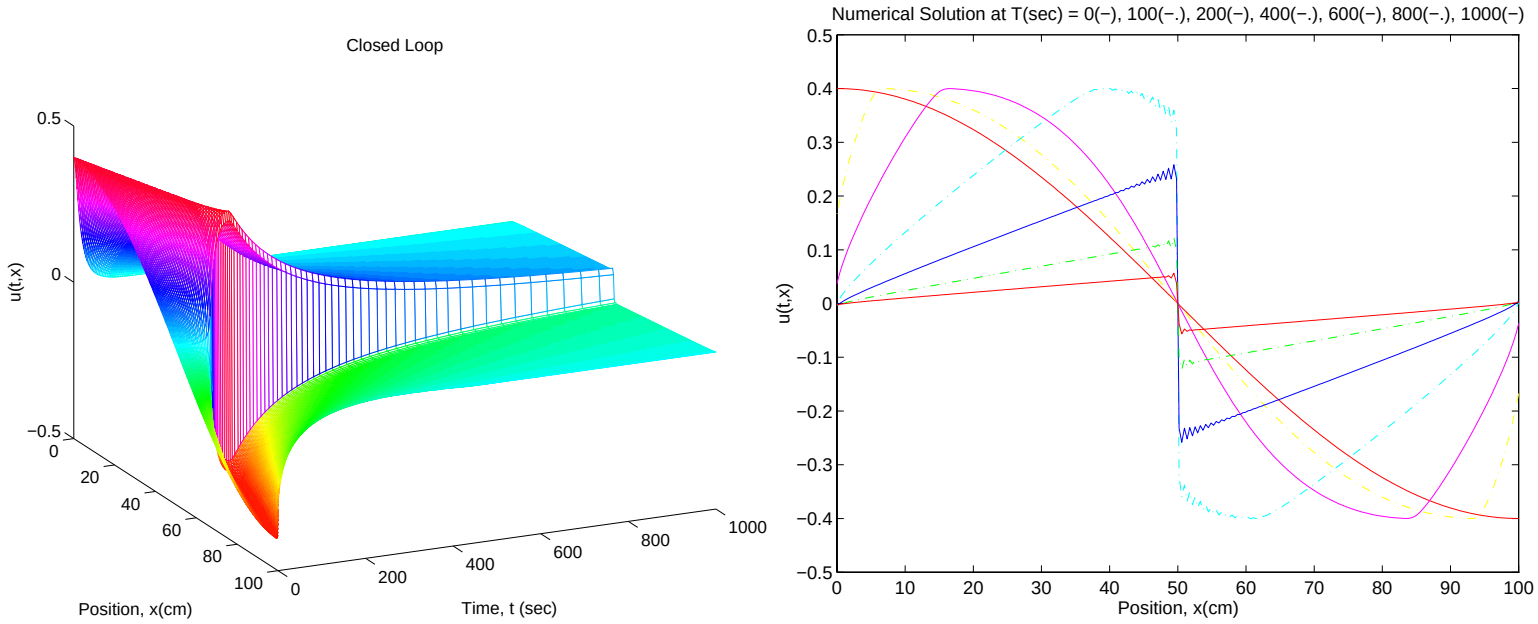


Figure 3.3.123: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{1.14}$, N = 256

### 3.3.5 Gains as a Factor of Reynolds Number

In the previous section, for a sufficiently small Reynolds number, there is a significant difference in the closed loop systems (3.22) and (3.23). This is due to the large difference in the two gain matrices for the systems, $K_\epsilon$ and $K_{1.14}$. Using the infinity norm to look at the difference between the two gain matrices when $\epsilon$ is varied from .939 to $\frac{1}{320}$ results in the following:

$$\| K_\epsilon - K_{1.14} \|_\infty \sim 35$$

for the aluminum rod with ten meter iron films, and

$$\| K_\epsilon - K_{1.14} \|_\infty \sim 44$$

for twenty-five centimeter iron films.

For the two film lengths, the previous infinity norm for each $\epsilon$ is approximately the same. This indicates that for each film length, the gain matrix, $K_\epsilon$, changes very little for a change in $\epsilon$. Using the Reynolds numbers 10, 20, 40, 80, 160, and 320 to vary $\epsilon$, $K_\epsilon$ is compared to $K_{.939}$. Again, using the infinity norms,

$$\| K_{.939} - K_{\frac{1}{Re}} \|_\infty \sim 10^{-8}$$

for ten meter iron films, and

$$\| K_{.939} - K_{\frac{1}{Re}} \|_\infty \sim 10^{-11}$$

for twenty-five centimeter iron films.

The similarities of these particular gains is illustrated by Figures 3.3.124 and 3.3.125 which are the functional gains for $\epsilon = .939$ and $\epsilon = \frac{1}{320}$ with ten meter films. The functional gains for $\epsilon = .939$ and $\epsilon = \frac{1}{320}$ with twenty-five centimeter films can be seen in Figures 3.3.126 and 3.3.127, respectively. In both cases the functional gains are almost identical for the different Reynolds numbers.

Since the gain matrices, $K_{\frac{1}{Re}}$, are almost identical for each film length, then the numerical solutions to the closed loop systems

$$\dot{\alpha}(t)^N(t) = (A^N_{\frac{1}{Re}} - B^N_{\frac{1}{Re}} K_{\frac{1}{Re}})\alpha(t) - \mathcal{F}^N_D(\alpha(t)) \tag{3.24}$$

and

$$\dot{\alpha}(t)^N(t) = (A^N_{\frac{1}{Re}} - B^N_{\frac{1}{Re}} K_{.939})\alpha(t) - \mathcal{F}^N_D(\alpha(t)) \tag{3.25}$$

are very similar, as well.

Figures 3.3.128, 3.3.129, 3.3.130, 3.3.131, 3.3.132, and 3.3.133 are the numerical solutions to (3.25) with ten meter iron films for Reynolds numbers of 10, 20, 40, 80, 160, and 320, respectively. Comparing them to the respective numerical solutions with optimal control (3.24) in Figures 3.3.89, 3.3.92, 3.3.95, 3.3.98, 3.3.101, and 3.3.104, shows that the numerical solutions to both systems are nearly identical. For example, Figures 3.3.128 and 3.3.89 have amplitudes of .0136 and .0120, respectively. As another example, Figures 3.3.132 and 3.3.101 both have amplitudes of .0183.

For film lengths of twenty-five centimeters, the numerical solutions to (3.24) and (3.25) are even closer. Figures 3.3.134, 3.3.135, 3.3.136, 3.3.137, 3.3.138, and 3.3.139 illustrate

the numerical solutions to (3.25). Their respective numerical solutions to (3.24) are Figures 3.3.107, 3.3.110, 3.3.113, 3.3.116, 3.3.119, and 3.3.122. For each Reynolds number, $\{10, 20, 40, 80, 160, \text{ and } 320\}$, the amplitudes for the numerical solutions to (3.25) and (3.24) are the same to at least four significant digits.

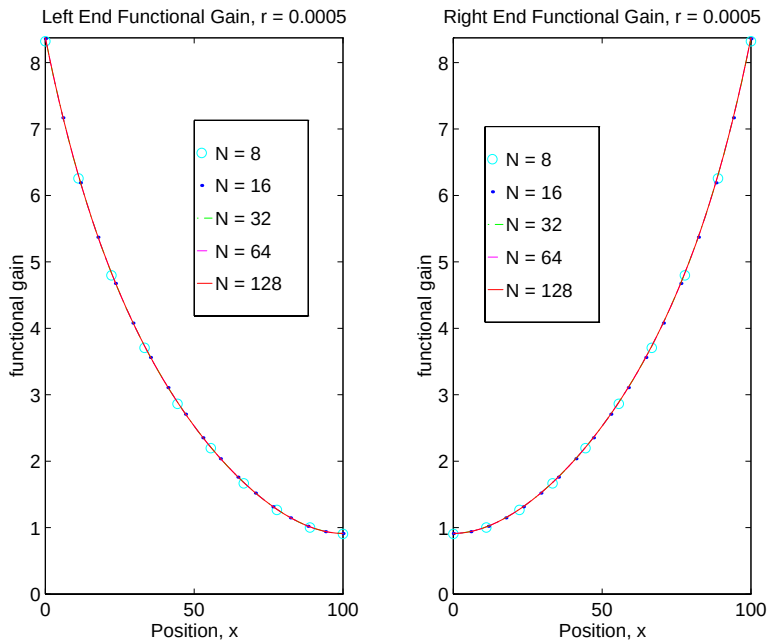


Figure 3.3.124: Functional gains, r = .0005 for a 1m Al rod with 10m Fe films, $\epsilon = .939$

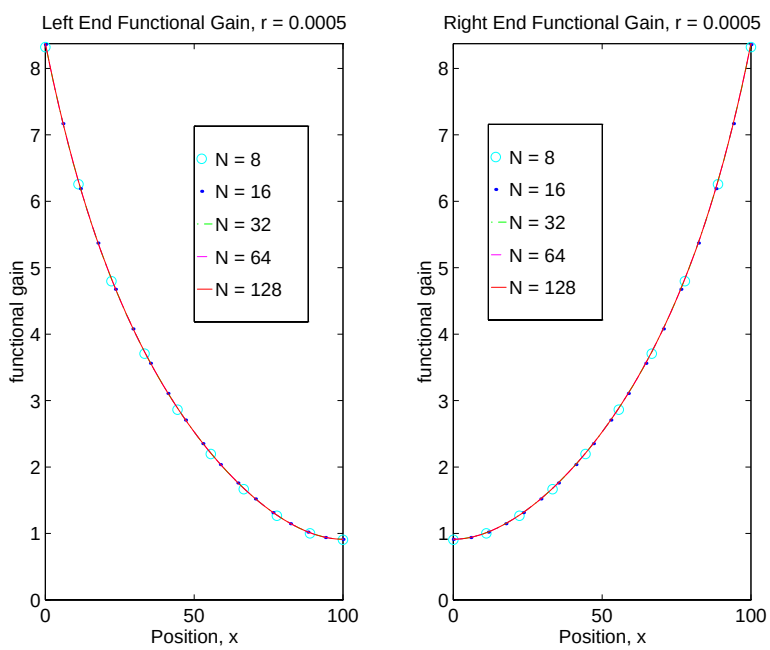


Figure 3.3.125: Functional gains, r = .0005 for a 1m Al rod with 10m Fe films, $\epsilon = \frac{1}{320}$

