Eleventh Industrial Mathematical and Statistical Modeling Workshop for Graduate Students

July 25-August 2, 2005
North Carolina State University
Raleigh, NC

Edited by Robert T. Buche, Mansoor A. Haider, Mette S. Olufsen, Ralph C. Smith and Hien Tran

Participants

Students

1. Affane Aji, Chadia, Auburn University
2. Bandulasiri, Ananda, Texas Tech University
3. Bartlett, Andy, College of Charleston
4. Benim, Robert, University of Portland
5. Berger, Andre, Emory University
6. Bowman, Mike, Montana State University
7. Braun, Thomas, NCSU
8. Damu, Seshadri, University of Tennessee
9. David, John, NCSU
10. Deems, Thomas, Slippery Rock University
11. Diawara, Norou, Auburn University
12. Froehlich, Mihaela, Duke University
13. Geis, Jennifer, Augsburg College
14. Gonzales, Chad, Arizona State University
15. Gonzalez, John, Northeastern University
16. Gruber, Mary, Michigan State University
17. Gunnarsdottir, Ingunn, Roskilde University
18. Han, Xiaoying, University of Buffalo
19. Hanlon, Bret, Texas Tech University
20. Haugh, Janine, NCSU
21. Hazaveh, Kamyar, Georgia Tech
22. Jian, Bing, University of Florida
23. Joyner, Sarah Lynn, Meredith College
24. Kose, Emek, Drexel University
25. Li, Zheng, Brown University
26. Lin, Chuan, NCSU
27. Lipkin, Craig, Drexel University
28. Mavinga, Nsoki, University of Alabama at Birmingham
29. Modlin, Danny, UNC Wilmington
30. Navaratna, Menaka, Texas Tech University
31. Osorio, Eduardo, Rutgers
32. Park, Min-Hyong, Cornell University
33. Perez, Elizabeth, University of South Carolina
34. Prabhakaran, Jyothsna, University of Alabama at Birmingham
35. Rupani, Sidharth, Worcester Polytechnic Institute
36. Saglanmak, Nesli, Roskilde University
37. Stepp, April, West Carolina University
38. Thukani, Thobile, Southern University and A&M College
39. Tingey, Troy, Arizona State University
40. Tuncer, Necibe, Auburn University
41. Tweedy, Eamonn, NCSU
42. Vijayat, Amith, Clarkson University

43. Vogl, Chris, Illinois Wesleyan University
44. Wan, Xiaohai, NCSU
45. Wheeler, Erik, Montana State University
46. Wong, Anastasia, Mills College
47. Xia, Xiaofeng, Emory University
48. Zhang, Wei, Tufts University
49. Zhou, Guohua, Clarkson University

Problem Presenters

1. Angelides, Greg, MIT Lincoln Laboratory
2. Allain, Eric J. and Coward-Kelly, Guillermo, Novozymes North America, Inc.
3. Howard, Anna, Lord Corporation
4. Georgieva, Anna, Novartis, Inc.
5. Ranjan, Priya, Intelligent Automation, Inc.
6. Ott, William, PAR Technologies LLC
7. Settlage, Mike and Snider, Glen, Progress Energy

Faculty Consultants

1. Buche, Robert
2. Chertock, Alina
3. Langville, Amy
4. Li, Zhilin
5. Lubkin, Sharon
6. Oates, William
7. Olufsen, Mette
8. Pang, Tao
9. Pelletier, Denis
10. Smith, Charlie
11. Smith, Ralph
12. Tran, Hien
13. Zenkov, Dmitry
14. Zhang, Jim

Preface

This volume contains the proceedings of the *Industrial Mathematical and Statistical Modeling Workshop for Graduate Students* that was held at the Center for Research in Scientific Computation at North Carolina State University (NCSU) in Raleigh, North Carolina from July 25th to August 2nd, 2005. This workshop was the eleventh to be held at NCSU and brought together 41 graduate students from a large number of graduate programs, as well as 8 undergraduate students from the 2005 NCSU mathematics REU (Research Experience for Undergraduates) summer program. Graduate students represented a large number of graduate programs including University of Alabama at Birmingham, Arizona State University, Auburn University, Brown University, University of Buffalo, College of Charleston, Clarkson University, Cornell University, Drexel University, Duke University, Emory University, University of Florida, Georgia Tech, Michigan State University, Montana State University, North Carolina State University, Northeastern University, Roskilde University, Rutgers, University of South Carolina, Southern University and A&M College, University of Tennessee, Texas Tech University, Tufts University, West Carolina University, UNC Wilmington and Worcester Polytechnic Institute.

Students were divided into seven teams to work on mathematical and statistical modeling problems presented, on the morning of the first day, by industrial scientists. In contrast to neat, well-posed academic exercises that are typically found in coursework or textbooks, the workshop problems were challenging real-world problems from industry that required the varied expertise and fresh insights of the group for their formulation, solution and interpretation. Each group spent the first eight days of the workshop investigating their project and then reported their findings in half-hour public seminars on the final day of the workshop.

The following is a list of the industrial presenters and the problems that they brought to the workshop:

- **Greg Angelides** (MIT Lincoln Laboratory) *Optimal Mission Planning*
- **Eric J. Allain and Guillermo Coward-Kelly** (Novozymes North America, Inc.) *Mathematical Modeling of Enzyme Action in Fuel Alcohol Production*
- **Anna Howard** (Lord Corporation) *Control of Dynamic System Response with Directional Forcing Sources*
- **Anna Georgieva** (Novartis, Inc.), *Mathematical Models of Signal Transduction: Model Reduction*
- **Priya Ranjan** (Intelligent Automation, Inc.) *Power Preallocation and Control in a Heavy Traffic Queueing Model for Wireless Systems*
- **William Ott** (PAR Technologies LLC) *Modeling of a Fluid Pump Driven by a Piezoceramic Disk Actuator*
- **Mike Settlage and Glen Snider** (Progress Energy) *Modeling and Forecasting of Electric Power Load*

These problems represent a broad spectrum of mathematical and statistical topics and applications. While the problems included many aspects that are challenging to address in the nine-day duration of the workshop, the reader will observe remarkable progress on all the projects.

As the organizers, we strongly believe that this type of modeling workshop provides a highly valuable non-academic research related experience for graduate students that, simultaneously, contributes to the research efforts of the industrial participants. In addition, the nature of this activity facilitates the development of graduate students' ability to communicate and interact with scientists who are not traditional mathematicians, but require and employ mathematical tools in their work. Through this unique experience that exposes students to the means by which Mathematics is applied outside Academia, the workshop has helped many students formalize their career aspirations. In several cases, in past workshops, this help has been in the form of direct hiring by the participating companies. By broadening the horizon beyond that which is typically presented in mathematics graduate education, students interested in academic careers also find a renewed sense of excitement about applied mathematics and statistics.

Active participation of all involved during a nine day period of almost uninterrupted work, complemented by a series of social activities, greatly enhanced the success of the workshop. The organizers are most grateful to the participants for their dedicated effort and contributions. Funding for the workshop was provided by the Statistical and Applied Mathematical Sciences Institute (SAMSI) with additional financial support, personnel, and facilities provided by the Center for Research in Scientific Computation (CRSC) and the Department of Mathematics at North Carolina State University. The eight REU students who participated in the workshop were also partially support by the National Security Agency. Finally, we would like to express our sincere gratitude to Rory Loycano, Lesa Denning, Brenda Currin and Amy Langville for their efforts and help in all administrative matters. We are also grateful to Brandy Benedict, Ryan Siskind and Corey Winton for their assistance in providing transportation that helped make the workshop a great success.

Robert T. Buche, Mansoor A. Haider, Mette S. Olufsen, Ralph C. Smith and Hien Tran
Raleigh, 2005

Contents

Chapter 1

Optimal Mission Planning

PROBLEM 1: OPTIMAL MISSION PLANNING

Andy Bartlett[1], André Berger[2], Craig Lipkin[3], Nsoki Mavinga[4], Elizabeth Perez[5], Eamonn Tweedy[6], Erik Wheeler[7]

Problem Presenter:
Greg Angelides
MIT Lincoln Laboratory

Faculty Consultants:
Dr. Amy Langville
College of Charleston
&
Dr. Zhilin Li
North Carolina State University

Abstract

Analytic and discrete optimization approaches for routing an aircraft in hostile air spaces have been developed. Using these approaches, an aircraft's optimal risk path (where the risk is being detected by enemy radars) with constraints on aircraft speed, altitude and turn capability, mission range, mission time, fuel, etc. is computed. In the discrete optimization approach, we use different methods to find the optimal path. We reformulate the problem as a constrained shortest path problem (CSPP) on a grid graph and use linear and dynamic programming to solve the CSPP. A stochastic model uses a random walk on a similar grid to find the aircraft's optimum path. The analytical approach based on the calculus of variations reduces the original optimization problem to a system of nonlinear differential equations.

1.1 Introduction

In this report we consider the problem of routing an aircraft through hostile air space. The objective is to find a path through a region avoiding enemy radars as much as possible, flying over a given target and obeying several constraints specified by the aircraft. We will first describe the assumptions and the statement of the problem.

When trying to formulate a mathematical model to describe the problem we make the following assumptions:

1. We consider a polygonally bounded region $\mathcal{R} \subset \mathbb{R}^2$.

2. The aircraft is flying with a constant speed.

[1]College of Charleston
[2]Emory University
[3]Drexel University
[4]University of Alabama at Birmingham
[5]University of South Carolina
[6]North Carolina State University
[7]Montana State University

3. The aircraft is flying at a constant altitude.

4. There is a finite number of polygonally bounded no-fly zones.

5. There is a finite number of radars described by circles.

6. Radar detection does not depend on the relative position nor on the angle of the path to the center of the radar.

7. The risk $r(x)$ of being detected at any location $x \in \mathcal{R}$ is equal to the number of radars containing that location.

8. The total risk of any path P is then the line integral $r(P) = \int_P r(x)dx$.

Note that, here we only discuss the ideas in two space dimensions. But we believe that they can be generalized to three dimensions with some effort. The computational components may require substantially more time.

We will discuss the possibility of relaxing some of these assumptions in the description of the models and in the conclusions. For example, risk may be defined differently in each model. The problem of finding an optimal path through the region can now be described as follows.

The input of the problem is a compact region $\mathcal{R} \subset \mathbb{R}^2$, whose boundary $\partial\mathcal{R}$ is a polygon, a target $t \in \mathcal{R}\backslash\partial\mathcal{R}$, a finite number of radars and no-fly zones inside the region $\mathcal{R}$, a total distance limit D and a minimum turn radius R_{min}. The objective is to find a path P through the region $\mathcal{R}$, which enters and leaves the boundary of $\mathcal{R}$, such that $t \in P$, the maximum curvature of P is at most $\frac{1}{R_0}$, the total (Euclidean) length of P is at most D, and such that P does not intersect any of the no-fly zones, minimizing the total risk $r(P)$ of that path. We do not specify a particular start point or endpoint of the path. This problem will be dealt with in the particular models.

Similar problems arise in many different areas. One example is moving a robot through a weighted-region terrain (e.g. in [13]). The weights of the region could be determined by a specific difficulty for the robot to move through a particular part of the region, e.g. sand, snow, ice, etc. Another similar problem is Quality of Service (QoS) routing in data networks [8, 3], in which one wants to find a cheapest path through a data network, e.g. IP networks or optical networks, which also satisfies additional constraints as a total delay time bounded by an upper limit and possibly others. Moreover, routing an aircraft through enemy radars has been considered before by Zabarankin et al. [19]. However, they do not consider the constraints of flying over a target and obeying a minimum turn radius.

The paper will be organized as follows. We first describe a discrete model (Section 1.2) and then a stochastic model (Section 1.3), which both use a grid on the region to discretize the space. In Section 1.4 we develop a continuous model of the problem and use the calculus of variations to solve it. We give a summary and conclusions in Section 1.5.

1.2 Discrete Model

1.2.1 The General Constrained Shortest Path Problem

Many algorithms that try to solve some kind of weighted region problem as described in the introduction use a discretization of the space. Doing so often turns the problem into an instance of the constrained shortest path problem (CSPP) in a weighted graph, or into a related graph optimization problem.

In this section we will first define the CSPP and then we will briefly describe some of the algorithms and heuristics that have been used to solve the CSPP.

Secondly, we will turn the optimal mission planning problem into a constrained shortest path problem on a grid graph defined on the region of interest. In the following two sections we will then discuss in more detail the below mentioned dynamic programming methods and the linear programming formulation of the problem and describe how they can be modified to fit our specific constraints given by the application in mission planning. Moreover, we will discuss the flexibility of the approaches in terms of being able to handle natural extensions and also their limitations.

The *constrained shortest path problem* in a (directed or undirected) graph $G = (V, E)$ with a cost function c and a distance function d on the edges of G is to find a minimum cost path (w.r.t. to c) from a given source $s \in V$ to one or all other vertices in V, which at the same time has a total distance (w.r.t. to d) of at most some specified limit D. We assume the functions c and d both to be nonnegative. Unlike the minimum cost path problem from one source to all destinations, the CSPP is NP-hard. However, due to its wide range of applications many algorithms and heuristics for the CSPP have been suggested in the literature, which either find an optimal solution, but will not run in polynomial time on every instance or which do run in a (for the particular problem) "reasonable" time but can only guarantee an approximate solution.

There are different approaches to solve the CSPP. Among them are dynamic programming [8, 11], label setting algorithms [5], Lagrangian relaxation [2, 9] and approximation schemes based on cost and weight scaling [10]. We refer to [6] for a survey on these methods. Dumitrescu and Boland [7] also compared these methods computationally and found that a label setting method "... strongly outperforms other methods in terms of both time and memory." Moreover, they show that simple pre-processing techniques can be very effective when solving the CSPP.

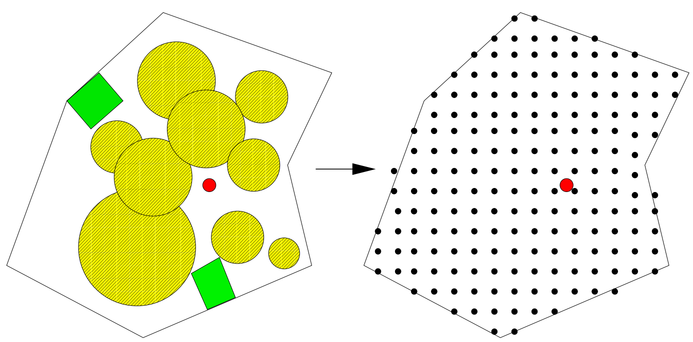


Figure 1.1: Turning the region into a grid.

We now turn to the problem of modeling the optimal mission planning problem as an instance of the CSPP. We discretize the region by imposing a grid on the plane, where the distance between the grid points is equal to the minimum turn radius of the aircraft (Figure 1.1). We construct a graph $G = (V, E)$, whose vertex set V consists of all grid points in the region. Each vertex (except vertices on the boundary) is adjacent with its eight neighboring vertices. Any feasible path can only make those moves as described in Figure 1.2. Since the distance between two horizontally or vertically neighboring grid points is chosen to be equal to the minimum turn radius, a $\pi/2$ turn (Figure 1.2 (B)) can actually be made by the aircraft, whereas the moves described in Figure 1.2 (C) are forbidden. The cost c_e of any edge $e \in E$ is equal to the number of radars containing it. If an edge is covered only partly by a radar only the particular fraction covered contributes to the cost of that edge. The distance d_e of an edge is equal to the Euclidean distance between its endpoints. For simplicity we assume that the target is a grid point, and that there are sufficiently many grid points on the boundary of the region serving as possible entry and exit points. The optimization problem now is to find a shortest path in the graph G (w.r.t. to the cost function c) whose total length (w.r.t. to d) is at most D. The constraints on the minimum turn radius, flying over the target and entering and exiting the boundary of the region will be dealt with differently in both approaches and we therefore refer to the following two sections for more details.

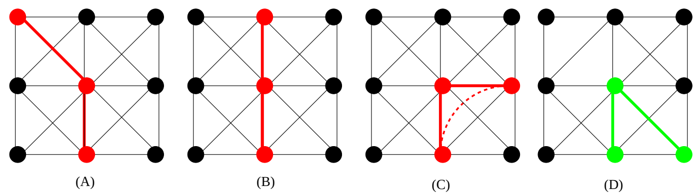


Figure 1.2: Possible [(A),(B)] and impossible [(C)] movements on the grid.

1.2.2 A Dynamic Programming Approach

In this section we will show how to modify Joksch's algorithm [11] to fit our framework. We start with describing Joksch's algorithm for finding a minimum cost path (w.r.t. to the cost function c) under the constraint that its total length is at most D from a source s to all other vertices in a graph $G = (V, E)$. The main idea comes from Dijkstra's classical shortest path algorithm, i.e. that a shortest path from s to t contains shortest paths from s to any vertex on that path.