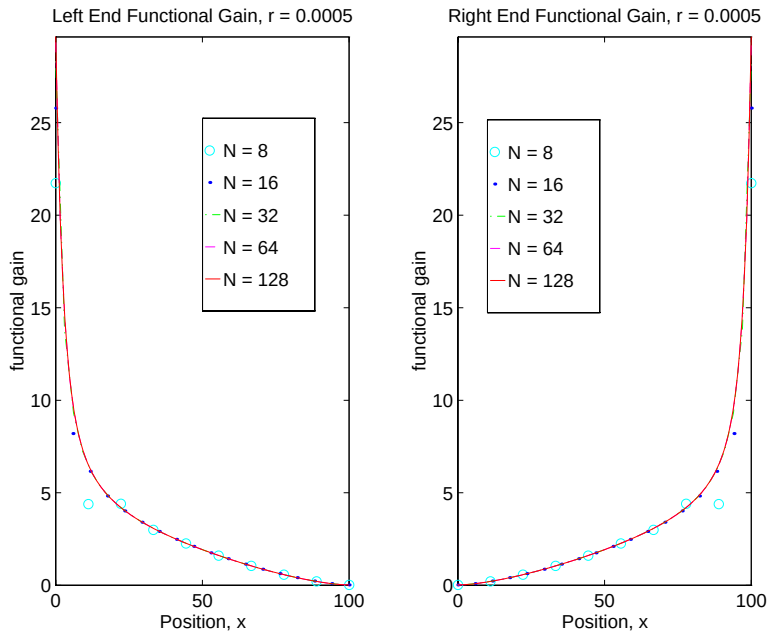


Figure 3.3.126: Functional gains, r = .0005 for a 1m Al rod with 25cm Fe films, $\epsilon = .939$

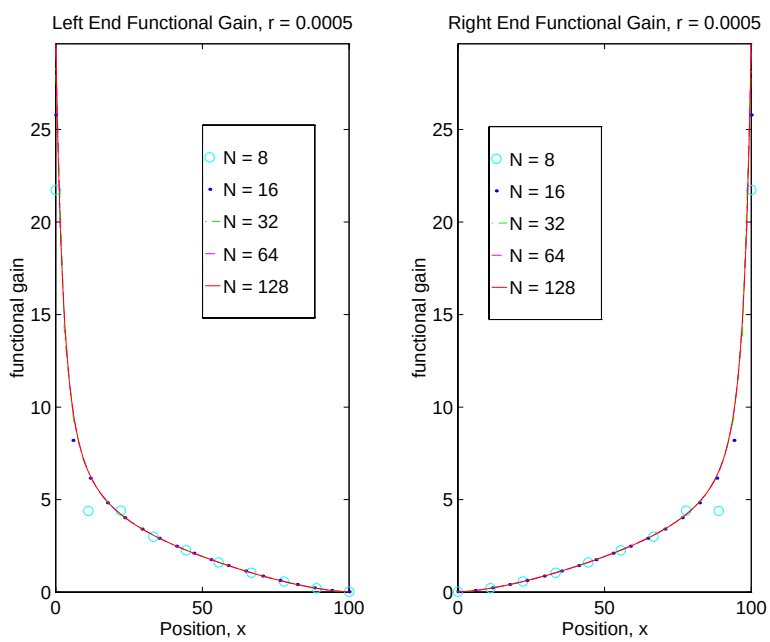


Figure 3.3.127: Functional gains, r = .0005 for a 1m Al rod with 25cm Fe films, $\epsilon = \frac{1}{320}$

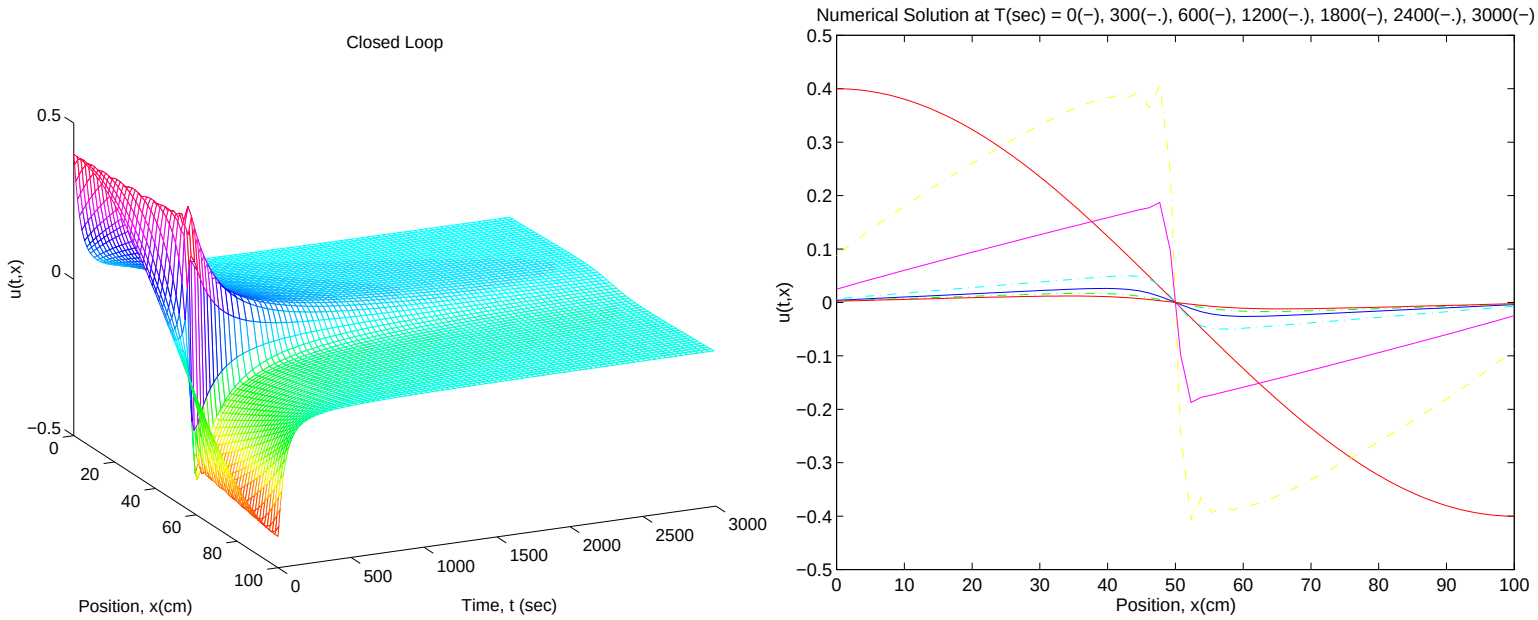


Figure 3.3.128: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{.939}$, N = 64

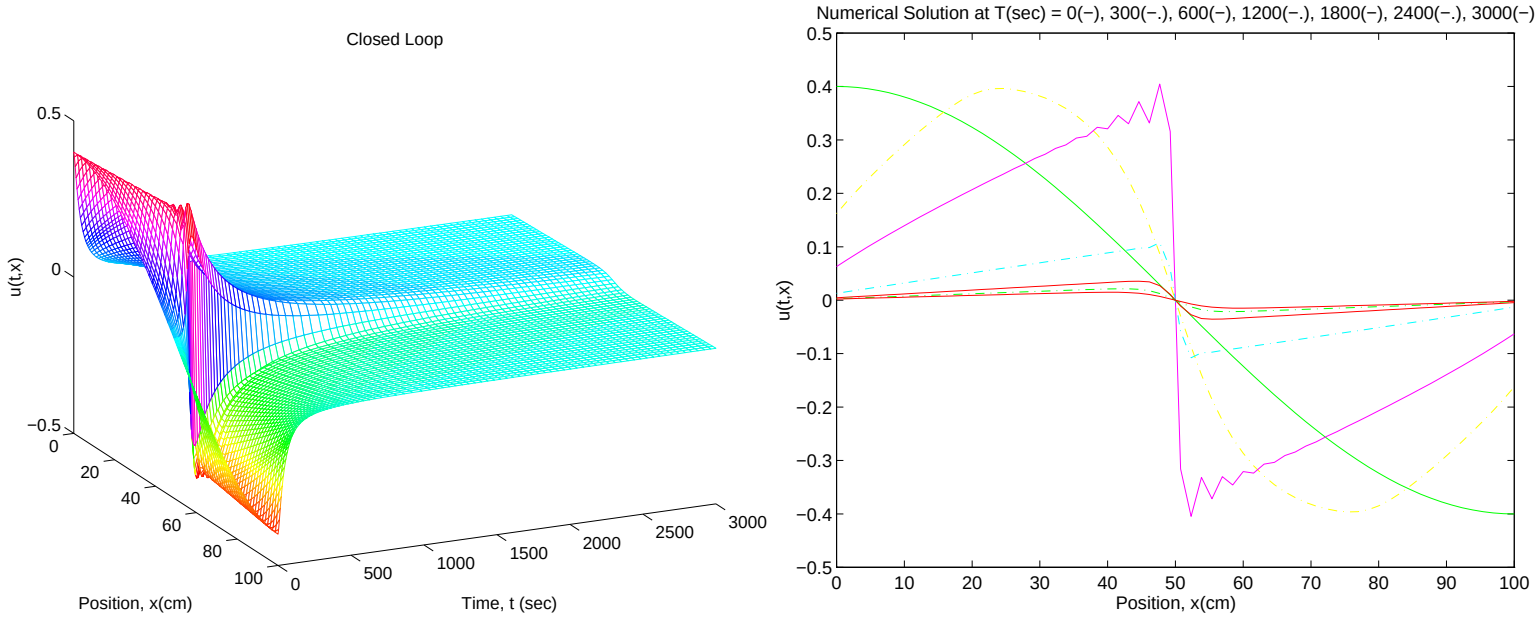


Figure 3.3.129: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{.939}$, N = 64

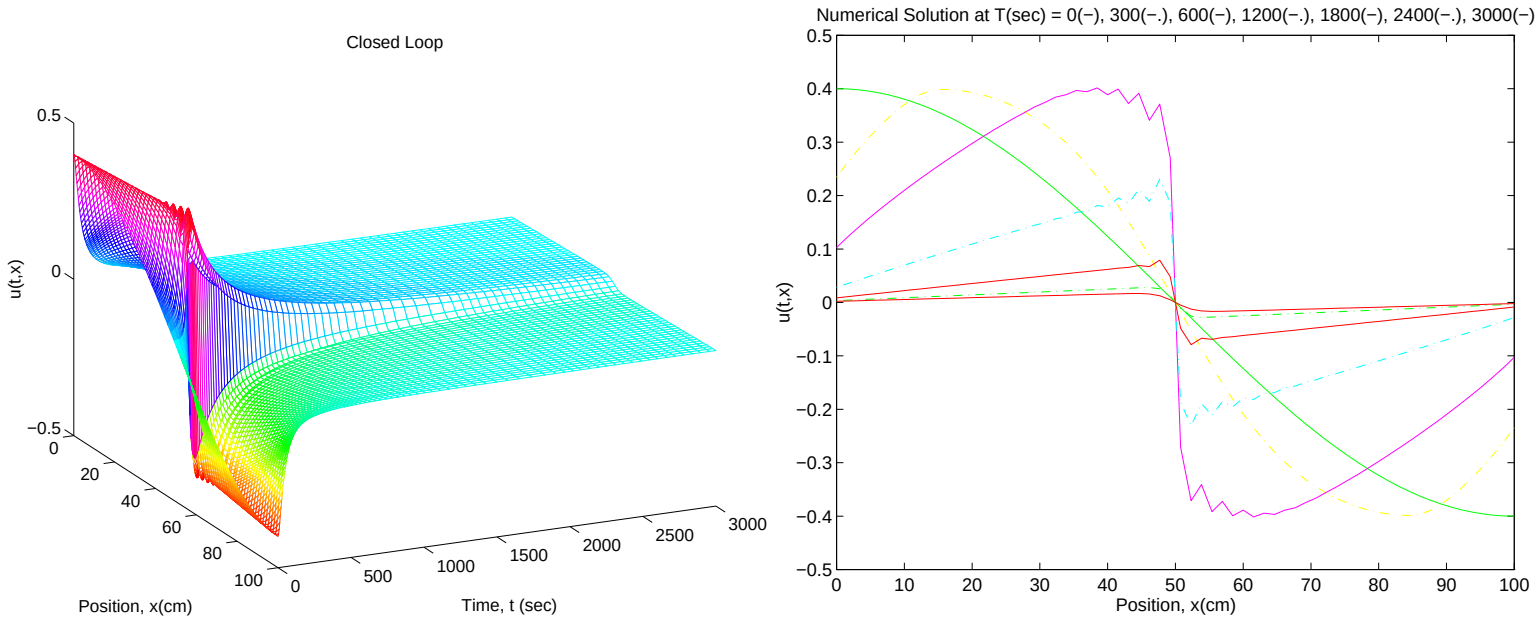


Figure 3.3.130: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{.939}$, N = 64

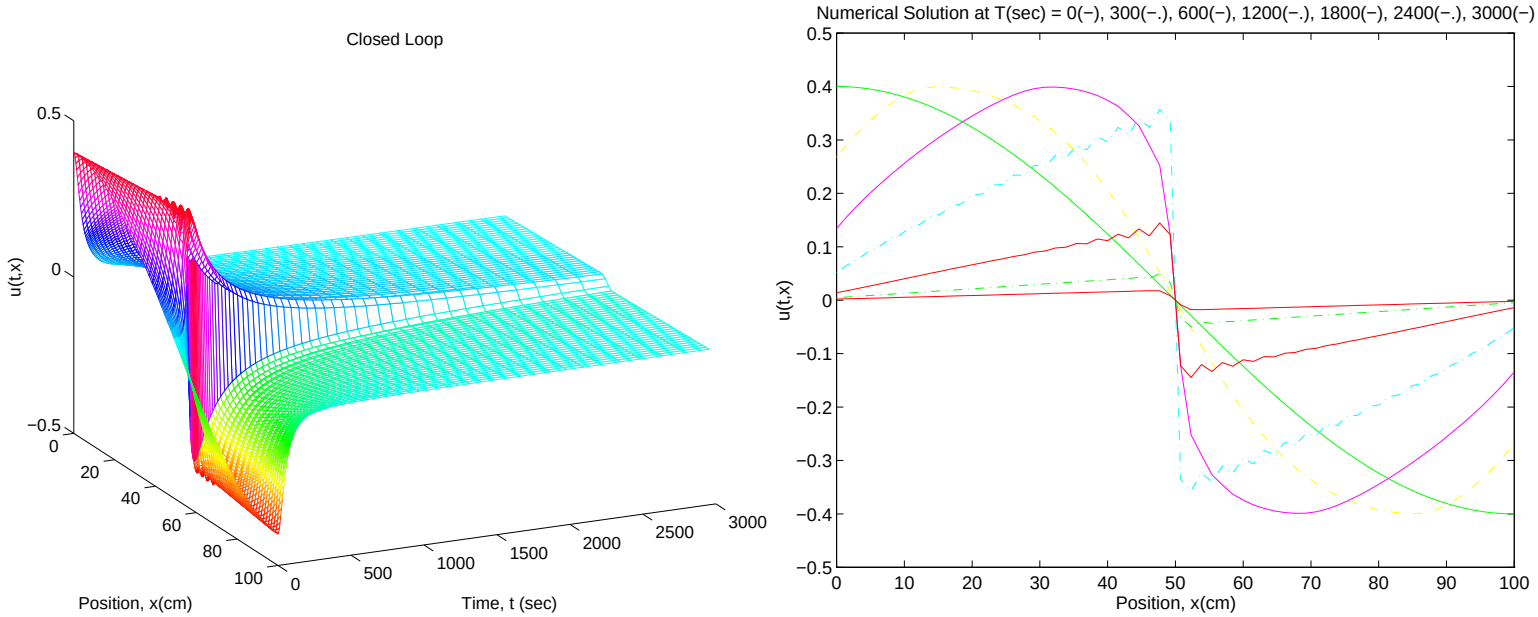


Figure 3.3.131: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{.939}$, N = 64

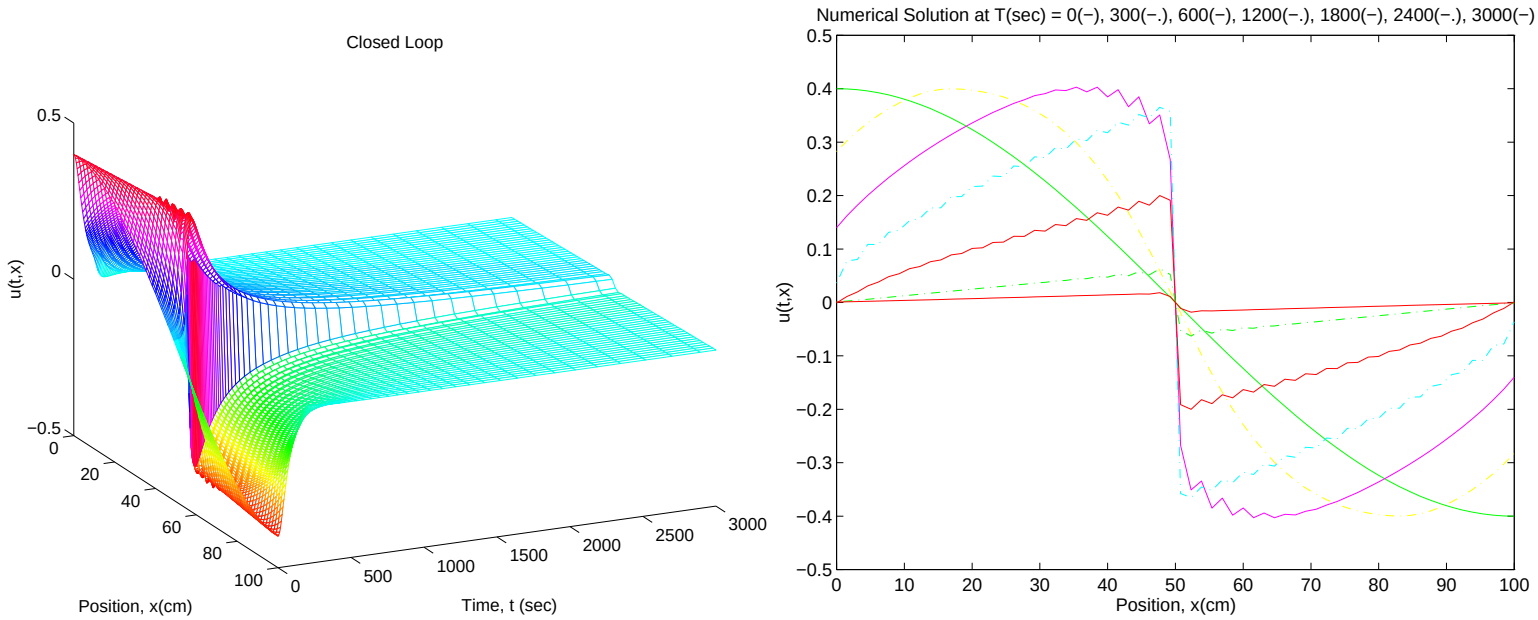


Figure 3.3.132: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{.939}$, N = 128