For any vertex $v \in V$ and any integer i ($0 \leq i \leq D$), the variable $w[v, i]$ stores the cost of the cheapest path from s to v having total distance at most i and the variable $l[v, i]$ stores the last edge entering v on that path. Initially, $w[v, i] = \infty$ for all $v \neq s$ and $w[s, i] = 0$ for all $0 \leq i \leq D$. We then have the following recursion

$$w[v, i] = \min\{w[v, i-1], w[u, i - d_{uv}] + c_{uv} : uv \in E, d_{uv} \leq i\}, \tag{1.1}$$

where we then set $l[v, i]$ according to which number achieves the minimum above.

Assuming that the distances d_e are integers for every $e \in E$, Joksch's algorithm will find an optimum solution in time $O((m + n \log n)D)$, where n and m are the number of vertices and edges, respectively, of the graph. This algorithm has led to several heuristics for the general problem by scaling and then rounding the distances to integers [3, 8, 10, 15]. Goel et al. [8] find an ε-approximation in time $O((m + n \log n)L/\varepsilon)$, where L is the (combinatorial) diameter of the graph G. It finds a $s-t$-path having cost at most the cost of an optimum $s-t$-path for any other vertex $t \neq s$. However, its total distance is bounded from above by $(1+\varepsilon)D$.

The recurrence relation (1.1) suggests how to modify Joksch's algorithm to solve the optimum mission planning problem. Since the algorithm can keep track of the shortest paths computed so far, i.e. those paths which achieve the value $w[v, i - d_{uv}]$ for the neighbors u of a vertex v, we can just restrict the minimum to be taken over those neighbors u of v for which the edge $l[u, i - d_{uv}]$ and the edge uv build a feasible subpath, i.e. the angle between those two edges is not $\pi/4$. Moreover, we compute a shortest path from the target to all other vertices in the grid having total distance at most $D/2$. Finally, we compute the cost of the concatenation of the shortest paths from the target to any pair of boundary grid points and, among those pairs for which

Input: A grid graph $G = (V, E)$ on the region $\mathcal{R}$ with costs $c : E \to \mathbb{R}^+$ and distances $d : E \to \mathbb{R}^+$, a target $t \in V$ and $D > 0$ as described above (for simplicity D even).

Output: A path entering and exiting the boundary of $\mathcal{R}$, going through t, having no two consecutive edges with an angle of $\pi/4$ and having total distance at most D.

1. Initialize $w[t, i] = 0$ for all $0 \leq i \leq D/2$.
2. Initialize $w[v, i] = \infty$ for all $v \in V \setminus \{t\}$ and $0 \leq i \leq D/2$.
3. Initialize $l[v, i] = NIL$ for all $v \in V \setminus \{t\}$ and $0 \leq i \leq D/2$.
4. For all $i = 1, \cdots, D/2$, $v \neq t$ do:

   4.1 Compute

   $$w[v, i] = \min\{w[v, i-1], w[u, i - d_{uv}] + c_{uv} : uv \in E, d_{uv} \leq i, \angle(l[u, i - d_{uv}], uv) > \pi/4\}.$$

   4.2 If $w[v, i] = w[v, i-1]$ set $l[v, i] = l[v, i-1]$.

   4.3 If $w[v, i] = w[u, i - d_{uv}] + c_{uv}$ for some $u \in V$ set $l[v, i] = uv$.
5. Let B be the set of grid points on $\partial\mathcal{R}$.
6. Compute

   $$\min_{u,v \in B}\{w[u, D/2] + w[v, D/2] : \text{ the concatenated paths from } t \text{ to } u \text{ and } v \text{ have no } \pi/4 \text{ angle}\}.$$
7. Return the path which achieves the minimum in Step 6.

Figure 1.3: A dynamic programming algorithm for optimal mission planning based on Joksch's algorithm.

the concatenation does not violate the angle constraint at the target, we choose the one of minimum cost. An overview of this algorithm is given in Figure 1.3.

We now analyze the running time of the algorithm. Let n be the number of grid points and b be the number of grid points on the boundary. Since the number of edges in the grid graph G is linearly bounded by n ($|E| \leq 4|V| = 4n$), we have $O(nD)$ iterations in step 4. Moreover, we have to compute the minimum of a set of size b^2 in step 6. Hence the overall running time is $O(nD + b^2)$.

The number of grid points and boundary grid points depends on the shape of the region and on the minimum turn radius. For example, in a square region with side length L we have $\sim (L/R_{min})^2$ grid points and $\sim 4(L/R_{min})$ boundary grid points. In general, any "reasonably" shaped region (e.g. convex regions) with diameter L should have approximately $O((L/R_{min})^2)$ grid points and $O(L/R_{min})$ boundary grid points and therefore the complexity of any algorithm on the grid we defined will depend largely on the value $M = L/R_{min}$. In particular, in these terms the dynamic programming algorithm from Figure 1.3 has running time $O(M^2 D)$. Note that this running time only depends on the region (L) and on the type of the aircraft (R_{min}, D). However, in the algorithm we assumed already a given cost function c and distance function d. The running time for computing c and d is $O(M^2(r + f))$, where r is the number of radars and f is the number of no-fly zones.

In the following we discuss how optimal the path returned may be. Whereas Joksch's algorithm finds an optimum path if all distances are integers, we have to deal with additional difficulties. First, the distances on the grid are 1 and $\sqrt{2}$. We can overcome this by using the distances 100 and 142 instead and also scaling up the distance limit D by 100. This introduces an error of at most 1%. Moreover, the recurrence (1.1) does not yield a proof for optimality if we add the angle constraint. A shortest feasible path, which obeys the angle constraint, may not contain shortest paths to all vertices on that path, since the angle constraint may force us to make a detour to one vertex, but not to a neighbor of that vertex. A computational analysis for different scenarios can show how optimal our heuristic works for the optimal mission planning problem.

1.2.3 A Linear Programming Approach

When trying to solve the CSPP as an integer linear program we will only consider a directed graph $G = (V, A)$. The following is an integer formulation of the CSPP:

$$\text{minimize} \sum_{a \in A} c_a x_a \text{ subject to:}$$

$$\sum_{a \in A} d_a x_a \leq D \tag{1.2}$$

$$\sum_{a \in \delta^+(v)} x_a - \sum_{a \in \delta^-(v)} x_a = \begin{cases} 1, & \text{if } v = s \\ -1, & \text{if } v = t \\ 0, & \text{if } v \in V \setminus \{s, t\} \end{cases} \qquad \forall v \in V \tag{1.3}$$

$$x_a \in \{0, 1\} \qquad \forall a \in A$$

where $\delta^+(v)$ denotes the set of arcs leaving the vertex v and $\delta^-(v)$ denotes the set of arcs entering v. Equality (1.3) ensures the set of arcs a having $x_a = 1$ indeed form an $s - t$-path and inequality (1.2) describes the constraint on the total distance of the path.

The main approach for solving the integer program for the CSPP is to use a Lagrangian relaxation of the problem. Handler and Zang [9] solve the Lagrangian dual by using Kelley's cutting plane method [12]. They then close the duality gap using the kth shortest path algorithm of Yen [18]. Beasley and Christofides [2] use sub-gradient optimization to solve the Lagrangian dual of a more general problem and then close the duality gap by using a branch and bound procedure. We refer to [6] for a survey on the Lagrange relaxation methods for the CSPP.

We now propose a linear programming formulation for the model of the optimum mission planning problem as described in section 1.2.1. We have a grid graph where the grid is bounded by a polygonal region $\mathcal{R}$, in which each vertex in the interior of the region is adjacent to its eight surrounding neighbors. Since we have a directed graph in the linear program, each pair of vertices neighboring u, v is connected by two directed arcs $\overrightarrow{uv}$ and $\overrightarrow{vu}$.

minimize $\sum_{a \in A} c_a x_a$ subject to:

$$\sum_{a \in A} d_a x_a \leq D \tag{1.4}$$

$$\sum_{a \in \delta^+(v)} x_a - \sum_{a \in \delta^-(v)} x_a = 0 \qquad \forall v \in \mathcal{R} \setminus \partial\mathcal{R} \tag{1.5}$$

$$\sum_{a \in \delta^-(t)} x_a \geq 1 \tag{1.6}$$

$$\sum_{v \in \partial\mathcal{R}} \sum_{a \in \delta^+(v)} x_a = 1 \tag{1.7}$$

$$x_a + x_b \leq 1 \qquad \forall a, b \in A : a, b \text{ adjacent and } \angle(a, b) = \pi/4 \tag{1.8}$$

$$x_a \in \{0, 1\} \qquad \forall a \in A.$$

We now describe the constraints (1.4)-(1.8). As before, (1.4) makes sure that the total distance of the path is at most D. For every interior vertex v the number of arcs entering and leaving v has to be equal, as defined in (1.5). Moreover, (1.6) and (1.7) make sure that at least one arc enters the target t and that exactly one arc enters the interior of the region $\mathcal{R}$ from the boundary. Then (1.5) implies that the path must also leave the target and eventually enter the boundary again. Since the airplane cannot make a turn below its minimum turn radius, we forbid the path to use two consecutive arcs which form an angle of $\pi/4$ in the grid by imposing the constraints (1.8).

In further research it has to be evaluated if the linear programming methods that were successfully used to solve the CSPP can also be used to solve our variant of the linear programming formulation.

1.2.4 Summary

First, we turned the optimum mission planning problem into a constrained shortest path problem on a grid graph. We have presented two different approaches to solve this particular constrained shortest path problem: a dynamic programming algorithm and a description of the problem as a linear program.

In the dynamic programming approach we suggested a modification of Joksch's algorithm to solve the optimum mission problem. The linear programming formulation we propose is just a slight variant of the linear program for the CSPP and we therefore hope that the same methods used to solve the LP for the CSPP will also work for our formulation.

It is not clear which approach will yield significantly better results in practice, since they are of quite a different structure. In future work, we would like to develop an algorithm to solve the linear model. Other reasonable future work could involve a comprehensive numerical comparison of these two approaches.

1.3 Stochastic Model

1.3.1 Introduction and Motivation

In order to model a flight path for the plane to follow, we consider again the grid defined for the discrete model. We then would like to create different paths the plane could follow. Again we use a square grid allowing for connections from one vertex to any of its 8 surrounding vertices. Moreover, we consider each vertex in the grid to be a single state space in a Markov chain, allowing for a Markov chain Monte Carlo (MCMC) approach to solve this problem. This model allows for easy adaptation to many limitations or expansions that might be necessary by a slight alteration of the algorithm.

Some of the limitations we consider are the turning radius of the plane and the maximum distance the plane can fly based on its fuel load. Also, no-fly zones and radars are factored into the model. The turning radius problem was easy to handle by switching to a second order Markov chain. By considering the previous movement, we simply limit the next possible moves. With this design in place we would not make movements that the plane could not physically handle. To account for the maximum distance the plane can fly, we set up the algorithm to keep track of the total distance moved in the grid. No-fly zones were designed into the algorithm by not allowing movement into nodes that were defined to be part of a no-fly zone. Finally, radars were accounted for through a cost function in the algorithm. The cost function, which will be defined later, was based on a few different factors including proximity to radar and to the boundary of the region. It is also desirable that our definition allow for expansion of the algorithm to incorporate other factors in the future. Since we use a MCMC approach, we then use this cost function to define the probabilities for each movement from one state space to the next.

By running a few test simulations it would be possible for us to incorporate a sensitivity analysis to determine the amount of error that our modeled path would incur with respect to the path of actual least cost. To do this we just need to run the algorithm about 50 times to completion, and record the total score for each path. Once we have these scores we can compute their sample variance in the standard manner and use the Central Limit Theorem to find out how many paths (n) we need to try at random until we are reasonably certain that we have found the best possible path. We find the following bound:

$$n \geq \frac{Z_\alpha^2 S^2}{C^2}, \tag{1.9}$$

where Z_α is the value from the standard normal distribution based on a $100(1-\alpha)$ percent confidence interval that we want our accuracy to be within C units of cost from the optimal flight path if the sample variance of the 50 test paths is S^2.

1.3.2 The Algorithm

The algorithm we use to find a near optimal path to and from the target is fairly straightforward. We form a grid on the region and take many random walks from the target to the boundary of the region, favoring those walks which minimize the probability of radar detection. Because of this favoritism and the large number of walks that are sampled, the best sample walk will be nearly optimal. To find a complete nearly optimal path (that is, one from the boundary, over the target, and back to the boundary), we combine two of the low-cost paths which obey the turn radius condition.

Now, we consider the algorithm in more detail. The first step is to form a grid over the desired region. For now, we assume that a uniform grid is placed on the region, where the the distance between two vertically or horizontally adjacent points is equal to the turn radius, as in the discrete method. This allows for more allowable moves on the grid. Later, we discuss the advantages of using Centroidal Voronoi Tesselations or other non-uniform, data dependent grids to improve our results. After the grid is in place, we assign a penalty to each node in the grid dependent on its distance to the center of various radars and whether it lies in a no-fly zone. We denote a node in the grid by I, let $R = \{R_i\}$ be the set of p radars for $i = 1 \ldots p$, and let $N = \{N_i\}$ for $i = 1 \ldots q$ be the set of q no-fly zones. The probability of entering states $I \in N_i$ for any i is set to zero. Otherwise, we consider each radar the node may fall into. For each radar R_i, let $radius_i$ be the radius of the radar and $dist(I, R_i)$ be the Euclidean distance of I to the center of R_i. Then, we define the penalty on I from this radar to be

$$pen_i(I) := \begin{cases} 1, & \text{if } I \notin R_i, \\ 1 + \dfrac{radius_i - dist(I, R_i)}{radius_i}, & \text{otherwise.} \end{cases}$$

Therefore, we may calculate the total penalty for a given node by

$$Pen(I) := \prod_{i=1}^{p} pen_i(I) \tag{1.10}$$

The advantage of defining the penalty on a node in this manner is that the node will receive a more severe penalty the closer it is to the center of a radar.

Next, we begin taking a random walk on the grid, starting at the target and moving towards the boundary of the region. The first step may be made completely at random. When deciding on the next step, a number of things must be taken into consideration. First of all, we must consider the previous move we made. The reason for this is that the plane will have a restricted turning radius, limiting its next possible moves. Since the grids in the discrete model and the stochastic model are identical, so are the sets of valid moves. Refer to Figure 1.2 for an illustration.

Of course, our possible moves may be limited further depending on whether they would move the aircraft into a no-fly zone. Such invalid moves will never be chosen because we set their probability of occurrence equal to zero. There may be some cases where every possible move will enter a no-fly zone. If this occurs, we simply discard the run and restart automatically. Another possible solution may be to further penalize nodes just outside of a no-fly zone so that the probability of having no valid move, and therefore having to discard a run, is low. Either way, we may ignore these illegal moves in the following discussion. Second, we consider the cost incurred by a certain move. Each possible step has an associated move cost which is calculated by using the radar/no-fly penalty and the distance of the desired node to the boundary. In other words, a possible move that leaves us closer to the boundary will have a lower cost than one that moves us farther from the boundary. At the same time, a move that places the aircraft within the radii of many radars will have a greater cost than one that moves it out of a radar. For example, consider Figure 1.4. The possible moves are to nodes a, b, c, d, and e. The move cost to node e would be the highest, because it is the center of a radar. The move costs of nodes a and b would be the smallest, because they are not in any radar and they are on the boundary. For a general move, we may define the move cost by considering movement of the aircraft towards the boundary. Let M_i for $i = 1, \ldots, 5$ be the five nodes we can possibly move to. For each move, define $dist(M_i, Exit)$ to be the Euclidean distance of the possible node to its nearest exit. Then, the move cost is defined by

$$MoveCost(M_i) := Pen(M_i) \cdot dist(M_i, Exit), \tag{1.11}$$

where $Pen(M_i)$ is defined as above.