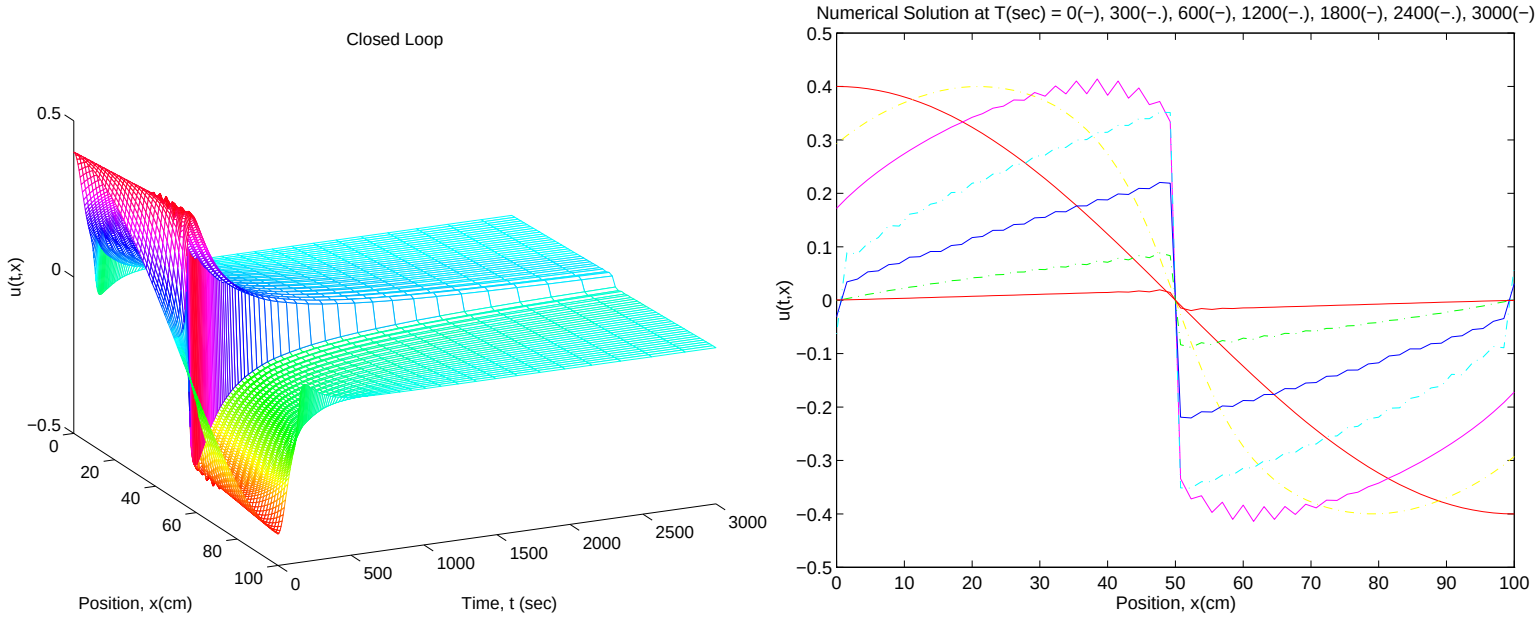


Figure 3.3.133: 1m Al rod, 10m Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{.939}$, N = 256

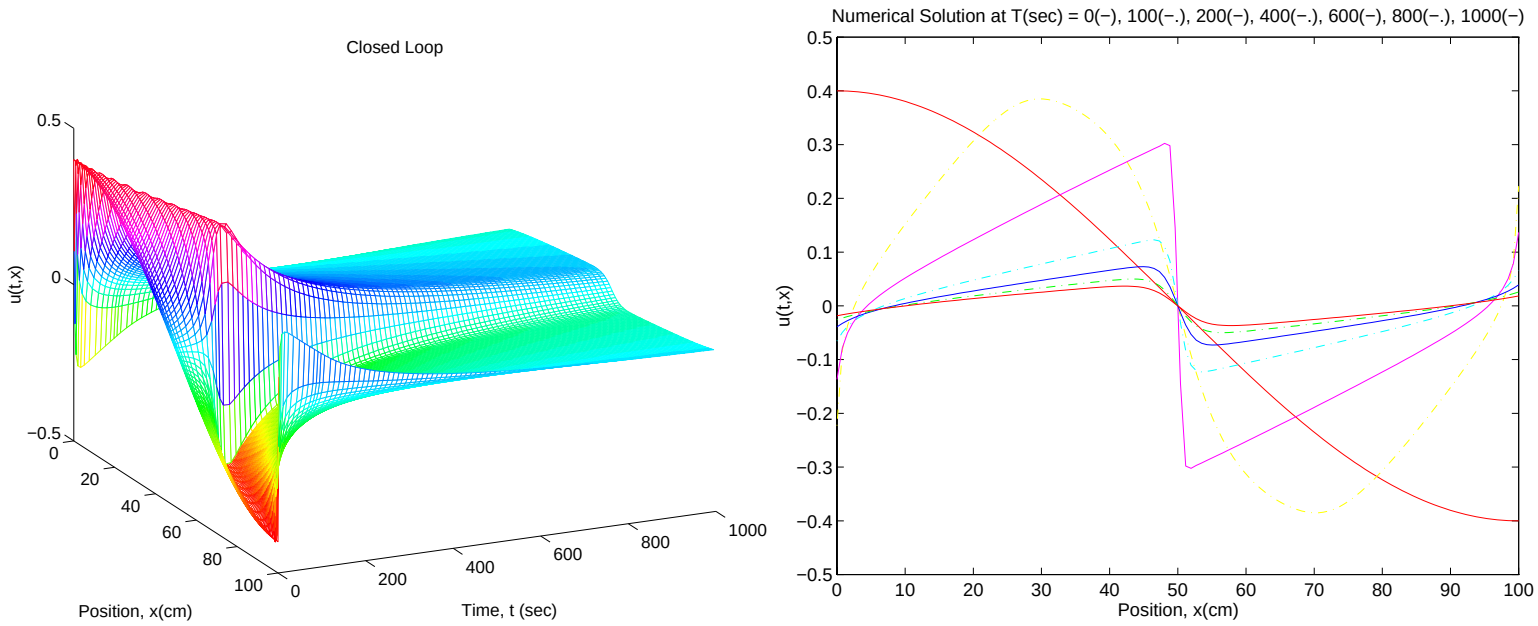


Figure 3.3.134: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 10, $K_{.939}$, N = 128

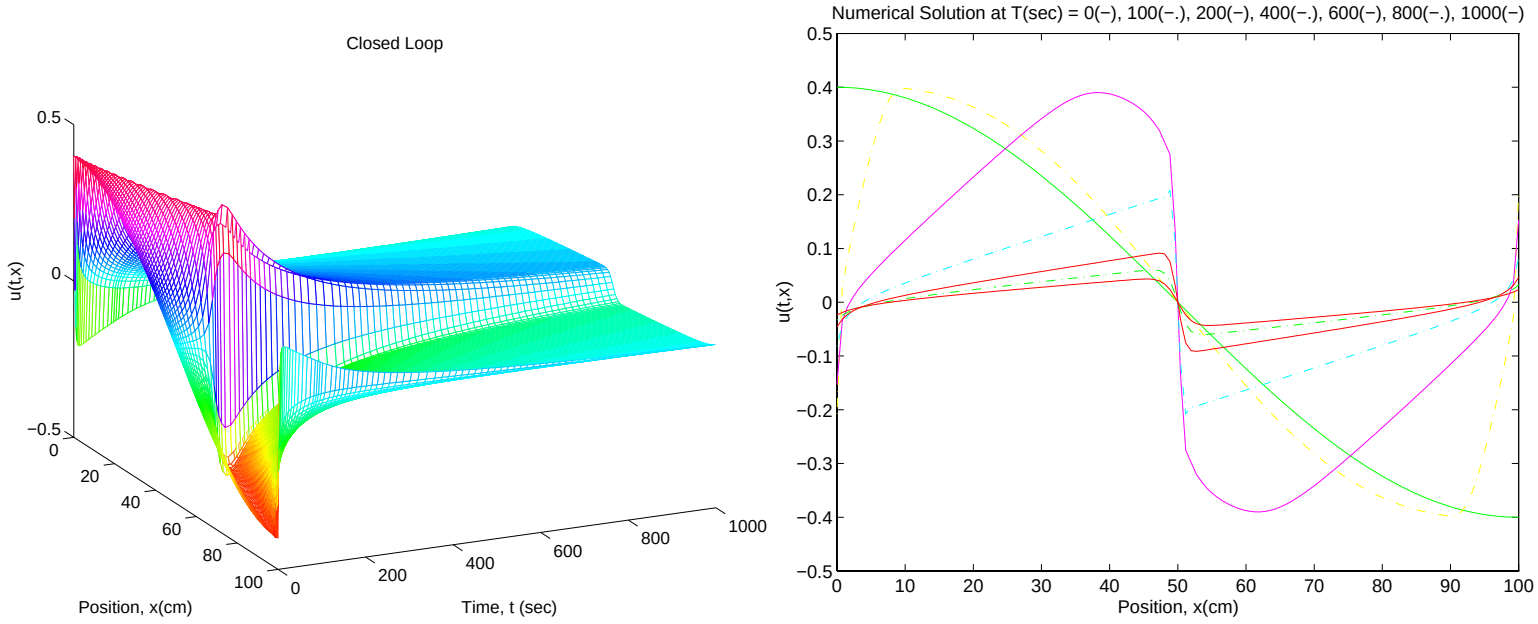


Figure 3.3.135: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 20, $K_{.939}$, N = 128

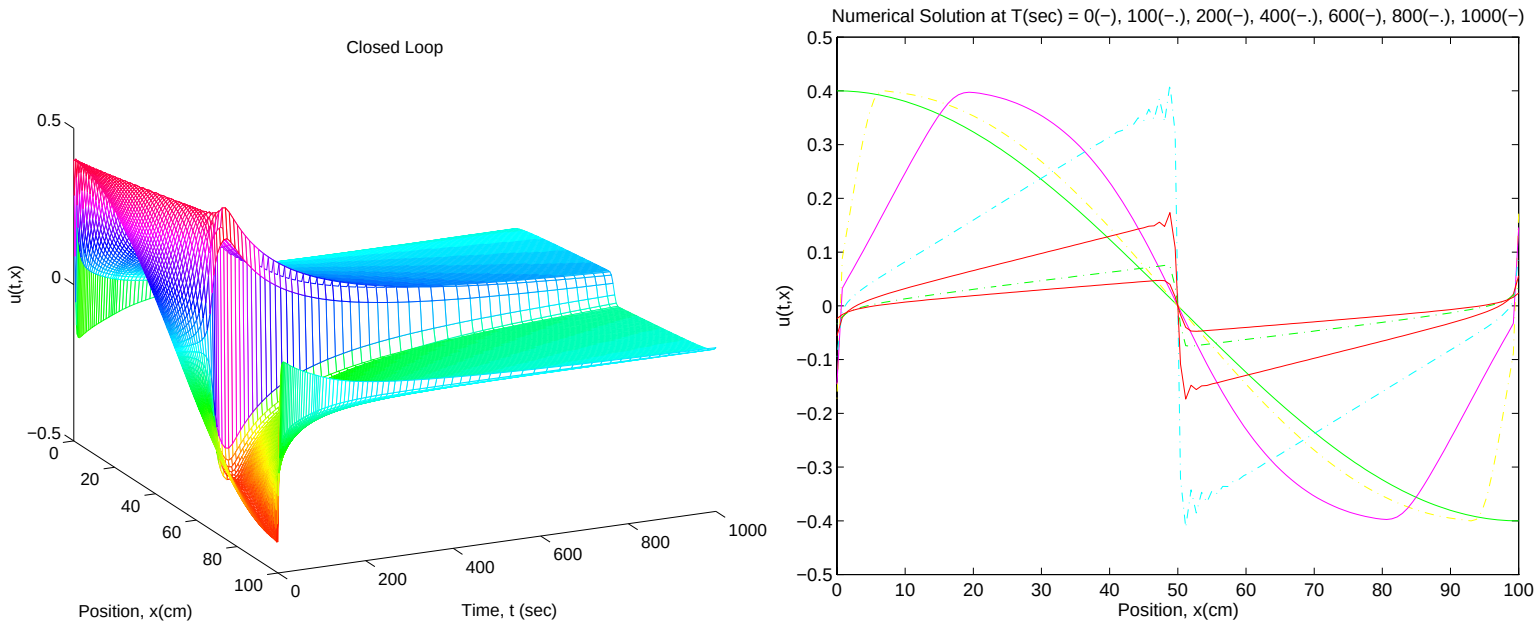


Figure 3.3.136: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 40, $K_{.939}$, N = 128

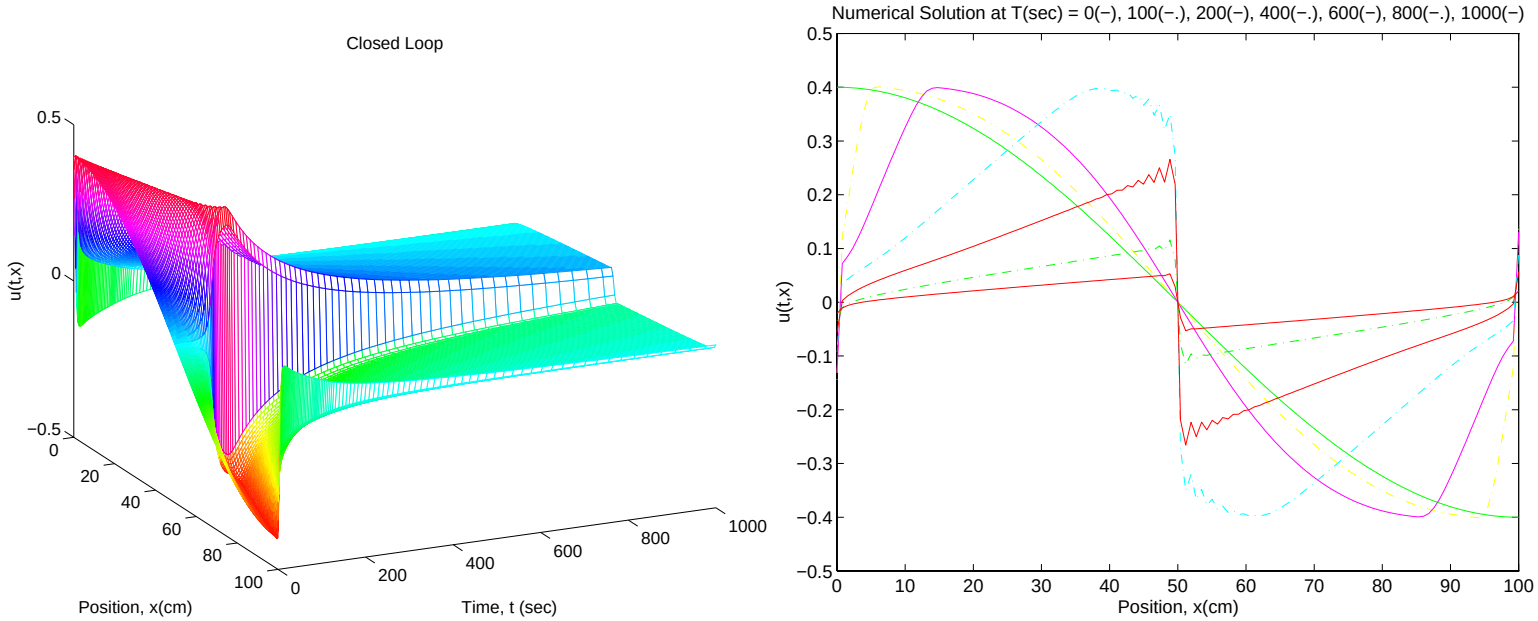


Figure 3.3.137: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 80, $K_{.939}$, N = 128

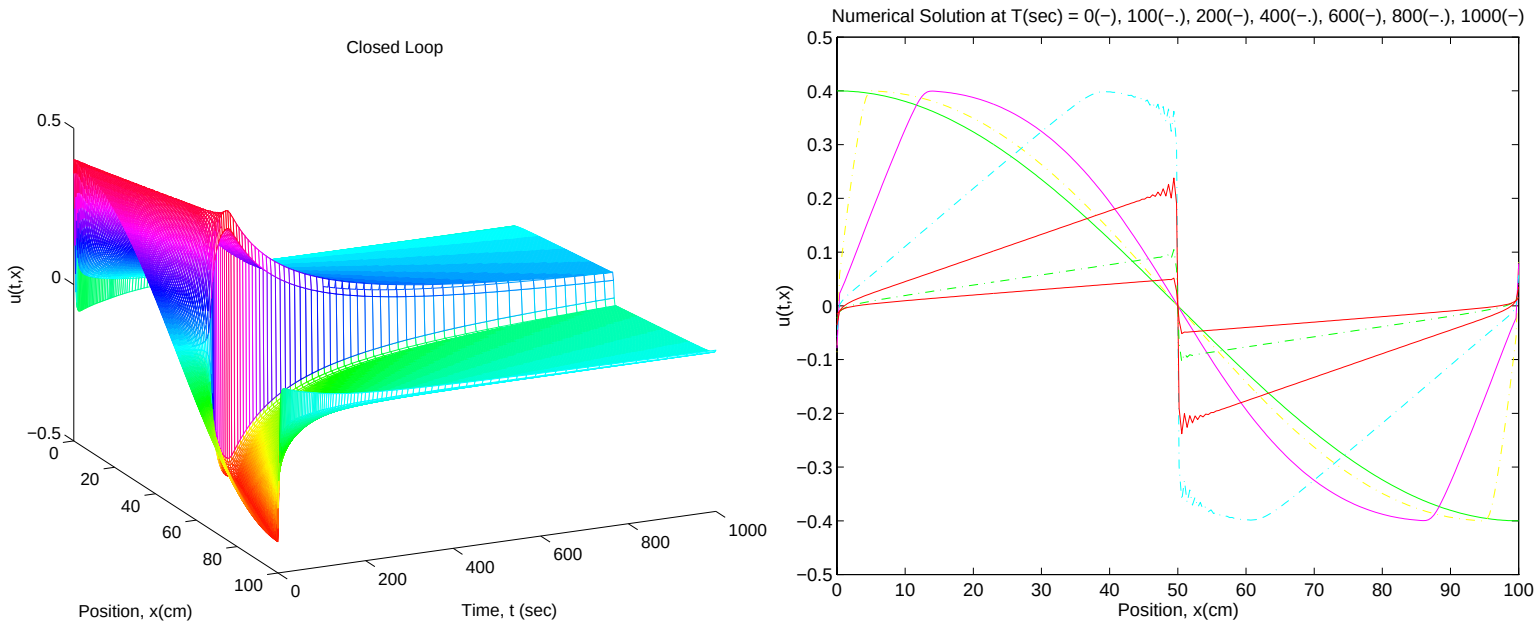


Figure 3.3.138: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 160, $K_{.939}$, N = 256