Once the move cost is calculated, we find the probability with which each possible step will occur. This is determined by the move cost in the following way:

$$Prob(M_i) := \frac{\sum_{j \neq i} MoveCost(M_j)}{4 \sum_{j=1}^{5} MoveCost(M_i)}. \tag{1.12}$$

Because the probability is defined in this way, moves with lower costs will occur with a higher probability than those with higher cost. Returning to Figure 1.4, the probabilities of moving to nodes a or b would be the

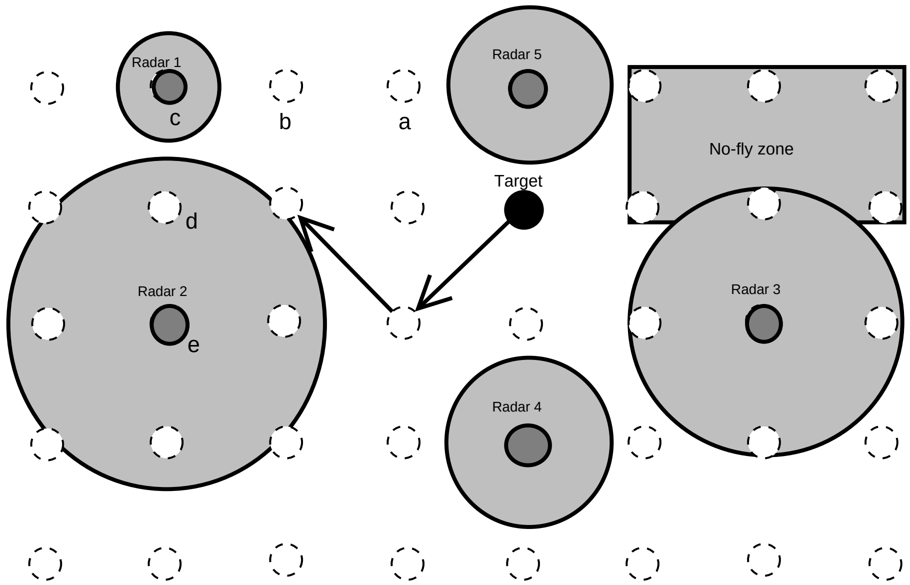


Figure 1.4: An example of a Markov decision.

highest and the probability of moving to nodes c or e would be smaller. The probability of moving to node d would be somewhere in the middle because it is in a radar zone, but it is not at the center. Because we give preference to moves with lower costs, we may compare our algorithm to a greedy algorithm. However, we still allow the possibility of a high cost move in case it will result in a lower total cost in the long term. Once these probabilities are determined, the next move is chosen and we repeat the entire process of determining a next move. The algorithm terminates when one of the following four things happens: the aircraft reaches the boundary of the region, the aircraft reaches a node where no possible moves exist (for example, all possible moves land in a no-fly zone), the distance of the random walk is too long, or the cost of the random walk is already greater than the maximum cost determined by a straight line walk from the target to the boundary doubled. The total cost computed in the algorithm is for two seperate runs from the target to the boundary combined. Markov chain theory tells us that after finding many random walks ("many" will be defined by the size of the problem), the best will be near optimal.

1.3.3 Matlab Implementation of the Markov Chain Approach

One may think of this problem as finding many possible exit routes from the target and forming a legal combination (one that does not violate the turn radius condition) of two of the best ones for a nearly optimal route. We require input from the user in regards to the grid size, the turn radius, radar locations, no-fly zone locations, target location, maximum distance, and maximum cost. The grid size is some input, (m, n) where m denotes the horizontal length and n denotes the vertical length. The turn radius is some input h. We define the distance between two horizontally or vertically adjacent points to be h, allowing for five legal movements. Because radar regions are typically circular, the locations of the radar are input as a radar matrix where each

Step 1: Define the grid with the distance between adjacent points equal to the turn radius.

Step 2: Calculate the radar/no-fly penalty for each node, I, in the following way:

(a) For each radar, R_i, the penalty associated with I is

$$pen_i(I) := \begin{cases} 1 & \text{if } I \notin R_i, \\ 1 + \dfrac{radius_i - dist(I, R_i)}{radius_i} & \text{otherwise.} \end{cases}$$

(b) Then, the total penalty for the node I is defined by

$$Pen(I) := \begin{cases} 10^6 & \text{if } I \in N_i \text{ for some } i = 1, \ldots, q, \\ \displaystyle\prod_{i=1}^{p} pen_i(I) & \text{otherwise.} \end{cases}$$

Step 3: Set the origin to be the target and choose a first move completely at random. Record in what direction this move was (for example, $[1, 1]$ would indicate a move up one and right one).

Step 4: Choose the next move in the following way:

(a) Define the next possible directions for movement. (In our example where the previous move was $[1, 1]$, these would be $[1, 1], [0, 1], [1, 0], [-1, 1]$, and $[1, -1]$.)

(b) For moves leading into a no-fly zone, define the probability of occurring to be 0. If the probability of each possible move is 0, terminate the program but do not increment the run number counter (so the sensitivity analysis will be consistent with the necessary number of runs).

(c) For each move with non-zero probability, determine the closest exit node, w. Use this and the radar/no-fly penalty to calculate the move cost as:

$$MoveCost(M_i) := Pen(M_i) \cdot dist(M_i, Exit).$$

(d) For each move with non-zero probability, calculate the probability of the move occurring as:

$$Prob(M_i) := \frac{\sum_{j \neq i} MoveCost(M_j)}{4 \sum_{j=1}^{5} MoveCost(M_i)}.$$

(e) Let a random number in the interval $[0, 1]$ determine the next move based on the probabilities from part (d).

(f) Update the location on the grid, what the previous move was, the distance traveled, and the cost. If the aircraft has reached the boundary, record results, increment the run number counter, and terminate the program. If the distance traveled is greater than the allowable range, then terminate the program but do not increment the run number counter (so the sensitivity analysis will be consistent with the necessary number of runs). If the total cost is higher than the maximum cost, then record for sensitivity analysis and restart the program. Otherwise, repeat step 3.

Figure 1.5: Outline of the random walk on the grid.

row refers to a radar, the first two columns describe the center of the radar, and the last column is the radius of the radar. No-fly zones have the potential of being very odd shapes so we would like to approximate them by more familiar objects. In our case, we approximate by rectangles. Therefore, the no-fly zones are input as another matrix where each row is a rectangle approximating a no-fly zone, the first two columns describe the coordinates of the bottom left corner, and the last two columns indicate the length and width of the rectangle. The target location is simply a 1×2 vector describing the target coordinates. The input of the maximum distance and the maximum cost allow us to terminate the program should the path become too long or too costly.

One of the outputs will be a $k \times 2$ matrix, L, where each row defines the coordinates of one move. Also, the cost and total distance traveled associated with the random walk, L, will be provided.

Figure 1.5 outlines the steps the algorithm described in the previous section. The program should be run multiple times to calculate nearly optimal paths for each exit from the target. Then, two low-cost paths from the target to the boundary may be combined for a single low-cost path through the entire space, passing over the target (provided they form a legal combination). For actual Matlab code, please contact the authors.

1.3.4 Advantages and Disadvantages

One of the advantages of the stochastic model is that it allows for unconstrained entry and exit points. The probability of movement to a particular node is calculated using the nearest exit point, allowing for many possible entry and exit nodes.

As mentioned before, the stochastic method we employ has an important advantage over a greedy algorithm, which always chooses the next move with lowest cost. However, because of the way the movement probabilities are defined, there is still a chance that a higher cost move will be made, resulting in a lower total cost.

Another advantage of the stochastic model is that many constraints such as the minimum turn radius, the risk of radar detection, and no-fly zone restrictions are easily incorporated. For example, the turn radius constraint is easily satisfied by restricting certain movements at every step of the random walk. The risk of radar detection is proportional to the radar penalty defined above and is similar to a Gaussian probability density function. Therefore, nodes that are closer to radar centers will have a higher penalty. Finally, the nodes in the no-fly zone are simply assigned a probability of zero, never allowing them to occur on the path.

We take random walks which have probabilities based on a constructive heuristic, resulting in a better approach than simulated annealing. We always look for a global optimum, whereas the simulated annealing looks at a local minimum and then makes jumps to find a global minimum. Also, programming those jumps can be very difficult. Furthermore, simulated annealing may not converge to a global minimum whereas the stochastic method has desirable convergent properties.

Another advantage of a method using a discrete grid is that it may easily deal with conversion to and from latitude and longitude coordinates. In addition, there are no limitations on the dimensions of the model since the grid may simply be defined as a k-dimensional state space.

Given that we used a MCMC approach to find an optimal solution, the resulting path will not be the true optimum. Our solution will, however, be very close to the optimum and interpolation could create a smooth optimal path. Also, there could potentially be a large run time associated with the algorithm which could be reduced significantly using parallel computing. By delegating the work among many processors, more paths may be computed in a short time.

1.3.5 Extensions

One thing to consider for further research is how to interpolate our resulting paths to find one that would be smoother and more realistic for a plane to follow.

Another way to improve results is to consider Voronoi Tessellations, where the grid is no longer uniform, but instead it is defined based on radar and no-fly zone locations. This kind of data dependent grid may allow low cost movements that otherwise would not exist. The most difficult factor to resolve when considering any non-uniform grid is how to define possible moves such that the turn radius condition will always be satisfied.

Since we are assuming that the aircraft has standard maneuvering abilities in terms of the bank angle, the uniform grid size will have a large effect on the steepness of angles that the plane can make. Hence we decided that at any given coordinate the plane would have only five possible moves to ensure that the plane does not make a turn smaller than is allowed by the turn radius. Theoretically, the turning angle α satisfies $\alpha = \frac{v^2}{g \tan \theta}$, where v is the velocity of the plane, g is the acceleration of gravity and θ is the bank angle. So slowing down the velocity to the stall speed gives you a smaller bank angle. Furthermore, the radius of the turn is greatly diminished by increasing the angle of the bank from zero degrees to 45 degrees. In this model we assumed that the velocity is constant and so we were only concerned about the turn angle. In future research, it may be advantageous to consider change in velocity. One way to reduce the probability of exposing the top of the plane to radar is to penalize certain turns. For example, the cost of a 90 degree turn may be multiplied by 1.5. By increasing the penalty, a 90 degree turn may still occur but with lower probability since it carries a higher risk of being detected by radar.

As mentioned in Section 1.3.4, the stochastic model is easily extended to higher dimensions. However, we have not yet pursued this research. When trying to account for the bank angle in three dimensions there is an issue of exposing the top of the plane, which allows for accurate signals back to the radar. So if the radar detection range is $R_0\sigma^{\frac{1}{4}}$, where R_0 is the nominal radar detection range and σ is the aspect dependent radar cross section, then we would need to consider adding a turning cost into the movement cost function which depends on distance to the center of the radar as well as the bank angle when turning into a radar.

1.4 Continuous Model

1.4.1 Calculus of Variations

Another very natural approach to this problem is one that finds a continuous path to the target subject to the constraints given. The scenario is translated into a Boundary Value Problem with specified origin, target, and destination points. We actually solve two problems which differ only in boundary conditions; one from the origin to the target, the other from the target to the destination. The standard method of calculus of variations is applied for a multiplicative line integral formulation, resulting in the optimal path which minimizes total radar track time and flight length.

1.4.2 The Model

This method calculates an optimal path by minimizing the value of the following cost function:

$$J = \int F(x, y(x), y'(x))dx, \tag{1.13}$$

$$F(x, y(x), y'(x)) = C(x, y(x))\sqrt{1 + y'(x)^2}. \tag{1.14}$$

$C(x, y)$ is a multiplicative weighting function defined on the region map which takes a value of 1 in non-restricted zones, but higher values in no-fly zones and within radar radii. Thus minimizing the cost function will not only minimize distance of the path, but also the time in radar track, as cost travel through these areas is weighted very heavily. Note that cost weighting in no-fly zones must be considerably higher than that in radar zones and elsewhere.

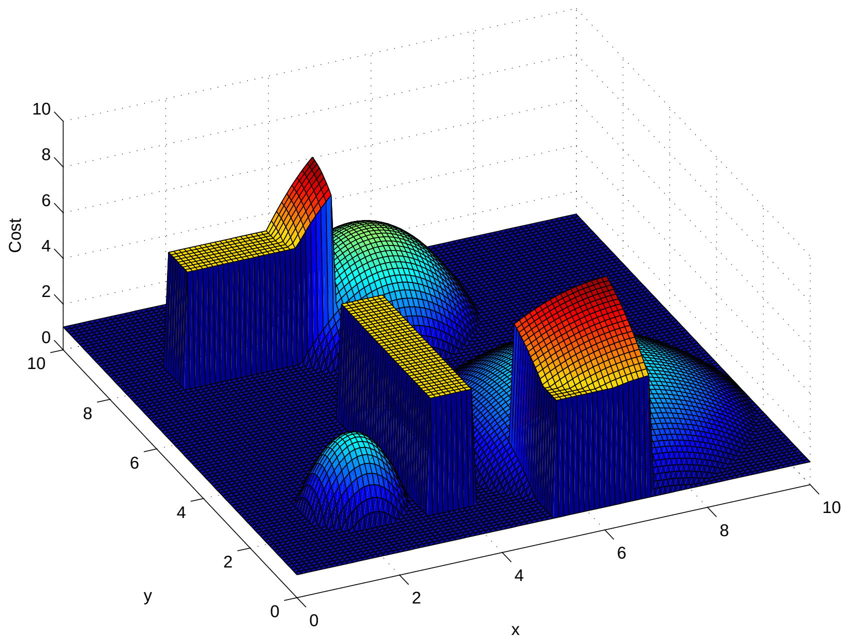


Figure 1.6: A simulated weighted cost function. Note the variable locations, radii, and strengths of radar zones, as well as higher cost of no-fly zones.

It is known that if there exists an optimal curve which minimizes this J, then it satisfies the Euler-Lagrange differential equation:

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) = 0. \tag{1.15}$$

Substituting yields

$$\frac{\partial}{\partial y}(C\sqrt{1 + (y')^2}) - \frac{d}{dx}\left(\frac{\partial}{\partial y'}(C\sqrt{1 + (y')^2})\right) = 0, \tag{1.16}$$

and rearrangement produces

$$\frac{\partial C}{\partial y}\sqrt{1+(y')^2} = \frac{d}{dx}\left(\frac{Cy'}{\sqrt{1+(y')^2}}\right), \tag{1.17}$$

with boundary conditions $y(0) = \alpha$ and $y(L_1) = \beta$ for the entry route and $y(L_1) = \beta$ and $y(L_1 + L_2) = \gamma$ for the exit route.

Due to the non-linearity of this differential equation, the conventional methods of determining a solution cannot be used. A finite difference method is implemented here. An initial solution $y_0(x)$ is specified, which correctly satisfies the boundary conditions. The interval $(0, L)$ is discretized into $N+1$ points for each route, $0 = x^1, \ldots, x^{N+1} = L_1$ for entry and $L_1 = x^{N+1}, ..., x^{2N+1} = L_1 + L_2$ for exit, with step size h. Then a system is constructed using the finite difference approximations:

$$\begin{pmatrix} a_{1,1} = \frac{1}{h^2} & 0 & 0 & \ldots & \ldots & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ldots & 0 & a_{N,N-1} & a_{N,N} & a_{N,N+1} \\ 0 & \ldots & \ldots & \ldots & 0 & a_{N+1,N+1} = \frac{1}{h^2} \end{pmatrix} \begin{pmatrix} y^1 = \alpha \\ y^2 \\ \vdots \\ \vdots \\ y^N \\ y^{N+1} = \beta \end{pmatrix} = \begin{pmatrix} b_1 = \frac{\alpha}{h^2} \\ b_2 \\ \vdots \\ \vdots \\ b_N \\ b_{N+1} = \frac{\beta}{h^2} \end{pmatrix},$$

where for $i = 2, \ldots, N$,

$$a_{i,i-1} = \frac{1}{h^2}\frac{C(x^i - \frac{h}{2}, \frac{y^i + y^{i-1}}{2})}{\sqrt{1 + (\frac{y^i - y^{i-1}}{h})^2}}, \tag{1.18}$$

$$a_{i,i+1} = \frac{1}{h^2}\frac{C(x^i + \frac{h}{2}, \frac{y^i + y^{i+1}}{2})}{\sqrt{1 + (\frac{y^{i+1} - y^{i}}{h})^2}}, \tag{1.19}$$

$$\text{and} \quad b_i = \frac{\partial C}{\partial y}(x^i, y^i)\sqrt{1 + (\frac{y^{i+1} - y^{i-1}}{2h})^2}. \tag{1.20}$$

An iterative method is then implemented to generate a solution to the nonlinear system. This method relies on the proposed guess to the solution to determine the finite differences matrix. An "inversion" is then implemented to left divide the matrix to leave the y vector alone on the left side. The resulting product on the right hand side of the equation is then computed, resulting in the iterative solution to y. The iterations are then continued until the resulting difference between iterations is below a given tolerance or a maximum iteration count is reached for a non-convergent solution. Convergent solutions then provide a reasonable approximation to the analytic solution to the differential equation, providing an optimal flight path. The exit path problem is solved in a similar manner, incorporating those respective boundary conditions with an additional condition imposed on $y'(x^{N+1})$. In this way, the value of x^{N+2} is specified when solving the exit route; this condition ensures smoothness across the partition at x^{N+1}.