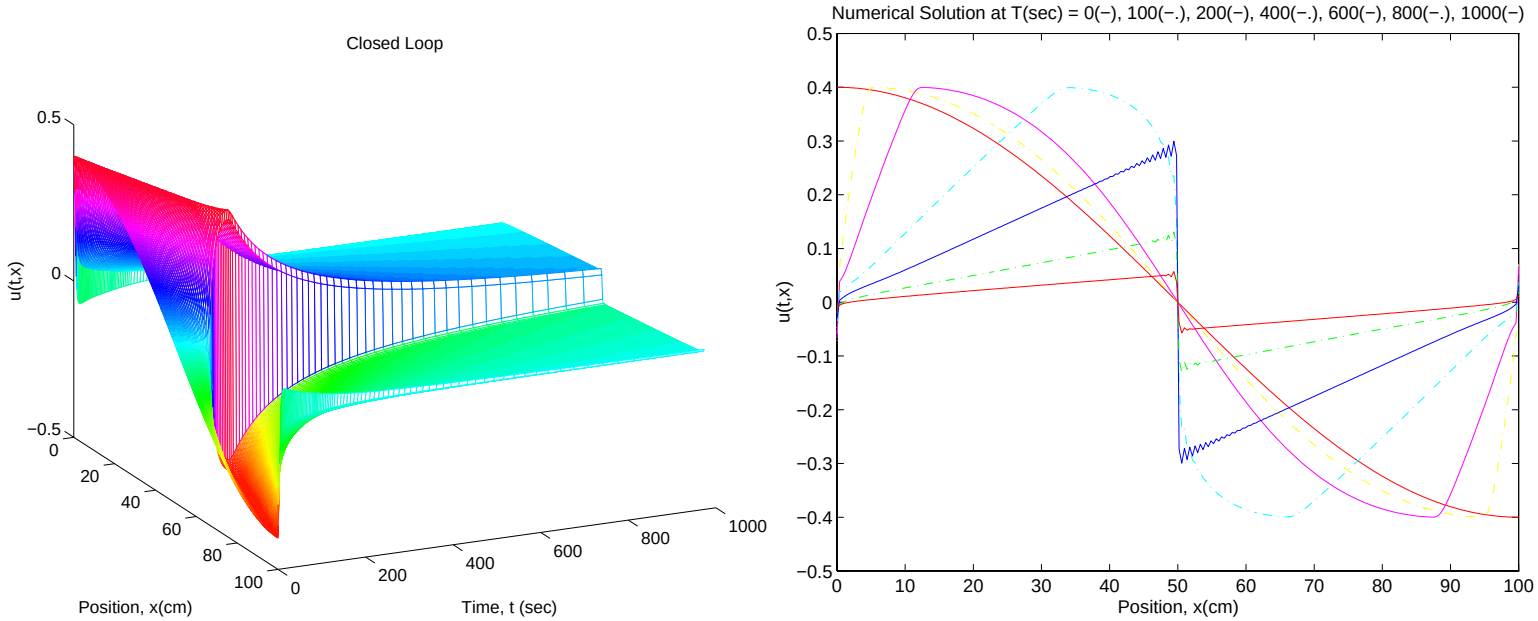


Figure 3.3.139: 1m Al rod, 25cm Fe films, $u_o(x) = .4\cos(\frac{\pi}{100}x)$, Re = 320, $K_{.939}$, N = 256

# Chapter 4

# Conclusions

## 4.1 Overview

In this thesis, the Galerkin/Conservation method is used to approximate solutions to the initial/boundary value problem (2.5). LQR control is used to find control laws which are applied to the homogeneous, non-linear Burgers' equation with Robin boundary conditions. Numerical solutions for open loop and closed loop systems are generated with various parameters and boundary conditions.

First, to test the MATLAB code, Example (3.2.1), from [16], is repeated. This example is used to confirm the convergence of the numerical solutions to the exact solution as the number of finite elements, N, is increased.

Next, open loop and closed loop systems for a fixed control function,

$$u_{1.14}(t) = -K_{1.14}\alpha(t),$$

and various initial conditions are simulated. The initial conditions consist of the functions $A_o \sin(\frac{\pi}{10}x)$, $A_o \cos(\frac{\pi}{10}x)$, and $A_o \sin(\frac{2\pi}{10}x)$ with various amplitudes, $A_o$. The closed loop numerical solutions decay toward the zero steady state function at a faster rate than the open loop numerical solutions, as predicted.

Then, for an aluminum rod with iron films, equivalent to Example (3.5.1) in [16], several control experiments are conducted. The weighting constant, $r$, from the cost function is varied from 5 to $5 \times 10^{-6}$ in order to test its effects on the optimal control. As $r$ decreased, the amplitudes of the functional gains increased at the boundaries. This procedure is conducted for iron films of lengths ten meters and twenty-five centimeters. In both cases, the decrease in the weighting constant results in an increase in the rate of decay of the numerical solutions. The largest change in this rate of decay occurs after $r = .0005$.

The rest of the simulations are run with $r$ fixed at .0005. To simulate going from approximate Neumann boundary conditions to approximate Dirichlet conditions, the lengths of the films are varied from ten meters to five centimeters. For each film length, an open loop system, a closed loop system with optimal control, and a closed loop system with the control $u_{1.14}(t)$ are generated. The amplitude of the functional gains for the optimal control at the boundaries increases as the film length decreases. The closed loop systems with the optimal control decay faster than the closed loop systems with $u_{1.14}(t)$, as expected. How-

ever, as the boundaries move toward Dirichlet conditions, the performance of the control $u_{1.14}(t)$ gets closer to the performance of the optimal control.

Further control experiments are conducted on open loop and closed loop systems with several Reynolds numbers. The quantities $\frac{\kappa_1}{L_1} = \frac{\kappa_2}{L_2}$ are fixed for two specific film lengths. First, the length of the iron films is set to ten meters. An open loop simulation, a closed loop simulation with optimal control, and a closed loop simulation with the control $u_{1.14}(t)$ are generated for Reynolds numbers of 10, 20, 40, 80, 160, and 320. At $Re = 80$, the open loop solutions begin to decay at a rate closer to that of the closed loop system. The performance of the control from the copper rod, $u_{1.14}(t)$, is significantly worse compared to the performance of optimal control until $Re = 80$. Then they perform very similarly at $Re = 160$ and $Re = 320$.

With the length of the iron films set to twenty-five centimeters, the same experiment is conducted. The open loop systems steepen toward a step function while the amplitudes at the boundaries remain unchanged. The closed loop solutions with the optimal control decay slightly faster than those with the control $u_{1.14}(t)$, until $Re = 160$. After which, the performance of each control function is nearly identical.

Finally, for both film lengths of ten meters and twenty-five centimeters, the gain matrix $K_{.939}$ is calculated. It is shown that $K_{.939}$ is nearly equal to the optimal gain matrix for all the various Reynolds numbers listed above. As a result, when the same closed loop experiments are repeated for the gain, $K_{.939}$, and compared with the numerical solutions

to the closed loop systems with optimal control, $K_{\frac{1}{Re}}$, they are nearly identical.

## 4.2 Conclusions

The numerical results of these control experiments are consistent with some of the results of Burns and Kang ([4], [5]). LQR control theory seems to control Burgers' equation with Robin boundary conditions for certain parameters.

The numerical solutions appear to be stable when the initial conditions are sufficiently small and the Reynolds number is sufficiently small. The rate of decay of the numerical solutions for closed loop systems decreases as the Reynolds number increases ($\epsilon$ decreases). This is also consistent with Burns and Kang, [5]. The rate of decay of the the numerical solutions for open loop systems is dependent upon $\epsilon$ as well.

The numerical solutions appear stable when the boundary conditions approximate Neumann and Dirichlet conditions. The performances of the optimal control function and another control function are nearly identical for approximate Dirichlet boundary conditions. For the approximate Neumann boundary conditions, the optimal control function performed significantly better than a non-optimal control function. However, as $Re$ increased their performances became increasingly similar.

## 4.3 Further Research

The numerical results of this thesis leave several areas of discussion left open. The following topics may be interesting to analyze and provide more insight into the stabilizing properties of Burgers' equation:

- Will the closed loop system for Burgers' equation with Robin boundary conditions reach a constant steady state in finite time?
- Are there limiting parameters, such as a Reynolds number or boundary conditions, for which the numerical solutions to the closed loop system become unstable?
- Why, for the same rod and films, are the gain matrices for various Reynolds numbers almost identical?

# Appendix A

# Symbols

| | |
|---|---|
| $A_\epsilon^N$ | open loop state matrix |
| $\mathcal{A}_\epsilon^N$ | closed loop state matrix |
| $B_\epsilon^N$ | control matrix |
| $D^N$ | matrix of integral calculations |
| $G^N$ | matrix of rod and film parameters |
| $\mathcal{F}_E^N$ | Galerkin non-linear term |
| $\mathcal{F}_D^N$ | Galerkin/Conservation non-linear term |
| $K^N$ | stiffness matrix |
| $K_\epsilon$ | $\epsilon$ - dependent gain matrix |
| $M^N$ | mass matrix |
| $c$ | specific heat |

| | |
|---|---|
| $\alpha_i(t)$ | time-dependent weight |
| $\epsilon$ | diffusivity constant |
| $f(t,x)$ | forcing function |
| $K_\epsilon$ | gain matrix |
| $h_i(x)$ | piecewise linear element |
| $\phi(x)$ | initial function |
| $\kappa$ | conductivity of the rod |
| $\kappa_1$ | conductivity of left end film |
| $\kappa_2$ | conductivity of right end film |
| $k_\epsilon^o$ | left end functional gain |
| $k_\epsilon^1$ | right end functional gain |
| $L_1$ | length of left end film |
| $L_2$ | length of right end film |
| $N$ | number of linear element functions |
| $Q$ | LQR weighting matrix |
| $q(t)$ | control vector |
| $R$ | LQR weighting matrix |
| $Re$ | Reynolds number |
| $r$ | weighting constant |
| $\rho$ | uniform density of the rod |

| | |
|---|---|
| $u(t,x)$ | solution to Burgers' equation |
| $u^N(t,x)$ | numerical approximation to the solution to Burgers' equation |
| $u_o(x)$ | initial condition |

# Bibliography


[1] Bail, Thomas R., " A Disturbance-Rejection Problem for a 2-D Airfoil ", Virginia Polytechnic Institute and State University, Master's Thesis, 1997.

[2] Berg, P.W., and McGregor, J.L., *Elementary Partial Differential Equations*, Holden-Day, Oakland, California, 1966.

[3] Burns, J.A., Balogh, A., Gilliam, D.S., and Shubov, V.I., " Numerical Stationary Solutions for a Viscous Burgers' Equation ", Journal of Mathematical Systems, Estimation, and Control, to appear.

[4] John A. Burns and Sungkwon Kang, *A Stabilization Problem for Burgers' Equation with Unbounded Control and Observation*, International Series of Numerical Mathematics, Vol. 100, 1991.

[5] John A. Burns and Sungkwon Kang, *A Control Problem for Burgers' Equation with Bounded Input/Output*, Nonlinear Dynamics 2: pp. 235-262, 1995.

[6] Byrnes, C.L., Gilliam, D.S., and Shubov, V.L., " Boundary Control, Feedback Stabilization, and the Existence of Attractors for a Viscous Burgers' Equation ", 1990.

[7] Christopher I. Byrnes and Anders Lindquist, *Theory and Applications of Nonlinear Control Systems*, North-Holland, 1986.

[8] Chandrasekharan, P.C., *Robust Control of Linear Dynamical Systems*, Academic Press Inc., San Diego, 1996.

[9] Djaferis, Theodore E., *Robust Control Design, A Polynomial Approach*, Kluwer Academic Publishers, Boston, 1995.

[10] Clive A. J. Fletcher, *Computational Galerkin Methods*, Springer-Verlag, New York, 1984.

[11] Clive A. J. Fletcher, *Burgers' Equation: A Model for all Reasons*, University of Syndey, North Holland Publishing Company, 1982.

[12] Marrekchi, H., " Dynamic Compensators for a Nonlinear Conservation Law ", Virginia Polytechnic Institute and State University, 1993.

[13] Olds, Shana D., " Modeling and LQR Control of a Two-Dimensional Airfoil ", pp. 33-34, Virginia Polytechnic Institute and State University, Master's Thesis, 1997.

[14] Steven M. Pugh, *Finite Element Approximations of Burgers' Equation*, Virginia Tech, Master's Thesis, 1995.

[15] Rubio, Diana, " Distributed Parameter Control of Thermal Fluids ", Virginia Polytechnic Institute and State University, Ph.D Dissertation, 1997.

[16] Smith III, Lyle, " Finite Element Approximations of Burgers Equation With Robin Boundary Condtions ", Virginia Polytechnic Institute and State University, Master's Thesis, 1997.

# Appendix B

# MATLAB Code

## RUNLQR.M

*Set up and run the LQR problem for an aluminum rod. This is the main body of the Finite Element Method for solving Burgers' Equation. The parameters of the problem are defined. The gain matrix is calculated.*

```
clear all
EN = 256 % The number of elements for the problem
Lrod = 100;  % length of the rod in cm

[unifh,Index,Node,XQSpt,Xpt,h,nlocal] = geometry(EN,Lrod);

% Setting up the Constants for the problem

L1=25;  % length of film 1 in cm
L2=25;  % length of film 2 in cm
kap1=.124;  % conductivity of film 1 in cal/(s*cm*C)
kap2=.124;  % conductivity of film 2 in cal/(s*cm*C)
kappa=.55;   % conductivity of the rod in cal/(s*cm*C)
epsilon=kappa/(2.7*.217);  % epsilon=kappa/(rho*c)
%epsilon=1/320
unifh
```

```
% Set up the matrices.

[MM,KK,A,B,F,G,M,v0] =
matrices(EN,Index,Node,XQSpt,Xpt,h,nlocal,L1,L2,kap1,kap2,kappa,epsilon,Lrod);

Q = MM;
R = .0005*eye(2);

[burgk,burgs,burge] = lqr(A,B,Q,R); % calculate the gain

save gaincl burgk;

Acl = A - B*burgk; % calculate the closed loop matrix

load gainKal256.mat   % load the necessary gain
load gaincu256.mat
Kal = Kal256;

Arey = A - B*Kal;
% Arey = A - B*Kcu;
tf = 3000;
save sysmat EN Lrod A B Acl Arey M MM Q R v0 Xpt
save gain Kal

clear
-------------------------------------------------------
-------------------------------------------------------
```

# GEOMETRY.M

*This script sets up the necessary arrays that allow the main program compute results for a specific problem. Contained herein is the correspondence between the actual layout of the problem and the layout of the functions used in the main program.*

```
function[unifh,Index,Node,XQSpt,Xpt,h,nlocal] = geometry(EN,Lrod)
% endpoints
a=0;
```

```
b=Lrod;

% step-size if uniform
unifh=(1/(EN+1))*(b-a);
numint=(b-a)/unifh;
h=0;
for i=1:numint, h(i)=unifh;, end

% Total number of nodes
numnode=max(size(h))+1;

% Set up the x-coordinate of each node
Xpt=zeros(numnode,1);
Xpt(1) = a;
for i=2:numnode, Xpt(i)=Xpt(i-1)+h(i-1);, end

% Set up Index, which describes the number of unknowns
% at each global unknown.

Index=zeros(numnode,1);
% Describe inhomogeneous boundary conditions
%Index(numnode)= -1;
%Index(1)= -1;
blue = 0;
for i=1:numnode
    if Index(i)==0, Index(i)=i-blue;,end
end

% Set up the quadrature points in each element.
% We're using one-point Guass quadrature, so we want
% the midpoint of each interval.

XQpt=0;
for i=1:max(size(h)), XQpt(i,1)=Xpt(i)+h(i)/2;, end
% nquad=1;
% XQpt

% Set up the 2-pt. quadrature points.
XQSpt=0;
for i=1:max(size(XQpt)),
```

```
    XQSpt(i,1)=XQpt(i) - .5773502692*h(i)/2;
    XQSpt(i,2)=XQpt(i) + .5773502692*h(i)/2;
end

% Set up the 3-pt. quadrature points.

XQ3pt=0;
for i=1:max(size(XQpt)),
    XQ3pt(i,1)=XQpt(i) - .7745966692*h(i)/2;
    XQ3pt(i,2)=XQpt(i);
    XQ3pt(i,3)=XQpt(i) + .7745966692*h(i)/2;
end

% Set up the relation between the local node for the
% basis functions and the global nodes.

Node=zeros(max(size(h)),2);
for i=1:max(size(h)), Node(i,1)=i;,end
for i=1:max(size(h)), Node(i,2)=i+1;, end
nlocal=2;
%-------------------------------------------
-------------------------------------------
```