In three dimensions, the non-linear minimal surface equation (1.17) will have this form

$$\frac{\partial C}{\partial z}\sqrt{1+|\nabla u|^2} = \frac{\partial}{\partial x}\left(\frac{C\frac{\partial u}{\partial x}}{\sqrt{1+|\nabla u|^2}}\right) + \frac{\partial}{\partial y}\left(\frac{C\frac{\partial u}{\partial y}}{\sqrt{1+|\nabla u|^2}}\right). \tag{1.21}$$

Compared with the approach described in [19] that is designed mainly for isolated radars (treated as point sources), one of the advantages of using the PDE form above is the flexibility of the weighted function $C(x, y)$ which can be used to model radars, no-fly zones, geological and geometric information. The method can be used in conjunction with the level set method. The turn angle can be controlled by the curvature of the zero level set function. The difficulty is how to solve the non-linear partial differential equation above efficiently and how to find the global minimum of the variational form. In [14], some techniques are proposed to solve such weighted minimum surface problems with different applications.

1.4.3 Implementation and Concerns

The continuous model allows a considerable degree of flexibility in scenario modifications. The multiplicative nature of the line integral allows a number of attributes of both radars and no-fly zones to be easily manipulated by defining the weighting factor $C(x, y)$ in the cost function appropriately. The continuous model can also be readily extended to a restriction on the minimum turn radius, a constraint which was not incorporated here. A further constraint on the total flight length for the aircraft can be incorporated from the method of Lagrange multipliers, which was not implemented in this case. Extending the model to a third dimension can be accomplished in a similar manner, where level set methods can provide the optimal path solution.

The most significant disadvantage to this model comes into play when solving the differential equation. Classical methods of direct solution are not effective with the addition of even one radar, and the finite differences method is extremely sensitive to initial conditions and hazard number and placement. Many scenarios prevent the algorithm from converging to a close approximation of the solution. The scenarios may deem an initial guess to the path an impossibility, developing a solution that is computationally taxing and potentially inefficient. Further limitations on finding the optimal solution arise from the necessary conditions needed for the differential equations. Many scenarios may be present where the optimal entry and exit locations are unknown, necessitating unconstrained end points. Therefore, the continuous model should be utilized when data exists for recommended entry and exits into the target zone.

The continuous model is based on solving a differential equation, which facilitates many potential improvements. Analytic solutions and different algorithmic methods for solving nonlinear systems potentially exist, providing optimal solutions in a fast, efficient process. Development of the solution method will allow more complex cost functions to be analyzed, permitting more realistic scenarios and more complex constraints on the limitations of the aircraft.

1.5 Summary

The objective of routing an aircraft through a hostile airspace is one that is much more complex than initially foreseen. Length concerns alone can be satisfied by simply flying in a straight line over the target. However, this sort of path may impose high risk on the aircraft, while other paths may compromise path length to avoid high-risk regions. The limitations on various aircraft further impose challenges in developing a consistent model, as physical and logistical constraints vary widely. Numerous reasonable mathematical methods exist for solving such a constrained shortest path problem, yet many challenges arise in implementation.

Here three models have been developed that provide methods for locating a path which minimizes risk and constrains path length in various ways. The continuous model incorporates calculus of variations to find a smooth solution to the problem. However, the limitations on having specific and relevant data for entry and exit, along with some sort of guess at the optimal solution, lend this model to more simplistic situations. More complicated or uncertain situations can be handled more flexibly using either discrete or stochastic algorithms on a mesh grid defined on the region. The discrete algorithms include linear and dynamic programming methods and the stochastic algorithm uses the Markov chain Monte Carlo method. All algorithms define a cost function on the grid edges based on the radars and no-fly zones to formulate the risk of flying over that particular part of the region. The constraints on the total length of the path and on the minimum turn radius are incorporated differently in these algorithms.

The models developed throughout this paper include many generalized assumptions that simplify the simulated situations. Further extensions include incorporating more complicated constraints and aircraft variabilities. These include expanding the model into a three dimensional space, such that changes in altitude can be implemented in the modeling scheme. Another reasonable inclusion is to account for varying aircraft speed in the models, as constant speed has been assumed thus far. This factor may be beneficial, as the risk for an aircraft may be much less when its speed increases within the radar zone. Locations of radar zones are further assumed to be definitively known, when realistically these locations may only be estimated with some degree of uncertainty. These can be incorporated into the various models by adjusting the corresponding probabilities and/or cost functions. While this extension seems simplistic, developing the distributions of the corresponding locations may be complicated and costly. These complex situations may be beneficial if run in a real time situation, as information may develop as the flight is being performed. Many other extensions exist to be included in one or more of the preceding models, simplified to some extent by the weighted cost function included throughout the model development.

The modeling of aircraft through various hostile locations is a complex process. The inclusion of various constraints and objectives further compounds the problem, making any solution a non-trivial matter. The development of various models has been described throughout this paper. The numerous methods presented provide many alternatives based on the situation perceived. Future development of these models and a computational analysis of the algorithms will provide great depth in obtaining optimal flight paths, for military and potentially commercial use.

Bibliography

[1] Ascher, U. M. and Petzold, L. (1998) *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, Philadelphia, PA.

[2] Beasley, J. E. and Christofides, N. (1989) An Algorithm for the Resource Constrained Shortest Path Problem, *Networks*, 19:379–394.

[3] Chen, S. and Nahrstedt, K. (1998) On Finding Multi-Constrained Paths, *International Conference in Communications (ICC'98).*

[4] Cheney, W. and Kincaid, D. (2002) *Numerical Analysis: Mathematics of Scientific Computing, Third Edition*, Brooks Cole, Pacific Grove, CA, 2002.

[5] Desrochers, M. and Soumis, F. (1988) A Generalized Permanent Labeling Algorithm for the Shortest Path Problem with Time Windows, *INFOR 26*, pp. 191–212.

[6] Dumitrescu, I. and Boland, N. (2001) Algorithms for the Weight Constrained Shortest Path Problem, *International Transactions in Operational Research*, 8:15–29.

[7] Dumitrescu, I. and Boland, N. (2003) Improved Preprocessing, Labeling and Scaling Algorithms for the Weight-Constrained Shortest Path Problem, *Networks*, 42:135-153.

[8] Goel, A., Ramakrishnan, K. G., Kataria, D. and Logothetis, D. (2001) Efficient Computation of Delay-Sensitive Routes from One Source to All Destinations, *INFOCOM*, pp. 854-858.

[9] Handler, G. Y. and Zang, I. (1980) A Dual Algorithm for the Constrained Shortest Path Problem, *Networks*, 10:293-309.

[10] Hassin, R. (1992) Approximation Schemes for the Restricted Shortest Path Problem, *Mathematics of Operations Research*, 17:36-42.

[11] Joksch, H. C. (1966) The Shortest Route Problem with Constraints, *J Math Anal Appl*, 14:191-197.

[12] Kelley, J. E. (1960) The Cutting Plane Method for Solving Convex Programs, *Journal of SIAM*, 8:703-712.

[13] Kindl, M., Shing, M. and Rowe, N. (1991) A Stochastic Approach to the Weighted-Region Problem, I: The Design of the Path Annealing Algorithm, Technical Report A423342.

[14] Li, Z., Lin, X.-B., Torres, M. and Zhao, H. (2003) Generalized Snell's law for weighted minimal surface, *Methods and Applications of Analysis*, 10: 199-214.

[15] Lorenz, D. and Raz, D. (1999) Simple Efficient Approximation Scheme for the Restricted Shortest Path Problem, *Technical Report 10009674-991214-04TM*, Bell Labs.

[16] Mitchell, J. and Papadimitriou, C. (1991) The Weighted Region Problem: Finding Shortest Paths through a Weighted Planar Subdivision, *Journal of the ACM*, 38:18-73.

[17] Suh, S. H. and Shin, K. G. (1988) A Variational Dynamic Programming Approach to Robot-Path Planning with a Distance-Safety Criterion, *IEEE Journal of Robotics and Automation*, 4:334-349.

[18] Yen, J. Y. (1971) Finding the k-Shortest, Loopless Paths in a Network, *Management Science*, 17:711-715.

[19] Zabarankin, M., Uryasev, S. and Pardalos, P. (2001) Optimal Risk Path Algorithms, in *Cooperative Control and Optimization* (Murphey, R. and Pardalos, P. eds.), pp. 271-303.

Chapter 2

Mathematical Modeling of Enzyme Action in Fuel Alcohol Production

PROBLEM 2: MATHEMATICAL MODELING OF ENZYME ACTION IN FUEL ALCOHOL PRODUCTION

Mihaela Froehlich[1], Mary Gruber[2], Kamyar Hazaveh[3], April Stepp[4], Chris Vogl[5], Xiaohai Wan[6], Xiaofeng Xia[7]

Problem Presenters:
Eric J. Allain, Ph.D. & Guillermo Coward-Kelly, Ph.D.
Novozymes North America, Inc.

Faculty Consultants:
Dr. Mette Olufsen & Dr. Charlie Smith
North Carolina State University

Abstract

Ethanol is an alcohol-based alternative fuel created from starches found in corn, wheat, potatoes and sugarcane.In the United States, the majority of ethanol comes from corn. Ethanol could become a practical alternative to gasoline if it could be produced efficiently at a low cost. Turning corn into ethanol is a multi-step process which includes: producing glucose from cornmeal through a series of enzymatic reactions, fermenting the glucose, and distilling the products of fermentation to create ethanol. In our project, we modeled the step in which cornmeal is turned into glucose by using Michaelis-Menten kinetics. We also analyzed the sensitivity of glucose production to the concentration of the different enzymes. Based on this analysis, some simplifications were made to the model. The resulting model and analysis may be used to reduce the total amount of enzymes used to convert the cornmeal to glucose and therefore reduce the total cost of producing ethanol.

2.1 Chemistry of Creating Ethanol

2.1.1 Description of Process

The process of turning corn into ethanol starts by milling corn into starch granules and placing the granules into water to create a starch slurry. This mixture is kept at 85 °C in order to convert most of the starch to a gelatinous form. In this form all of the starch molecules within a starch granule are available for the reaction which speeds up the conversion to glucose. The resulting shorter soluble dextrin mixture is then cooled to 32 °C so that yeast can be added. The enzyme glucoamylase, which hydrolyzes dextrin into glucose, is added

[1]Duke University
[2]Michigan State University
[3]Georgia Tech
[4]Western Carolina University
[5]Illinois Wesleyan University
[6]North Carolina State University
[7]Emory University

as well. The lower temperature allows yeast to survive and ferment the glucose into ethanol. This process is called simultaneous sacharification fermentation.

Some ethanol producers, however, are now using a no-heat method in which the starch slurry, enzymes and yeast are added simultaneously. This means that the temperature during the process is limited by the heat tolerance of the yeast. One advantage of this method is energy savings as a result of the reduced temperature. Another advantage is that a heat sensitive enzyme that is naturally occurring in corn will also work to produce glucose rather than being deactivated by the high temperature. The disadvantage to the lower temperature, however, is that the starch remains in granular form, and only the starch molecules on the surface of the granule are available to the enzymes. As the enzymes work, the granules shrink and eventually are broken down entirely. This is the method of ethanol production that we modeled.

2.1.2 Starch

On average each starch granule consists of $10,000$ starch molecules. Starch molecules come in two distinct forms: amylose and amylopectin. Amylose is a linear polymer that averages 300 glucose molecules per chain and makes up $20-30\%$ of starch molecules [4]. Amylopectin, which makes up the remaining $70-80\%$, is a branched starch that contains about 21 glucose molecules between branches [4].

As the starch molecules are broken up by the enzymes, four different sugars are produced: dextrins, maltotriose, maltose and glucose. Dextrins are polymers which are shorter and simpler than starch molecules but have at least four linked glucose molecules. Dextrins remain on the starch granule until they are decomposed into the other three simpler sugars. Maltotriose and maltose are respectively three- and two-glucose sugars which will break away from the starch granule into the liquids where enzymes will quickly convert them into the final sugar, glucose.

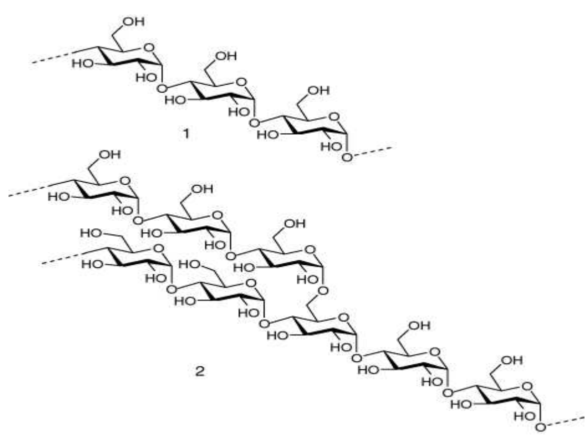


Figure 2.1: Molecule of amylose consists of chain of glucose molecules (1) and amylopectin (2) is a branched structure of glucose molecules.

2.1.3 Enzymes

Three specific enzymes are used to create glucose in the no-heat ethanol production. Two of the enzymes, α-amylase and glucoamylase, are added to the corn slurry while α-glucosidase is naturally present in the corn.

Enzyme α-amylase works on breaking any bond connecting glucose molecules within the starch. Therefore, depending on where it reacts in the chain, it can produce any of the four types of simple sugars. Enzyme α-amylase prefers to act on the longer starch and sugar molecules which are still part of the starch granules. On the other hand, glucoamylase prefers to act on simpler sugars and will only break the final bond on a chain. Hence, glucoamylase always produces glucose. The third enzyme, α-glucosidase, works only on maltose and maltotriose and therefore it always produces glucose.

When glucoamylase and α-glucosidase work on maltitrose the result is one maltose and one glucose. When glucoamylase and α-glucosidase work on maltose the result is two glucose molecules. Figure 2.2 shows the process by which starch becomes glucose.

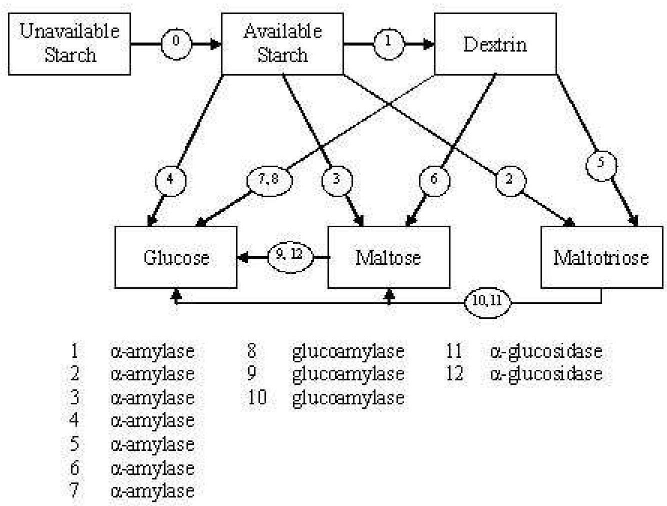


Figure 2.2: Diagram of the enzyme reactions that occur when converting starch granules into glucose. Each reaction is numbered with the corresponding enzyme involved.

2.2 Modeling Using Michaelis-Menten Enzyme Kinetics

All the reactions in the process of creating glucose are enzyme mediated and therefore can be modeled using Michaelis-Menten kinetics. In enzyme mediated reactions

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_{cat}} E + P.$$

The enzyme E and the substrate S, which in this case would be a starch or sugar, react to create the enzyme-substrate complex ES which yields the product and the unchanged enzyme. The constants k_1 and k_{-1} are the rates for the association and disassociation of substrate, and enzyme and k_{cat} is a first-order rate constant for converting the enzyme-substrate complex to the product and the recovered enzyme.

In a Michaelis-Menten model, the reaction rates determine the change in concentration of each substrate. The reaction rates are given by

$$r_i = \frac{k_{cat}[E][S]}{K_m + [S]}. \tag{2.1}$$

The terms $[E]$ and $[S]$ are the enzyme and the substrate concentrations, respectively. The Michaelis-Menten constant K_m is a concentration of the substrate at which the rate of reaction is half of the maximum reaction rate. Both constant terms are dependent on the specific enzyme and substrate involved in the reaction. The table in Figure 2.3 shows constants k_{cat} (in min^{-1}) and K_m (in $MolL^{-1}$) for all the reactions occurring in the specific process we are modeling. The table in Figure 2.4 shows typical concentrations of enzymes (in $MolL^{-1}$) used in the process. The total rate of change of concentration of a substrate is the sum of the rates

Reaction Number	Enzyme	Substrate	Product	k_{cat}	K_m
1	α-amylase	starch(available)	dextrin	14.48	37.17
2	α-amylase	starch(available)	maltotriose	0.16	0.46
3	α-amylase	starch(available)	maltose	0.36	0.09
4	α-amylase	starch(available)	glucose	n/a	n/a
5	α-amylase	dextrin	maltotriose	0.16	0.46
6	α-amylase	dextrin	maltose	0.36	0.09
7	α-amylase	dextrin	glucose	n/a	n/a
8	glucoamylase	dextrin	glucose	0.85	0.25
9	glucoamylase	maltose	glucose	0.13	1210
10	glucoamylase	maltotriose	maltose/glucose	0.27	360
11	α-glucosidase	maltotriose	maltose/glucose	1.07	1300
12	α-glucosidase	maltose	glucose	1.07	1300

Figure 2.3: Table of the k_{cat} and K_m used in the model.

into and out of the substrate. For example, Figure 2.2 shows that maltotriose is created from reaction 2 where α-amylase breaks down available starch and reaction 5 where α-amylase breaks down dextrins. Reactions 10 and 13 decrease the concentration of maltotriose when glucoamylase and α-glucosidase respectively create maltose and glucose. Therefore, the total change in concentration of maltotriose is given by:

$$\frac{dMT}{dt} = r_2 + r_5 - r_{10} - r_{11}. \tag{2.2}$$

Enzyme	Concentration
α-amylase	$9.8 \cdot 10^{-5}$
glucoamylase	$1.0 \cdot 10^{-4}$
α-glucosidase	$3.02 \cdot 10^{-2}$

Figure 2.4: Table of the enzyme concentrations used in the model.