# MATRICES.M

*This is the subroutine where the mass and stiffness matrices are assembled. It also sets up the initial condition and right hand side matrices.*

```
% Set up the MM and KK matrices.
% Build F(v), the inhomogeneous term matrix.

function[MM, KK, A, B, F, G, M, v0] =
matrices(EN, Index, Node, XQSpt, Xpt, h, nlocal, L1, L2,...
 kap1, kap2, kappa, epsilon, Lrod)
numunk=0;
for i=1:max(size(Index))
if Index(i)>0,numunk=numunk+1;,end
end
MM=zeros(numunk,numunk);
```

```
KK=zeros(numunk,numunk);
FF=zeros(numunk,2);
V=zeros(numunk,1);

for nel=1:max(size(h))
  for quad=1:2
 quad=1;
      for j=1:nlocal
if Index(Node(nel,j))>0
  % integrate basis functions
  x=XQSpt(nel,quad);
  nnn=Index(Node(nel,j));
  KK(nnn,nnn)=KK(nnn,nnn)-(basisd(nnn-1,x,Xpt,h)^2)*h(nel)/2;
  MM(nnn,nnn)=MM(nnn,nnn)+(basis(nnn-1,x,Xpt,h)^2)*h(nel)/2;
  if nnn<max(size(MM))
     KK(nnn,nnn+1)= ...
      KK(nnn,nnn+1)-basisd(nnn-1,x,Xpt,h)*basisd(nnn,x,Xpt,h)*h(nel)/2;
    KK(nnn+1,nnn)=KK(nnn,nnn+1);
    MM(nnn,nnn+1)= ...
      MM(nnn,nnn+1)+basis(nnn-1,x,Xpt,h)*basis(nnn,x,Xpt,h)*h(nel)/2;
    MM(nnn+1,nnn)=MM(nnn,nnn+1);
  end

% Integrate basis functions against f(x)

 FF(nnn,1)=FF(nnn,1)+(beta*(quadA*x*x+quadB*x+quadC-(sin(pi*x))^2)- ...
% epsilon*(2*quadA-2*pi*pi*((cos(pi*x))^2 -(sin(pi*x))^2)))...
                  *basis(nnn-1,x,Xpt,h)*h(nel)/2;

% FF(nnn,2)=FF(nnn,2)+(quadA*x*x+quadB*x+quadC-(sin(pi*x))^2)...
%*(2*quadA*x+quadB-2*pi*sin(pi*x)*cos(pi*x))*basis(nnn-1,x,Xpt,h)*h(nel)/2;

% Homogeneous Burgers'
    FF(nnn,1)=FF(nnn,1)+(0)*basis(nnn-1,x,Xpt,h)*h(nel)/2;
    FF(nnn,2)=FF(nnn,2)+(0)*basis(nnn-1,x,Xpt,h)*h(nel)/2;

%  integrate basis functions against the initial function

b = Lrod;
```

```
% V(nnn,1)=V(nnn,1)+(quadA*x*x+quadB*x+quadC-(sin((pi/b)*x))^2)*
       basis(nnn-1,x,Xpt,h)*h(nel)/2;
% V(nnn,1)=V(nnn,1)+10*sin((pi/b)*x)*basis(nnn-1,x,Xpt,h)*h(nel)/2;
% V(nnn,1)=V(nnn,1)+.4*cos((pi/b)*x)*basis(nnn-1,x,Xpt,h)*h(nel)/2;
% V(nnn,1)=V(nnn,1)+10*sin(2*(pi/b)*x)*basis(nnn-1,x,Xpt,h)*h(nel)/2;
% V(nnn,1)=V(nnn,1)+.5*(100*x*x*(1-x)*(1-x))*basis(nnn-1,x,Xpt,h)*h(nel)/2;

end
      end
    end
end

% Correct Stiffness Matrix
KK(1,1)=-1/h(1) - kap1/(kappa*L1);
KK(numunk,numunk) = -1/h(numunk-1) - kap2/(kappa*L2);

G=zeros(numunk,2);
G(1,1)=kap1/(kappa*L1);
G(numunk,2)=kap2/(kappa*L2);

M=inv(MM);

% Set Up open loop matrices
A=epsilon*(inv(MM))*KK;
B=epsilon*(inv(MM))*G;

% Forcing function for inhomogeneous Burgers' equation
F=(inv(MM))*FF;

% Set up the initial condition
v0 = zeros(numunk,1);
v0=M*V;
---------------------------------------------
---------------------------------------------
```

# RUNODE.M, OPRUNODE.M, REYRUN.M

*This function sends the time interval, initial condition, the number of time steps, and the number of unknowns to the differential equation solver. The time step and solution arrays for the closed loop*

*problem are returned. oprunode.m and reyrun.m are analogous except the appropriate open loop and closed loop matrice must be sent to the differential equation solver.*

```
load sysmat.mat
tf = 1000;
time3 = cputime;


[Tcl,vcl] = simult(0,tf,Acl,v0,M);

save solutncl Lrod Xpt Tcl vcl tf EN

time4=cputime;
odetime = time4 - time3
clear
---------------------------------------------
---------------------------------------------
```

# SIMULT.M, OPSIMULT.M

*This function forms the right hand side for the ODE solver by calling the appropriate right hand side function. (opsimult.m calls oprhs.m)*

```
function [Ttemp,vtemp] = simult(tini, tfinal, Am, vini, Mtemp)
Aclrhs = Am;
save clrhsmat Aclrhs Mtemp
[Ttemp,vtemp] = ode45('clrhs', tini, tfinal, vini);
---------------------------------------------
---------------------------------------------
```

# CLRHS.M, OPRHS.M

*Solves the closed-loop (controlled) right hand side, y = Acl*x + F.*
*oprhs.m solves the open loop right hand side, y = Ax + F.*

```
function vdot = clrhs(t,y)
load clrhsmat.mat
```

```
y = y(:);
ll = length(y);

% Construct the Galerkin Conservation nonlinear term (vector form)
p = y.*y;
J = [p(2:ll); p(ll)] - [p(1); p(1:ll-1)];

% Calculating vdot with the given right hand side
vdot = Aclrhs*y - .25*(Mtemp*J);
---------------------------------------------
---------------------------------------------
```

# PLOTCL.M, PLOTOP.M

*This function plots the solution to the Closed Loop system in 3D and plots the solution against the x-coordinates at various times. plotop.m plots the appropriate open loop solutions.*

```
load solutncl.mat % Load the solution file
T = Tcl;
v = vcl;
b = Lrod;
h = .4;
%tf = 10;

step = ceil(max(size(T))/100); % Set up time steps
count=0;
xxx=zeros(5*EN+6,1);
vplot=zeros(ceil(max(size(T))/step),EN+2);
Tplot=zeros(ceil(max(size(T))/step),1);
for i = 1:step:max(size(T))
   count=count+1;
   count2=0;
   vplot(count,:)=v(i,:);   % Pick plotting points
  Tplot(count,1)=T(i);
end

subplot(2,1,1) % Plot the 3-D solution
mesh(Xpt, Tplot, (vplot))
view(65,25)
```

```
axis([0,b,0,tf,-h,h])
   xlabel('Position, x(cm)')
   ylabel('Time, t (sec)')
   zlabel('u(t,x)')
  title('Closed Loop')

subplot(2,1,2)
plot(Xpt,vplot(1,:),'-',Xpt,vplot(d/10,:),'-.' ,Xpt,vplot(d/5,:),'-', ...
Xpt,vplot(2*d/5,:), '-.' ,Xpt,vplot(3*d/5,:),'-' , ...
Xpt,vplot(4*d/5,:),'-.', Xpt,vplot(d,:),'-')
axis([0,b,-h,h])
   xlabel('Position, x(cm)')
   ylabel('u(t,x)')
   zlabel('u(t,x)')
   title(['Numerical Solution at T(sec) = ',num2str(0),'(-), ' ...
,num2str(g*d/10), '(-.), ' num2str(g*d/5),'(-), ', ...
 num2str(g*2*d/5),'(-.), ', num2str(g*3*d/5),'(-), ', ...
 num2str(g*4*d/5),'(-.), ', num2str(g*d),'(-)'])

norm(vplot(d,:)',inf) % Find the amplitude at tf
---------------------------------------------
---------------------------------------------
```

# FGAIN.M

*This script file calculates the functional gains for both controllers, given the parameters of the problem.*

```
clear all

Lrod = 100;
ts=cputime;
Xfpt=zeros(130,6);
count=0;
fgain1 = zeros(130,6);
fgain2 = zeros(130,6);
fgain3 = zeros(130,6);
for l = 2:7
EN = 2^l;
```

```
[unifh,Index,Node,XQSpt,Xfpt,Xpt,h,nlocal,count] = ...
geom(EN,count,Xfpt,Lrod);

L1=1000;  % length of film 1 in cm
L2=1000;  % length of film 2 in cm
kap1=.124;  % conductivity of film 1 in cal/(s*cm*C)
kap2=.124;  % conductivity of film 2 in cal/(s*cm*C)
kappa=.55;   % conductivity of the rod in cal/(s*cm*C)
%epsilon = kappa/(2.7*.217);  % epsilon=kappa/(rho*c)
epsilon = 1/320;

[MM,KK,A,B,F,G,M,v0] = ...
matrices(EN,Index,Node,XQSpt,Xpt,h,nlocal,L1,L2,kap1,kap2,kappa,epsilon,Lrod);
Q = MM;
R = .0005*eye(2);

[burgk,burgs,burge] = lqr(A,B,Q,R);

for j = 1 : EN+2
fgain1(j,l-1) = M(j,:)*((burgk(1,:))');
fgain2(j,l-1) = M(j,:)*((burgk(2,:))');
end
end

te=cputime;
fgtime = te - ts
----------------------------------------------
----------------------------------------------
```

# PLTGAIN.M

*This file plots the functional gains for the left end and right end controllers.*

```
height = 10;
subplot(2,1,1)
plot(Xfpt(1:10,2),fgain1(1:10,2),'o',Xfpt(1:18,3),fgain1(1:18,3),'.', ...
        Xfpt(1:34,4),fgain1(1:34,4),'-.',Xfpt(1:66,5),fgain1(1:66,5), '--', ...
        Xfpt(1:130,6),fgain1(1:130,6))
xlabel('Position, x')
```

```
ylabel('functional gain')
title('qo functional gain for N = 8(o), 16(.), 32(-.), 64(--), 128(-)')
axis([0,Lrod,0,height])

subplot(2,1,2)
plot(Xfpt(1:10,2),fgain2(1:10,2),'o',Xfpt(1:18,3),fgain2(1:18,3),'.', ...
        Xfpt(1:34,4),fgain2(1:34,4),'-.',Xfpt(1:66,5),fgain2(1:66,5), '--', ...
        Xfpt(1:130,6),fgain2(1:130,6))
xlabel('Position, x')
ylabel('functional gain')
title('q1 functional gain for N = 8(o), 16(.), 32(-.), 64(--), 128(-)')
axis([0,Lrod,0,height])
---------------------------------------------
---------------------------------------------
```

# VITA

Kenneth Lawrence Massa was born in Summit, New Jersey on May 13, 1973. He graduated from Toms River East High School in 1991. He attended Virginia Polytechnic Institute and State University from 1991 to 1997, where he received Bachelor of Science degrees in Applied Computational Mathematics and Physics, and a Master of Science degree in Applied Computational Mathematics. He is a member of Pi Mu Epsilon National Mathematics Honorary Society. In September, 1997, Ken began work as an engineer for Lockheed Martin, Government Electronic Systems in Moorestown, New Jersey.