Similarly, the changes in concentration of the remaining substrates are given by:

$$\begin{aligned}
\frac{dS_{unavailable}}{dt} &= -r_0, \\
\frac{dS_{available}}{dt} &= r_0 - r_1 - r_2 - r_3 - r_4, \\
\frac{dD}{dt} &= r_1 - r_5 - r_6 - r_7 - r_8, \\
\frac{dM}{dt} &= r_3 + r_6 + r_{10} + r_{11} - r_9 - r_{12}, \\
\frac{dG}{dt} &= r_4 + r_7 + r_8 + r_9 + r_{10} + r_{11} + r_{12}.
\end{aligned} \tag{2.3}$$

The resulting is a system of coupled, first-order differential equations that can be solved using Matlab and PLAS, an existing pharmacokinetic software package.

Since there is no way of measuring how much of the total starch is in a form of unavailiable or availiable starch, the reaction rate r_0 cannot be obtained experimentaly in the same way we found all the other reaction rates. However, we still want to model r_0 using Michaelis-Menten kinetics. Two known measurements that are useful in modeling r_0 are percentages of total starch converted to glucose measured 24 and 50 hours into the process. These values were 85% and 98% respectively. Since the k_{cat} and K_m values for overall reaction of all enzymes on the starch are unknown, the Michaelis-Menten equation is simplified; k_{cat} and $[E]$ are combined to form V_{max}:

$$r_0 = \frac{k_{cat}[E][S_{total}]}{K_m + [S_{total}]} = \frac{V_{max}[S_{total}]}{K_m + [S_{total}]}.$$

The two unknowns, V_{max} and K_m, can be found using two known values, ($24h$,$0.15[S_{total}]$) and ($50h$, $0.02[S_{total}]$). V_{max} and K_M are found to be 0.005 $MolL^{-1}min^{-1}$ and 250 $MolL^{-1}$ respectively. Using experimental data to determine the rate r_0 gives us an accurate time scale for the simulation.

2.3 Computational Implementation

2.3.1 Solving the Model Using Matlab

The model was implemented in *Matlab* using the built-in *ode45* ordinary differential equation solver. The solver uses a step size adjusting algorithm that adjusts the step size until the solution has a relative error less than 10^{-4}. The algorithm computes the error as the difference between a solution obtained through a 4^{th} degree Runge-Kutta approximation and a separate solution obtained through a 5^{th} degree Runge-Kutta approximation. The final result is a 4^{th} degree Runge-Kutta solution with an error bound of 10^{-4}. The *ode45* solver is used along with the kinetic constants shown in Figures 2.3 and 2.4, an initial starch concentration of 0.1512 $mMolL^{-1}$, and the following initial conditions:

$$\begin{aligned}
\frac{dS_{unavailable}}{dt} &= 0.1512,\\
\frac{dS_{available}}{dt} &= 0,\\
\frac{dD}{dt} &= 0,\\
\frac{dM}{dt} &= 0,\\
\frac{dMT}{dt} &= 0,\\
\frac{dG}{dt} &= 0,
\end{aligned}$$

to produce the following solution to the model (Fig. 2.5):

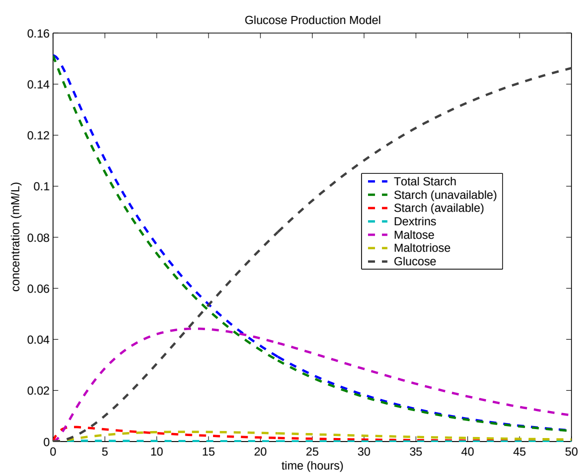


Figure 2.5: Total starch, available starch, dextrin, maltotriose, maltose and glucose concentrations over time. Notice that most of the starch becomes glucose in 50 hours. This corresponds to the actual production time in industry. Also, the shape of these graphs for total starch, maltotriose, maltose and glucose resemble the shapes of the graphs found in the industrial process.

2.3.2 Solving the Model Using PLAS

We also use an existing pharmacokinetic software package, *PLAS*, to solve this model. It provides an alternative approach to validate the results of *Matlab* codes. With this program, the only thing we need to do to solve the model is to input the ordinary differential equations, the initial conditions and the kinetic parameters so that we can focus on tuning the parameters of the model itself.

PLAS, developed by António E. N. Ferreire, is an interactive and user-friendly program for simulating dynamical system models [5]. It serves as a numerical toolbox to solve the ordinary differential equations by using the BDF (Backward Differentiation Formulae) solver, which is known as the Gear implementation of the BDF [2].

We use the same parameters as in the *Matlab* code to compare the results from the previous section.

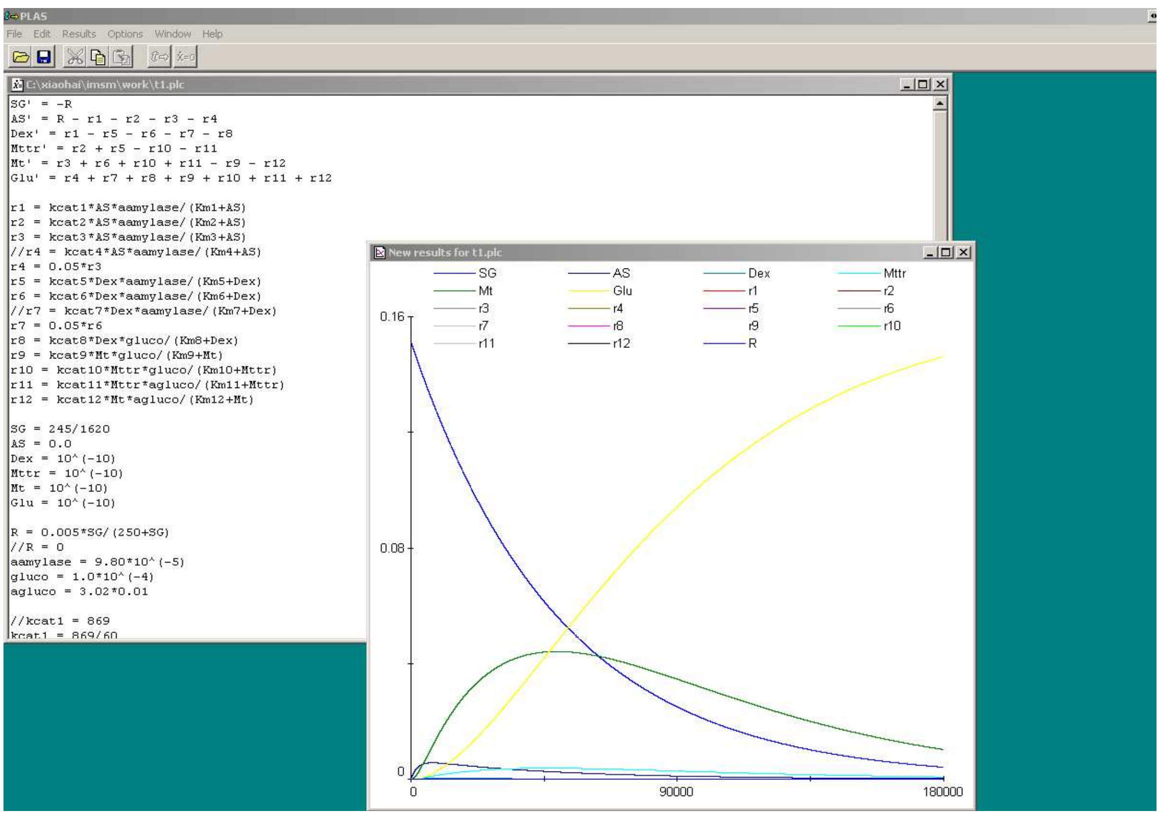


Figure 2.6: This screen shot of PLAS shows how simple it is to solve our model with this program. The resulting graph is the same as in Figure 2.5 (the horizontal axis represent time in *sec*).

2.4 Sensitivity Analysis

A sensitivity analysis was performed on the model in order to determine the overall effect of the concentration of α-amylase, glucoamylase, and α-glucosidase on glucose concentration (as well as the concentration of the other state variables). Knowing the effect of enzyme levels on glucose output is extremely useful in the glucose production process, since it may be possible to use significantly less enzyme to produce roughly the same amounts of glucose, lowering the overall production cost.

The analysis consists of finding the values of $\frac{d[StateVariable]}{d[\alpha-amylase]}, \frac{d[StateVariable]}{d[glucoamylase]}, \frac{d[StateVariable]}{d[\alpha-glucosidase]}$ for each of the six state variables. Since all six state variables depend on the Michaelis-Menten equations, we will start the derivation there. First we will rewrite equation (2.1) as:

$$r([E_X],[S]) = \frac{k_{cat}[E_X][S]}{K_m + [S]}. \tag{2.4}$$

To obtain the sensitivity of each rate, we differentiate equation (4.2) with respect to each enzyme (E_Y). Notice, however, that the derivative when E_X is the same as E_Y is not the same as when they differ:

$$(E_X = E_Y) \rightarrow \frac{d(r([E_X],[S]))}{d([E_Y])} = k_{cat}(\frac{[S] + [E_X]\frac{d([S])}{d([E_Y])}}{K_m + [S]} - \frac{[S][E_X]\frac{d([S])}{d([E_Y])}}{(K_m + [S])^2}),$$

$$(E_X \neq E_Y) \rightarrow \frac{d(r([E_X],[S]))}{d([E_Y])} = k_{cat}\frac{K_m[E_X]\frac{d([S])}{d([E_Y])}}{(K_m + [S])^2}.$$

To simplify this difference, we define the following two equations:

$$s_i = k_{cat}(\frac{[S] + [E_X]\frac{d([S])}{d([E_Y])}}{K_m + [S]} - \frac{[S][E_X]\frac{d([S])}{d([E_Y])}}{(K_m + [S])^2}), \tag{2.5}$$

$$d_i = k_{cat}\frac{K_m[E_X]\frac{d([S])}{d([E_Y])}}{(K_m + [S])^2}. \tag{2.6}$$

where we will use equation (2.5) when E_X and E_Y are the same and equation (2.6) when they are different.

Now that the rate sensitivities are defined, we must find the sensitivities of each of the state variables. This is done by defining a second system of coupled differential equations. We take the derivative of equations (2.2) and (2.3) with respect to each of the three enzymes and then, assuming continuity, reverse the order of differentiation. For example:

$$\frac{dG}{dt} \rightarrow \frac{dG}{dtdE} = \frac{dG}{dEdt}.$$

The solution to the following system of differential equations is the sensitivity of each state variable to each enzyme (AA = α-amylase, GA = glucoamylase, and AG = α-glucosidase):

$$\begin{aligned}
\frac{dS_{available}}{dAAdt} &= -s_1 - s_2 - s_3 - s_4, \\
\frac{dS_{available}}{dGAdt} &= -d_1 - d_2 - d_3 - d_4, \\
\frac{dS_{available}}{dAGdt} &= -d_1 - d_2 - d_3 - d_4, \\
\frac{dD}{dAAdt} &= s_1 - s_5 - s_6 - s_7 - s_8, \\
\frac{dD}{dGAdt} &= d_1 - d_5 - d_6 - d_7 - d_8, \\
\frac{dD}{dAGdt} &= d_1 - d_5 - d_6 - d_7 - d_8, \\
\frac{dM}{dAAdt} &= s_3 + s_6 + d_{10} + d_{11} - d_9 - d_{12}, \\
\frac{dM}{dGAdt} &= d_3 + d_6 + s_{10} + d_{11} - s_9 - d_{12}, \\
\frac{dM}{dAGdt} &= d_3 + d_6 + d_{10} + s_{11} - d_9 - s_{12}, \\
\frac{dM}{dAAdt} &= s_2 + s_5 - d_{10} - d_{11},
\end{aligned}$$

$$\begin{aligned}
\frac{dM}{dGAdt} &= d_2 + d_5 - s_{10} - d_{11}, \\
\frac{dM}{dAGdt} &= d_2 + d_5 - d_{10} - s_{11}, \\
\frac{dG}{dAAdt} &= s_4 + s_7 + d_8 + d_9 + d_{10} + d_{11} + d_{12}, \\
\frac{dG}{dGAdt} &= d_4 + d_7 + s_8 + s_9 + s_{10} + d_{11} + d_{12}, \\
\frac{dG}{dAGdt} &= d_4 + d_7 + d_8 + d_9 + d_{10} + s_{11} + s_{12}.
\end{aligned}$$

The state variable $S_{unavailable}$ is not dependent on enzyme concentration at all and is therefore excluded from this analysis. The system was solved together with the original model to produce the following results: Figure 2.7 shows that the overall glucose production is sensitive to both α-amylase and glucoamylase, but not sensitive to α-glucosidase. Thus, adding additional α-glucosidase to the amount already naturally occurring in corn would not have a significant effect on glucose production and should not be considered.

2.5 Simplified Model

We now consider a simplified model as shown in Figure 2.8, where we combine the first three compartments *Starch Granule, Available Starch* and *Dextrins* into one compartment *Starch*. The reason for doing this is that substrates in these three compartments actually are the same material with different sizes of molecules.

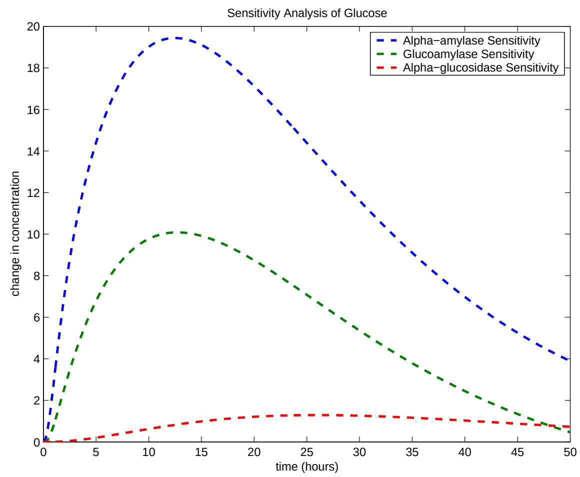


Figure 2.7: Sensitivity of the glucose concentration to the enzymes.

In addition, the rates between these compartments cannot be measured in experiments. After combining them into one compartment, we no longer need to consider intermediate reactions. Furthermore, since certain enzyme mixture reactions do not produce any maltotriose, we can exclude the *Maltotriose* compartment from our model in these cases.

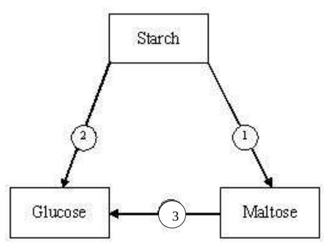


Figure 2.8: Diagram of the enzyme reactions in the simple model.

With this simplified model, we get a simpler ODE system:

$$\begin{aligned}
\frac{d[Starch]}{dt} &= -\widetilde{r}_1 - \widetilde{r}_2, \\
\frac{d[Maltose]}{dt} &= \widetilde{r}_1 - \widetilde{r}_3, \\
\frac{d[Glucose]}{dt} &= \widetilde{r}_2 + \widetilde{r}_3,
\end{aligned} \tag{2.7}$$

where rates are modeled in the same way as before, using the Michaelis-Menten kinetics.

Since the original model has simulated the reaction from maltose to glucose, we combine the reaction rates r_9 and r_{10}, and adopt the same kinetic parameters.

$$\begin{aligned}
\widetilde{r}_1 &= \frac{\widetilde{k}_{cat1}[Starch][\alpha amylase]}{\widetilde{K}_{m1} + [Starch]}, \\
\widetilde{r}_2 &= 0.05\widetilde{r}_1 + \frac{\widetilde{k}_{cat2}[Starch][gluco]}{\widetilde{K}_{m2} + [Starch]}, \\
\widetilde{r}_3 &= \frac{\widetilde{k}_{cat3}[Mt][gluco]}{\widetilde{K}_{m3} + Mt} + \frac{\widetilde{k}_{cat4}[Mt][agluco]}{\widetilde{K}_{m4} + Mt},
\end{aligned} \tag{2.8}$$

where $\widetilde{k}_{cat3} = k_{cat9}, \widetilde{K}_{m3} = K_{m9}, \widetilde{k}_{cat4} = K_{cat12}, \widetilde{K}_{m4} = K_{m12}$, and $\widetilde{k}_{cat1}, \widetilde{k}_{m1}, \widetilde{k}_{cat2}, \widetilde{k}_{m2}$ are unknown.

To compare the simulated results with the experimental data, we prefer the experimental data to be measured at equal time interval points, which is not the case in experiments [3]. Thus we begin with fitting the data of glucose to simplify computation. Since the curve of production of glucose is pretty smooth, we use a quadratical function to fit it [3].

Given specific values of kinetic parameters, we can use an ODE solver, such as *ode45* in *Matlab 7*, to obtain the transient values for starch, maltose and glucose. We fit the simulated glucose values to the experimental data so that

$$\sum_t (Glucose_{simulated} - Glucose_{measured})^2,$$

is minimized. Using the optimization toolbox in *Matlab 7* we can solve the parameter values iteratively with the initial guess close to the values used in the full model. The kinetic parameters we obtained from optimization are

$$\begin{aligned}
\widetilde{k}_{cat1} &= 0.23074 \quad \widetilde{K}_{m1} = 0.30761 \\
\widetilde{k}_{cat2} &= 42.419 \quad\ \ \widetilde{K}_{m2} = 2.1747.
\end{aligned}$$

Figure 2.9 shows the ODE solution of Glucose with the optimized parameters and the experimental data, where the solid line represents the ODE solution of Glucose and the circle represents the experimental data. Figure 2.10 shows the depletion of starch and the transient production of maltose and glucose.

2.6 Different Approach - S-Systems

In current biochemical research, S-systems have been applied extensively because of their elegant properties in mathematical analysis and computation [5]. Each equation in an S-system has only two terms; influx and outflux:

$$\dot{X}_i = \alpha_i \prod_{j=1}^{n+m} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{n+m} X_j^{h_{ij}},$$

where α_i and β_i are non-negative rate coefficients, and g_{ij}, h_{ij} are kinetic orders. Additionally, an S-system can be derived from a GMA(general mass action) system, which has the mathematical structure:

$$\dot{X}_i = \sum_j V_{ij}^{+} - \sum_j V_{ij}^{-}. \tag{2.9}$$

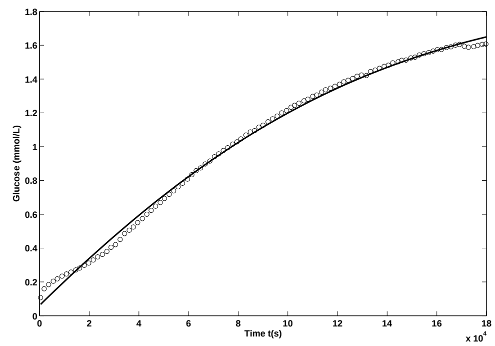


Figure 2.9: ODE solution of Glucose with the optimized parameters (solid line) vs. experimental data (circles).

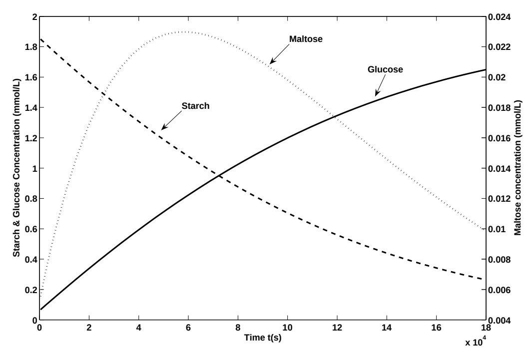


Figure 2.10: Depletion of starch and transient production of maltose and glucose.

Every term in (2.9) is a power function of substrate or enzyme concentrations, which is a general form for analysis and computation. By combining all positive and negative terms, we can get an S-system from a GMA system. The rate coefficients and kinetic orders of the S-system can be obtained using several different approaches. One approach uses optimization techniques to estimate the unknown parameters based on dynamic data. Another approach uses steady state data to estimate the parameters [5].

Here we give S-system equations of the simplified model as follows:

$$\begin{aligned}
\frac{d[Starch]}{dt} &= -\alpha_1 [Starch]^{h_1} [\alpha - amylase]^{h_2} [gluco]^{h_3}, \\
\frac{d[Maltose]}{dt} &= \alpha_2 [Starch]^{h_2} [\alpha - amylase]^{h_3} [gluco]^{h_4} - \alpha_3 [Starch]^{h_5} [gluco]^{h_6} [\alpha - gluco]^{h_7}, \\
\frac{d[Glucose]}{dt} &= \alpha_4 [Starch]^{h_8} [\alpha - amylase]^{h_9} [gluco]^{h_{10}} [\alpha - gluco]^{h_{11}},
\end{aligned} \tag{2.10}$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4, h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9, h_{10}, h_{11}$ are unknown parameters. Then we use the data of

Maltose and Glucose to find out all unknown parameters. Finally, we can solve the above ordinary differentiate equation system to model the relation between the yield of Glucose and the concentration of Enzymes.

2.7 Conclusions

We have succesfully produced a working model of the glucose production from starch that gives 98% yield of glucose in 50 hours. With the parameters shown in Figures 2.3 and 2.4 it gives results which correspond to the actual data collected in the industrial process. We have also performed sensitivity analyis on this model. The results of the sensitivity analysis tell us that there is no advantage in adding additional amounts of α-glucosidase, the naturally occuring enzyme. However, the overall glucose production is highly sensitive to both α-amylase and glucoamylase. One of the biggest problems we encountered was a lack of reliable data. To make any further conclusions we would need more accurate and complete data. This would allow us to check the accuracy of the model and make any changes. Only then we could investigate further the dependence of the process on the two enzymes. Another thing we would like to consider in the future is looking into a different approach. One suggestion would be looking into S-system representation of the reactions.

Bibliography

[1] Brandam, C., Meyer, X. M., Proth, J., Strehaiano, P. and Pingaud, H. (2003) An Original Kinetic Model for the Enzymatic Hydrolysis of Starch During Mashing, *Biochemical Engineering Journal*, 13: 43-52.

[2] Hindmarsh, A. C. (1972) *Linear Multistep Methods For Ordinary Differential Equations: Method Formulations, Stability, And The Methods Of Nordsieck And Gear*, Ucrl-51186 Rev. 1, Lawrence Livermore Lab.

[3] Schittkowski, K. (2002) *Numerical Data Fitting in Dynamical Systems: A Practical Introduction with Applications and Software*, Kluwer Academic Publishers.

[4] Takeda, Y., Shitaozono, T. and Hizukuri,S. (1988) Molecular Structure of Corn Starch, *Starch*, 2:51-54.

[5] Voit, E. O. (2000) *Computational Analysis of Biochemical Systems*, Cambridge University Press, Cambridge

[6] http://www.uea.ac.uk/cap/carbohydrate/projects/amylopectin_biosynthesis.htm

2.8 Appendix

sanalysis.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Creates and plots the model using kinetic values from the three %
% enzyme vectors.  Uses static starch starting value (245 g/L).   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Convert starch from g/L to mMol/L
starch = 245 / 1620;

%Set initial conditions
starch_unavail = 1*starch;
starch_avail = 0*starch;
dextrins = 1e-10;
maltose = 1e-10;
maltotriose = 1e-10;
glucose = 1e-10;

v0 = [starch_unavail; starch_avail; dextrins; maltose; maltotriose; glucose; zeros(18,1)];

%Use ODE solver to solve model
[t, v] = ode45(@senseqs, [0 180000], v0, ['Reltol',1e-4,'Abstol',1e-4], aa, ga, ag);

dstarch_unavaildAA = v(:, 7);
dstarch_availdAA = v(:, 8);
ddextrinsdAA = v(:, 9);
dmaltosedAA = v(:, 10);
dmaltotriosedAA = v(:, 11);
dglucosedAA = v(:, 12);
```

```
dstarch_unavaildGA = v(:, 13);
dstarch_availdGA = v(:, 14);
ddextrinsdGA = v(:, 15);
dmaltosedGA = v(:, 16);
dmaltotriosedGA = v(:, 17);
dglucosedGA = v(:, 18);

dstarch_unavaildAG = v(:, 19);
dstarch_availdAG = v(:, 20);
ddextrinsdAG = v(:, 21);
dmaltosedAG = v(:, 22);
dmaltotriosedAG = v(:, 23);
dglucosedAG = v(:, 24);

t = t ./ 3600;

figure(1)
plot(t, dstarch_unavaildAA, '--', t, dstarch_unavaildGA, '--', t, dstarch_unavaildAG, '--', 'LineWidth'
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Starch Unavailable');
xlabel('time (hours)');
ylabel('change in concentration');

figure(2)
plot(t, dstarch_availdAA, '--', t, dstarch_availdGA, '--', t, dstarch_availdAG, '--', 'LineWidth', 2)
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Starch Available');
xlabel('time (hours)');
ylabel('change in concentration');
```

```
figure(3)
plot(t, ddextrinsdAA, '--', t, ddextrinsdGA, '--', t, ddextrinsdAG, '--', 'LineWidth', 2)
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Dextrins');
xlabel('time (hours)');
ylabel('change in concentration');

figure(4)
plot(t, dmaltosedAA, '--', t, dmaltosedGA, '--', t, dmaltosedAG, '--', 'LineWidth', 2)
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Maltose');
xlabel('time (hours)');
ylabel('change in concentration');

figure(5)
plot(t, dmaltotriosedAA, '--', t, dmaltotriosedGA, '--', t, dmaltotriosedAG, '--', 'LineWidth', 2)
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Maltotriose');
xlabel('time (hours)');
ylabel('change in concentration')

figure(6)
plot(t, dglucosedAA, '--', t, dglucosedGA, '--', t, dglucosedAG, '--', 'LineWidth', 2)
legend('Alpha-amylase Sensitivity', 'Glucoamylase Sensitivity', 'Alpha-glucosidase Sensitivity');
title('Sensitivity Analysis of Glucose');
xlabel('time (hours)');
ylabel('change in concentration');
```

senseqs.m

```
function dv = sensitiv(t, v, alpha_amylase, glucoamylase, glucosidase)

%ORIGINAL MODEL
starch_unavail = v(1);
starch_avail = v(2);
dextrins = v(3);
maltose = v(4);
maltotriose = v(5);
glucose = v(6);
r0 = .005*starch_unavail/(250 + starch_unavail);
r1 = MM(alpha_amylase(2), alpha_amylase(3), alpha_amylase(1), starch_avail);
r2 = MM(alpha_amylase(4), alpha_amylase(5), alpha_amylase(1), starch_avail);
r3 = MM(alpha_amylase(6), alpha_amylase(7), alpha_amylase(1), starch_avail);
r4 = .05 * r3;
r5 = MM(alpha_amylase(10), alpha_amylase(11), alpha_amylase(1), dextrins);
r6 = MM(alpha_amylase(12), alpha_amylase(13), alpha_amylase(1), dextrins);
r7 = .05 * r6;
r8 = MM(glucoamylase(2), glucoamylase(3), glucoamylase(1), dextrins);
r9 = MM(glucoamylase(4), glucoamylase(5), glucoamylase(1), maltose);
r10 = MM(glucoamylase(6), glucoamylase(7), glucoamylase(1), maltotriose);
r11 = MM(glucosidase(2), glucosidase(3), glucosidase(1), maltotriose);
r12 = MM(glucosidase(4), glucosidase(5), glucosidase(1), maltose);

dv = zeros(24, 1);
dv(1) = -r0;
dv(2) = r0 - r1 - r2 - r3 - r4;
dv(3) = r1 - r5 - r6 - r7 - r8;
dv(4) = r3 + r6 + r10 + r11 - r9 - r12;
dv(5) = r2 + r5 - r10 - r11;
```

```
dv(6) = r4 + r7 + r8 + r9 + r10 + r11 +r12;


%ALPHA AMYLASE ANALYSIS
dstarch_unavail_dAA = v(7);
dstarch_avail_dAA = v(8);
ddextrins_dAA = v(9);
dmaltose_dAA = v(10);
dmaltotriose_dAA = v(11);
dglucose_dAA = v(12);

dr0_dAA = .005*dstarch_unavail_dAA*(1/(250+starch_unavail)-starch_unavail/(250+starch_unavail)^2);
ds1_dAA = DS(alpha_amylase(2), alpha_amylase(3), alpha_amylase(1), starch_avail, dstarch_avail_dAA);
ds2_dAA = DS(alpha_amylase(4), alpha_amylase(5), alpha_amylase(1), starch_avail, dstarch_avail_dAA);
ds3_dAA = DS(alpha_amylase(6), alpha_amylase(7), alpha_amylase(1), starch_avail, dstarch_avail_dAA);
ds4_dAA = .05 * ds3_dAA;
ds5_dAA = DS(alpha_amylase(10), alpha_amylase(11), alpha_amylase(1), dextrins, ddextrins_dAA);
ds6_dAA = DS(alpha_amylase(12), alpha_amylase(13), alpha_amylase(1), dextrins, ddextrins_dAA);
ds7_dAA = .05 * ds6_dAA;
dn8_dAA = DN(glucoamylase(2), glucoamylase(3), glucoamylase(1), dextrins, ddextrins_dAA);
dn9_dAA = DN(glucoamylase(4), glucoamylase(5), glucoamylase(1), maltose, dmaltose_dAA);
dn10_dAA = DN(glucoamylase(6), glucoamylase(7), glucoamylase(1), maltotriose, dmaltotriose_dAA);
dn11_dAA = DN(glucosidase(2), glucosidase(3), glucosidase(1), maltotriose, dmaltotriose_dAA);
dn12_dAA = DN(glucosidase(4), glucosidase(5), glucosidase(1), maltose, dmaltose_dAA);

dv(7) = -dr0_dAA;
dv(8) = dr0_dAA - ds1_dAA - ds2_dAA - ds3_dAA - ds4_dAA;
dv(9) = ds1_dAA - ds5_dAA - ds6_dAA - ds7_dAA - dn8_dAA;
dv(10) = ds3_dAA + ds6_dAA + dn10_dAA + dn11_dAA - dn9_dAA - dn12_dAA;
dv(11) = ds2_dAA + ds5_dAA - dn10_dAA - dn11_dAA;
dv(12) = ds4_dAA + ds7_dAA + dn8_dAA + dn9_dAA + dn10_dAA + dn11_dAA + dn12_dAA;
```

```
%GLUCOAMYLASE ANALYSIS
dstarch_unavail_dGA = v(13);
dstarch_avail_dGA = v(14);
ddextrins_dGA = v(15);
dmaltose_dGA = v(16);
dmaltotriose_dGA = v(17);
dglucose_dGA = v(18);

dr0_dGA = .005*dstarch_unavail_dAA*(1/(250+starch_unavail)-starch_unavail/(250+starch_unavail)^2);
dn1_dGA = DN(alpha_amylase(2), alpha_amylase(3), alpha_amylase(1), starch_avail, dstarch_avail_dGA);
dn2_dGA = DN(alpha_amylase(4), alpha_amylase(5), alpha_amylase(1), starch_avail, dstarch_avail_dGA);
dn3_dGA = DN(alpha_amylase(6), alpha_amylase(7), alpha_amylase(1), starch_avail, dstarch_avail_dGA);
dn4_dGA = .05 * dn3_dGA;
dn5_dGA = DN(alpha_amylase(10), alpha_amylase(11), alpha_amylase(1), dextrins, ddextrins_dGA);
dn6_dGA = DN(alpha_amylase(12), alpha_amylase(13), alpha_amylase(1), dextrins, ddextrins_dGA);
dn7_dGA = .05 * dn6_dGA;
ds8_dGA = DS(glucoamylase(2), glucoamylase(3), glucoamylase(1), dextrins, ddextrins_dGA);
ds9_dGA = DS(glucoamylase(4), glucoamylase(5), glucoamylase(1), maltose, dmaltose_dGA);
ds10_dGA = DS(glucoamylase(6), glucoamylase(7), glucoamylase(1), maltotriose, dmaltotriose_dGA);
dn11_dGA = DN(glucosidase(2), glucosidase(3), glucosidase(1), maltotriose, dmaltotriose_dGA);
dn12_dGA = DN(glucosidase(4), glucosidase(5), glucosidase(1), maltose, dmaltose_dGA);

dv(13) = -dr0_dGA;
dv(14) = dr0_dGA - dn1_dGA - dn2_dGA - dn3_dGA - dn4_dGA;
dv(15) = dn1_dGA - dn5_dGA - dn6_dGA - dn7_dGA - ds8_dGA;
dv(16) = dn3_dGA + dn6_dGA + ds10_dGA + dn11_dGA - ds9_dGA - dn12_dGA;
dv(17) = dn2_dGA + dn5_dGA - ds10_dGA - dn11_dGA;
dv(18) = dn4_dGA + dn7_dGA + ds8_dGA + ds9_dGA + ds10_dGA + dn11_dGA + dn12_dGA;
```

```
%ALPHA GLUCOSIDASE ANALYSIS
dstarch_unavail_dAG = v(19);
dstarch_avail_dAG = v(20);
ddextrins_dAG = v(21);
dmaltose_dAG = v(22);
dmaltotriose_dAG = v(23);
dglucose_dAG = v(24);

dr0_dAG = .005*dstarch_unavail_dAG*(1/(250+starch_unavail)-starch_unavail/(250+starch_unavail)^2);
dn1_dAG = DN(alpha_amylase(2), alpha_amylase(3), alpha_amylase(1), starch_avail, dstarch_avail_dAG);
dn2_dAG = DN(alpha_amylase(4), alpha_amylase(5), alpha_amylase(1), starch_avail, dstarch_avail_dAG);
dn3_dAG = DN(alpha_amylase(6), alpha_amylase(7), alpha_amylase(1), starch_avail, dstarch_avail_dAG);
dn4_dAG = .05 * dn3_dAG;
dn5_dAG = DN(alpha_amylase(10), alpha_amylase(11), alpha_amylase(1), dextrins, ddextrins_dAG);
dn6_dAG = DN(alpha_amylase(12), alpha_amylase(13), alpha_amylase(1), dextrins, ddextrins_dAG);
dn7_dAG = .05 * dn6_dAG;
dn8_dAG = DN(glucoamylase(2), glucoamylase(3), glucoamylase(1), dextrins, ddextrins_dAG);
dn9_dAG = DN(glucoamylase(4), glucoamylase(5), glucoamylase(1), maltose, dmaltose_dAG);
dn10_dAG = DN(glucoamylase(6), glucoamylase(7), glucoamylase(1), maltotriose, dmaltotriose_dAG);
ds11_dAG = DS(glucosidase(2), glucosidase(3), glucosidase(1), maltotriose, dmaltotriose_dAG);
ds12_dAG = DS(glucosidase(4), glucosidase(5), glucosidase(1), maltose, dmaltose_dAG);

dv(19) = -dr0_dAA;
dv(20) = dr0_dAG - dn1_dAG - dn2_dAG - dn3_dAG - dn4_dAG;
dv(21) = dn1_dAG - dn5_dAG - dn6_dAG - dn7_dAG - dn8_dAG;
dv(22) = dn3_dAG + dn6_dAG + dn10_dAG + ds11_dAG - dn9_dAG - ds12_dAG;
dv(23) = dn2_dAG + dn5_dAG - dn10_dAG - ds11_dAG;
dv(24) = dn4_dAG + dn7_dAG + dn8_dAG + dn9_dAG + dn10_dAG + ds11_dAG + ds12_dAG;
function drde = DS(kcat, km, enzyme, substrate, dsubdenz)
drde = kcat*((substrate+enzyme*dsubdenz)/(km+substrate)-substrate*enzyme*dsubdenz/(km+substrate)^2);
```

```
function drde = DN(kcat, km, enzyme, substrate, dsubdenz)
drde = kcat*(enzyme*dsubdenz*km)/(km+substrate)^2;

function r = MM(kcat, km, enzyme, substrate)
r = (kcat*(enzyme)*substrate)/(km+substrate);
```

diffeqs.m

```
function dv = diffeqs(t, v, alpha_amylase, glucoamylase, glucosidase)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This function contains all the differential equations in vector %
% form.  The first two inputs are required for the ODE solver and %
% the last three are used to input the constants for the model.   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Extract current values
starch_unavail = v(1);
starch_avail = v(2);
dextrins = v(3);
maltose = v(4);
maltotriose = v(5);
glucose = v(6);

%Set all rate values;
r0 = .005*starch_unavail / (250+starch_unavail);
r1 = MM(alpha_amylase(2), alpha_amylase(3), alpha_amylase(1), starch_avail);
r2 = MM(alpha_amylase(4), alpha_amylase(5), alpha_amylase(1), starch_avail);
r3 = MM(alpha_amylase(6), alpha_amylase(7), alpha_amylase(1), starch_avail);
r4 = .05 * r3;
r5 = MM(alpha_amylase(10), alpha_amylase(11), alpha_amylase(1), dextrins);
r6 = MM(alpha_amylase(12), alpha_amylase(13), alpha_amylase(1), dextrins);
r7 = .05 * r6;
r8 = MM(glucoamylase(2), glucoamylase(3), glucoamylase(1), dextrins);
r9 = MM(glucoamylase(4), glucoamylase(5), glucoamylase(1), maltose);
r10 = MM(glucoamylase(6), glucoamylase(7), glucoamylase(1), maltotriose);
r11 = MM(glucosidase(2), glucosidase(3), glucosidase(1), maltotriose);
```

```
r12 = MM(glucosidase(4), glucosidase(5), glucosidase(1), maltose);

%Compute differential equations
dv = zeros(6, 1);
dv(1) = -r0;
dv(2) = r0 - r1 - r2 - r3 - r4;
dv(3) = r1 - r5 - r6 - r7 - r8;
dv(4) = r3 + r6 + r10 + r11 - r9 - r12;
dv(5) = r2 + r5 - r10 - r11;
dv(6) = r4 + r7 + r8 + r9 + r10 + r11 +r12;
```

initialize.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Creates three vectors for the three enzymes.  The %
% vectors have the concentration first and then the %
% various kcat and km values afterwards.            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aa = zeros(15, 1);
ga = zeros(7, 1);
ag = zeros(5, 1);

%ALPHA-AMYLASE (JA126)
    %Concentration
    aa(1) = 9.8e-5;
    %Starch (available) to Dextrins reaction (1)
    aa(2) = 869/60;
    aa(3) = 37.17;
    %Starch (available) to Maltotriose reaction (2)
    aa(4) = 9.62/60;
    aa(5) = 0.458;
    %Starch (available) to Maltose reaction (3)
    aa(6) = 21.72/60;
    aa(7) = 0.09003;
    %Starch (available) to Glucose reaction (4)
    aa(8) = 0;
    aa(9) = 0;
    %Dextrins to Maltotriose reaction (5)
    aa(10) = 9.62/60;
    aa(11) = 0.458;
    %Dextrins to Maltose reaction (6)
```

```
    aa(12) = 21.72/60;
    aa(13) = 0.09003;
    %Dextrins to Glucose reaction (7)
    aa(14) = 0;
    aa(15) = 0;

%GLUCOAMYLASE (A. niger)
    %Concentration
    ga(1) = 1e-4;
    %Dextrins to Glucose reaction (8)
    ga(2) = 51.13/60;
    ga(3) = .25;
    %Maltose to Glucose reaction (9)
    ga(4) = 7.6/60;
    ga(5) = 1210;
    %Maltotriose to Maltose/Glucose reaction (10)
    ga(6) = 16.49/60;
    ga(7) = 360;

%ALPHA-GLUCOSIDASE
    %Concentration
    ag(1) = 3.02e-2;
    %Maltotriose to Maltose/Glucose reaction (11)
    ag(2) = 64/60;
    ag(3) = 1300;
    %Maltose to Glucose reaction (12)
    ag(4) = 64/60;
    ag(5) = 1300;
```

MM.m

```
function r = MM(kcat, km, enzyme, substrate)
r = (kcat*(enzyme)*substrate)/(km+substrate);
```

model.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Creates and plots the model using kinetic values from the three %
% enzyme vectors.  Uses static starch starting value (245 g/L).   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Convert starch from g/L to mMol/L
starch = 245 / 1620;

%Set initial conditions
starch_unavail = 1*starch;
starch_avail = 0*starch;
dextrins = 1e-10;
maltose = 1e-10;
maltotriose = 1e-10;
glucose = 1e-10;

v0 = [starch_unavail; starch_avail; dextrins; maltose; maltotriose; glucose];

%Use ODE solver to solve model
[t, v] = ode45(@diffeqs, [0 180000], v0, ['Reltol',1e-4,'Abstol',1e-4], aa, ga, ag);

starch_unavail = v(:, 1);
starch_avail = v(:, 2);
dextrins = v(:, 3);
maltose = v(:, 4);
maltotriose = v(:, 5);
glucose = v(:, 6);

%Compute total starch in the system
starch_total = starch_unavail + starch_avail + dextrins;
```

```
%Compute rates
r0 = zeros(length(t), 1);
r1 = zeros(length(t), 1);
r2 = zeros(length(t), 1);
r3 = zeros(length(t), 1);
r4 = zeros(length(t), 1);
r5 = zeros(length(t), 1);
r6 = zeros(length(t), 1);
r7 = zeros(length(t), 1);
r8 = zeros(length(t), 1);
r9 = zeros(length(t), 1);
r10 = zeros(length(t), 1);
r11 = zeros(length(t), 1);
r12 = zeros(length(t), 1);
for i = 1:length(t)
r0(i) = .01*starch_unavail(i) / (10+starch_unavail(i));
r1(i) = MM(aa(2), aa(3), aa(1), starch_avail(i));
r2(i) = MM(aa(4), aa(5), aa(1), starch_avail(i));
r3(i) = MM(aa(6), aa(7), aa(1), starch_avail(i));
r4(i) = .05 * r3(i);
r5(i) = MM(aa(10), aa(11), aa(1), dextrins(i));
r6(i) = MM(aa(12), aa(13), aa(1), dextrins(i));
r7(i) = .05 * r6(i);
r8(i) = MM(ga(2), ga(3), ga(1), dextrins(i));
r9(i) = MM(ga(4), ga(5), ga(1), maltose(i));
r10(i) = MM(ga(6), ga(7), ga(1), maltotriose(i));
r11(i) = MM(ag(2), ag(3), ag(1), maltotriose(i));
r12(i) = MM(ag(4), ag(5), ag(1), maltose(i));
end
```

```
%Change timescale from seconds to hours
t = t ./ 3600;

%Plot results
figure(1);
P=plot(t, starch_total, '--', t, starch_unavail, '--', t, starch_avail, '--', t, dextrins, '--', t, mal
legend('Total Starch', 'Starch (unavailable)', 'Starch (available)', 'Dextrins', 'Maltose', 'Maltotrios
title('Glucose Production Model');
xlabel('time (hours)');
ylabel('concentration (mM/L)');

figure(2);
plot(t, r0, '--', t, r1, '--', t, r2, '--', t, r3, '--', t, r4, '--', t, r5, '--', t, r6, '--', t, r7,
legend('r0 S_U \rightarrow S_A', 'r1 S_A \rightarrow D [AA]', 'r2 S_A \rightarrow MT [AA]', 'r3 S_A \ri
title('Reaction Rates for Glucose Model');
xlabel('time (hours)');
ylabel('rate ([mM/L]/s');
```

Chapter 3

Control of Dynamic System Response with Directional Forcing Sources

PROBLEM 3: CONTROL OF DYNAMIC SYSTEM RESPONSE WITH DIRECTIONAL FORCING SOURCES

Robert Benim[1], Mike Bowman[2], John David[3], John Gonzalez[4], Emek Köse[5], Eduardo Osorio[6]

Problem Presenter:
Anna Howard, Ph.D.
Lord Corporation

Faculty Consultants:
Dr. Hien Tran & Dr. Dmitry Zenkov
North Carolina State University

Abstract

The attenuation of helicopter vibrations is essential for increasing their fuel efficiency and mobility. A control strategy involving sensors and actuators is currently being developed. In this note we consider the problem of finding the optimal positions of actuators and optimal angles of force actuation. We discuss analytical and numerical methods for solving this problem. The results are based on a model for sensor-actuator communication developed by Lord Corporation.

3.1 Introduction and Motivation

Lord Corporation is a privately held firm specializing in the development and manufacture of automotive and industrial adhesives, flight-critical elastomer components for use by aerospace firms, and MR-Fluid technology components, among other things. One of the interests at Lord Corporation is the development of active systems for helicopters to control vibration and noise. Adaptive control of dynamic system response to disturbances using actuators and sensors is a fairly well understood problem. Examples of such systems are used for vibration and noise reduction on various aircraft. However, current system models do not incorporate the dynamic ability to adjust the direction of the actuator forces within the aircraft. The purpose of this project is to study the mathematical properties of an expanded mathematical model which includes the capability to adjust the actuator force directions after installation.

3.2 Physical and Mathematical Background

The physical system being modeled is an active control system to dynamically control vibration for a helicopter with a rotor with N blades spinning at an angular frequency ω. Inside the helicopter are n sensors (e.g.,

[1]University of Portland
[2]Montana State University
[3]North Carolina State University
[4]Northeastern University
[5]Drexel University
[6]Rutgers University

accelerometers) $e_1, \cdots, e_n$ and m actuators $u_1, \cdots, u_m$. Assuming no ability to adjust the directions of the actuator forces, the system is described in the frequency domain at $N\omega$ by the following equation:

$$e = Cu + d, \tag{3.1}$$

where e is an $n \times 1$ complex vector of sensor signals, u is an $m \times 1$ complex vector of actuator signals, C is an $n \times m$ complex matrix representing the transfer function between sensors and actuators, and d is an $n \times 1$ complex vector representing the disturbance as measured by the sensors at $N\omega$. The corresponding optimal control problem is to minimize the following cost function:

$$J = e^* e, \tag{3.2}$$

where * denotes the conjugate transpose of a matrix. In other words, the goal is to find the u which minimizes J. The optimal control problem just described is easy to solve analytically. It has the following solution:

$$u_{opt} = -(C^* C)^{-1} C^* d = -C^+ d, \tag{3.3}$$

where C^+ denotes the pseudoinverse of C, C is assumed to be full rank, and $n < m$. Also, when $n = m$, $u_{opt} = -C^{-1} d$. However, the disturbance d is not known, so in practice the problem is solved with an efficient numerical method (e.g., gradient descent).

The expanded model includes a set of m angles, $\theta_1, \cdots, \theta_m$ which allow the directions of the actuator forces to be adjusted (i.e., rotated in a plane). The addition of the actuator adjustment angles to the problem gives rise to a new mathematical formulation. The basic equation is:

$$e = Cf + d, \tag{3.4}$$

where

$$f = T(\theta) u, \tag{3.5}$$

and

$$T(\theta) = \left[\frac{diag\{\cos(\theta_i)\}}{diag\{\sin(\theta_i)\}} \right]. \tag{3.6}$$

We still have e as an $n \times 1$ complex vector of sensor signals, u as an $m \times 1$ complex vector of actuator signals, and d as an $n \times 1$ complex vector representing the disturbance. However, now C is a $n \times 2m$ complex matrix of transfer functions, f is a $2m \times 1$ complex vector of actuator signals, θ is an $m \times 1$ real vector of m actuator adjustment angles, and $T(\theta)$ is a $2m \times m$ real matrix as described by (3.6). The optimal control problem is still to minimize $J = e^* e$. However, we will now be seeking a set of optimal u_i's and a set of optimal θ_i's, which we will denote by the $m \times 1$ vectors u_{opt} and θ_{opt}. Expressing J as a function of u and θ, this is equivalent to finding u_{opt} and θ_{opt} which minimize

$$J(u, \theta) = (CTu + d)^* (CTu + d) = (u^* T^* C^* + d^*)(CTu + d). \tag{3.7}$$

where $T = T(\theta)$ is described by (3.6). This optimal control problem is more difficult to solve, and it is the focus of this project.

3.3 Project Goals

Given the optimal control problem described in the preceding section, the goals of this project are as follows:

1. To study the existence and uniqueness properties of u_{opt} and θ_{opt} and to determine their analytic formulas.

2. To develop an efficient numerical algorithm to solve the problem in practical settings and to determine the conditions for convergence of such an algorithm.

3.4 Solutions

First in section 4.1 we will discuss the problem of finding analytic expressions for a minimizer of (3.7). In section 4.3 we prove existence of a global minimum of (3.7). In section 4.4 we give equivalent formulations of the minimization problem that are useful for numerical methods. Finally in section 4.5 we discuss the numerical methods that can be used.

3.4.1 Analytical Solutions

Finding Critical Values of J

The first approach to minimizing (3.7) was to try to find analytic formulas for the critical values. For this we apply the first derivative test to compute critical values. These derivatives were also used for our numerical methods. The partial derivative with respect to u is given by

$$\frac{\partial J}{\partial u} = (d^* + u^* T^* C^*) CT. \tag{3.8}$$

We can also break u into vectors of the real and imaginary parts. Expressing the vector u as $u = u_R + u_I$ where u_R and u_I denote the real and imaginary parts of u, respectively, and u_I includes i, we get the following formulas for the derivatives with respect to the real and imaginary parts:

$$\frac{\partial J}{\partial u_R} = A^t + (B + B^t) u_R + (B - B^t) u_I + A^*, \tag{3.9}$$

and

$$\frac{\partial J}{\partial u_I} = A^t - (B + B^t) u_I - (B - B^t) u_R - A^*, \tag{3.10}$$

where t denotes transpose, and where

$$A = d^* C T u, \tag{3.11}$$

and

$$B = u^* T^t C^* C T u. \tag{3.12}$$

For the derivative with respect to θ,

$$\frac{\partial J}{\partial \theta} = \left(\frac{\partial J}{\partial \theta_1} \cdots \frac{\partial J}{\partial \theta_m} \right), \tag{3.13}$$

where

$$\frac{\partial J}{\partial \theta_k} = u^* T^* C^* C \left(\frac{\partial T}{\partial \theta_k} \right) u + d^* C \left(\frac{\partial T}{\partial \theta_k} \right) u + u^* \left(\frac{\partial T}{\partial \theta_k} \right)^* C^* C T u + u^* \left(\frac{\partial T}{\partial \theta_k} \right)^* C^* d, \tag{3.14}$$

and

$$\frac{\partial T}{\partial \theta_k} = \left[\frac{\partial T_{i,j}}{\partial \theta_k} \right]_{i,j}, \tag{3.15}$$

i.e. (3.15) is the matrix obtained by differentiating each entry in T with respect to θ_k. Formula (3.14) is derived by applying the product rule to (3.7) to get

$$\frac{\partial J}{\partial \theta_k} = [u^* T^* C^* + d^*] \left[\frac{\partial}{\partial \theta_k} (CTu + d) \right] + \left[\frac{\partial}{\partial \theta_k} (u^* T^* C^* + d^*) \right] [CTu + d], \tag{3.16}$$

which is equivalent to

$$\frac{\partial J}{\partial \theta_k} = [u^* T^* C^* + d^*] \left[C \left(\frac{\partial T}{\partial \theta_k} \right) u \right] + \left[u^* \left(\frac{\partial T}{\partial \theta_k} \right)^* C^* \right] [CTu + d]. \tag{3.17}$$

Multiplying out the terms in (3.17) gives (3.14).

By setting (3.8) and each component of (3.13) equal to zero, or (3.9), (3.10) and each component of (3.13) equal to zero, we get systems of equations in terms of u and θ to be solved to determine the critical values of J. We could hope to solve for the critical values and then plug them into (3.7) to determine the minimum of J. However, since the system of equations is highly nonlinear it is difficult to find closed-form solutions except in very small cases (e.g., $n = 2, m = 1$). Therefore, we have determined that this approach is not good enough for solving this problem in the typical settings that would be encountered by Lord Corporation.

Change of Variables

Our cost function $J = e^* e$, when viewed as a function of f, is a quadratic function of $2m$ complex variables (equivalently $4m$ real variables) and, thus, it may be viewed as a paraboloid in $\mathbb{R}^{4m+1}$. In general, it may have ellipsoidal cross-sections rather than circular ones. For use in our existence proofs and for use as a possible method for solving the optimal control problem, we derived a change of variables formula for transforming the paraboloid from one with ellipsoidal cross-sections into one with circular cross-sections. The strength of the transformation lies in the fact that it is linear so that distances are preserved. We started with

$$J(f) = e^* e = (f^* C^* + d^*)(Cf + d) = f^* C^* C f + f^* C^* d + d^* C f + d^* d, \tag{3.18}$$

and developed the change of variables formula

$$l = \sqrt{D} P (f - h), \tag{3.19}$$

where the matrices P and D come from the eigenvalue decomposition of $C^* C$, $C^* C = P^* D P$, and

$$h = -(C^* C)^{-1} C^* d, \tag{3.20}$$

to derive a new function, $\hat{J}(l)$, equal to $J(f)$ under the change of variables and with the following form as a function of l:

$$\hat{J}(l) = l^* l + k, \tag{3.21}$$

where k is the constant term given by

$$k = d^* Ch + d^* d. \tag{3.22}$$

Note that the formula for $\hat{J}$ is much simpler than the formula for J given in (3.18) because it is a paraboloid with circular cross sections. Furthermore, minimizing $\hat{J}$ over all l of the form (3.19) for some f of the form (3.5) is equivalent to minimizing $J(f)$ over all f of the form (3.5). The change of variables described above is always possible because C^*C is Hermitian, so it can always be diagonalized into $C^*C = P^*DP$ where P is a unitary matrix ($P^*P = I$ and $PP^* = I$) and D is the diagonal matrix with the eigenvalues of C^*C along the diagonal. We know all of the eigenvalues of C^*C are real because C^*C is Hermitian and, if we assume that C^*C is positive definite, then we see that all eigenvalues are actually positive. Therefore, defining $\sqrt{D}$ to be the diagonal matrix with the square roots of the eigenvalues of C^*C along the diagonal, it is always possible to construct the change of variables described by (3.19). By solving (3.19) for f and plugging into (3.18), one can verify that (3.21) holds.

We explored the feasibility of using (3.21) to solve the optimal control problem, by substituting $f = T(\theta)u$ into (3.19), getting a new function $\hat{l}$ defined by

$$\hat{l}(u, \theta) = \sqrt{D} P (Tu - h), \tag{3.23}$$

and noting that minimizing (3.21) is equivalent to minimizing $\hat{l}^* \hat{l}$. Using that idea, we developed an explicit formula for u and θ in Maple for the case where $m = 1$. However, as was the case for finding critical values, the formulas arising from this method become unwieldy when m was greater than 1, so this method is not practical for the problems Lord Corporation will be attempting to solve. It is important to keep in mind, though, that the results from this change of variables approach were also used to help prove existence of a minimum. Hence, this work on change of variables is important and was not a wasted effort.

3.4.2 Existence of Global Minimum

In this section we shall prove the existence of a global minimum. We make no claim about uniqueness of the minimizer but, as mentioned before, we observed uniqueness numerically.

We divide the proof into two parts, but the key fact is the following lemma.

Lemma. $J(u, \theta) \to \infty$ uniformly in θ as $\|u\| \to \infty$
Proof. Let $f = T(\theta)u$. Then $\|f\|^2 = \|u\|^2$. Thus, it suffices to show that $J(f) \to \infty$ as $\|f\| \to \infty$.
Consider the change of variables from f to l given by

$$f = P^\star (\sqrt{D})^{-1} l + h,$$

where $C^\star C = P^\star DP$ and $h = -(C^\star C)^{-1} C^\star d$. Then $J(u, \theta) = l^\star l + k$, for some constant k. Thus, by the triangle inequality, we have

$$\|f\| \leq \|P^\star (\sqrt{D})^{-1} l\| + \|h\|,$$

and since $\|h\|$ is constant, we see that $\|f\| \to \infty$ implies that $\|P^\star(\sqrt{D})^{-1}l\| \to \infty$. Now $\|P^\star(\sqrt{D})^{-1}l\| \leq \|P^\star(\sqrt{D})^{-1}\|\|l\|$ where

$$\|P^\star(\sqrt{D})^{-1}\| = sup_{||v||=1}\|P^\star(\sqrt{D})^{-1}v\|,$$

i.e. the usual matrix norm.
Therefore $\|f\| \to \infty$ implies that $\|l\| \to \infty$ and, since $J(f) = \hat{J}(l) = \|l\|^2 + k$, we have that also $J \to \infty$. $\Box$

The above lemma allows us to use the fact that continuous functions on compact sets have a global minimum.

Proposition. $J(u, \theta)$ has a global minimum.
Proof. Let (u_0, θ_0) be arbitrary and consider $J(u_0, \theta_0) = b_0$. Then there is an M such that $\|u\| > M$ implies that, for every θ and u, $J(u, \theta) > b_0$. Thus we have

$$\{(u, \theta) : J(u, \theta) \leq b_0\} \subset \{u : \|u\| \leq M\} \times [0, \pi]^m,$$

and since the right hand side is compact and the left hand side is closed, we have that the left hand side is compact. Thus, J has a global minimum over the left hand side set. Therefore, it must also be a global minimum over all the possible combinations of $u's$ and $\theta's$ since any element not in the set on the left hand side is a combination of u and θ which gives $J(u, \theta) > b_0$.

3.4.3 Equivalent Formulations

In this section we state and prove three equivalent statements for the minimization problem. The first equality is obvious so we prove the second one.
Proposition The following equalities hold:

$$\begin{aligned} min_{(u,\theta)\in\mathbb{C}^m\times[0,\pi]^m} J_1(u, \theta) &= min_{(u_R,u_I,\theta)\in\mathbb{R}^m\times iR^m\times[0,\pi]^m} J_3(u_R, u_I, \theta) \\ &= min_{f_i \bar{f}_{m+i}\in R, i=1\cdots m} J_2(f), \end{aligned}$$

where

$$J_1(u, \theta) := J(T(\theta)u),$$

$$J_2(f) := J(f),$$

$$J_3(u_R, u_I, \theta) := J(T(\theta)(u_R + u_I)),$$

and thus we have two equivalent minimization problems.
Proof:
The first equality is clear. We prove the second equality. It suffices to show that the sets

$$\begin{aligned} S_1 &:= \{T(\theta)u | (u, \theta) \in \mathbb{C}^m \times [0, \pi]\}, \\ S_2 &:= \left\{f \in \mathbb{C}^{2m} | f_i \bar{f}_{i+m} \in \mathbb{R}, i = 1, \ldots, m\right\}, \end{aligned}$$

are equal.

Let $f \in S_1$. There exists $(u, \theta) \in \mathbb{C}^m \times [0, \pi]^m$ so that $f = T(\theta)u$. Then

$$\begin{aligned} f_i \bar{f}_{m+i} &= (u_i \cos\theta_i)(\bar{u_i} \sin\theta_i) \\ &= |u_i|^2 \cos\theta_i \sin\theta_i \end{aligned}$$

which is in $\mathbb{R}$. Therefore we have that $S_1 \subset S_2$. For the other direction, let f be in S_2, that is, $f_i \bar{f}_{m+i} \in \mathbb{R}$. We can find u_i and θ_i such that

$$\begin{aligned} f_i &= u_i \cos\theta_i, \\ f_{m+i} &= u_i \sin\theta_i. \end{aligned}$$

If $f_i \neq 0$ then $\frac{f_{m+i}}{f_i} = \frac{f_{m+i}\bar{f}_i}{|f_i|^2}$ which is in $\mathbb{R}$. Thus we can take $\theta_i = \arctan \frac{f_{m+i}}{f_i}$ where $\theta_i \in [0, \pi]$. If $\cos\theta_i \neq 0$ then we can take $u_i = f_i / \cos\theta_i$, otherwise we must have $\theta_i = \pi/2$ so that $\sin\theta_i = 1$ and hence we may take $u_i = f_{m+i}$. If $f_i = 0$ then take $\theta_i = \pi/2$ and $u_i = f_{m+i}$.

3.4.4 Numerical Solutions

The goal is to minimize the function

$$J(u, \theta) = e^\star e, \tag{3.24}$$

where $\star$ represents the complex conjugate transpose and thus J is a function

$$J : \mathbb{C}^m \times \mathbb{R}^m \to \mathbb{R}.$$

However, we would like to implement the Nelder-Mead algorithm so we need to rewrite this problem as a function of a real vector. If we write

$$u = u_1 + iu_2, \tag{3.25}$$

then we can realize the cost function J as a function $J : \mathbb{R}^{3m} \to \mathbb{R}$ or, if we let $N = 3m$, $J : \mathbb{R}^N \to \mathbb{R}$. The Nelder-Mead algorithm is a deterministic sampling method, i.e., it only requires function values and no gradient information and is thus an easy algorithm to implement. Given a $N + 1$ simplex of points in $\mathbb{R}^N, \{\lambda^1, \lambda^2, \cdots, \lambda^{N+1}\}$, it first sorts the simplex points such that $J(\lambda^1) \leq J(\lambda^2) \leq \cdots \leq J(\lambda^{N+1})$. It then attempts to minimize the function by replacing the simplex point with highest function value, the worst point, with a point with a lower function value. It first computes the centroid of the simplex, not including the worst points

$$\bar{\lambda} = \frac{1}{N} \Sigma \lambda^i. \tag{3.26}$$

It then attempts to replace λ_N with

$$\lambda_{NEW} = \lambda_N + \beta\bar{\lambda}, \tag{3.27}$$

where

$$\beta = \{\beta_R, \beta_E, \beta_{OC}, \beta_{IC}\}, \tag{3.28}$$

and the values for λ taken from

$$-1 < \beta_{IC} < 0 < \beta_{OC} < \beta_R < \beta_E.$$

If none of these values are better than the previous worst point the algorithm shrinks the simplex towards the best point, i. e. it replaces the points with

$$\bar{\lambda_i} = \frac{\lambda_i + \lambda_0}{2}.$$

The algorithm then re-sorts the points and iterates.

To better illustrate this method, consider the case when $N = 2$. In this case our parameter space is $\mathbb{R}^2$ and our simplex is a triangle in the plane. Figure 1 illustrates what Nelder-Mead does in this case. Note λ_3 is the worst parameter and the algorithm is attempting to replace this point with either a reflection, r, expansion, e, outward contraction, oc, or inward contraction, ic. There are various stopping criteria for this algorithm including iteration number and difference between the function at the best and worst points. For a more detailed treatment see [1].

Steepest descent is an iterative optimization method that attempts to use the derivative to minimize a given function. A direction d is a descent direction if $\frac{d}{dt}f(x+td)|_{t=0} = \nabla^T f(x)d < 0$ holds. Note that ∇f is always a descent direction. Steepest descent is implemented with the formula

$$x_+ = x_c - \mu \nabla J(x_c),$$

where μ is a parameter. For a given μ value $f(x_+) < f(x_c)$ may not hold. Thus, in many implementations of steepest descent, a line search is used. There are many ways to employ a line search. A simple example is that $\mu = 1$ is used and a new iterate x_+ is computed. If $J(x_+) < J(x_c)$ holds, another iterate of steepest descent is computed. If $J(x_+) < J(x_c)$ does not hold, μ is reduced and a new x_+ is computed. μ may be reduced until $J(x_+) < J(x_c)$ holds and, if it does, then μ may be reset to 1.
There are various ways to implement a line search, but it is likely that steepest descent may fail if one is not used. For a more detailed treatment see [1].

The BFGS (Broyden, Fletcher, Goldfarb, Shanno) method is based on the idea of forming a quadratic model of the function at the given iterate x_c :

$$m_c(x) = J(x_c) + \nabla^T J(x_c)(x - x_c) + \frac{1}{2}(x - x_c)^T \nabla^2 J(x_c)(x - x_c).$$

We then attempt to find a critical value of

$$\nabla m_c(x) = \nabla J(x_c) + \nabla^2 J(x_c)(x - x_c) = 0,$$

and, if the Hessian of J is positive definite, then the critical value is a minimum. Plugging x_+ into $\nabla m_c(x) = \nabla J(x_c) + \nabla^2 J(x_c)(x - x_c) = 0$ and solving for x_+ gives us the formula for x_+. BFGS is a quasi-Newton method which, instead of using the true Hessian matrix at a given iterate (which may or may not be positive definite), uses an approximation of the Hessian and updates it at each step with a rank-two update. An iterate for BFGS includes the steps

$$x_+ = x_c + \lambda H_c^{-1} \nabla J(x_c). \tag{3.29}$$

During this step a line search may be employed. The Hessian is updated with

$$s = x_+ - x_c, \tag{3.30}$$

$$y = \nabla J(x_+) - \nabla J(x_c), \tag{3.31}$$

$$H_+ = H_c + \frac{yy^T}{y^T s} - \frac{(H_c s)(H_c s)^T}{s^T H_c s}. \tag{3.32}$$

Without going into the details, this method is locally superlinearly convergent if the initial iterate and the initial Hessian are close enough to the minimizer and the Hessian at the minimizer, respectively. Both BFGS and steepest descent terminate on small norms of the gradient. For a more detailed treatment see [1].

Each of these methods was rather effective in minimizing this function. For numerical experiments we were forced to use random values for C and d as we were not given specific values for these quantities. As a test of the efficiency, we ran 50 optimizations with various initial iterates, C's and d's and calculated the mean time (in seconds) it took for each algorithm to converge. The statistics for the various values of m and n are reported below.

	m=2, n=2	m=3, n=7	m=5, n=11	m=7, n=15
NM	0.2369	0.7787	4.4798	9.8425
SD	0.9321	0.9763	3.0415	12.8129
BFGS	0.0466	0.055	0.1066	0.1605

BFSG performs extremely well, converging almost two orders of magnitude quicker than either of the other two methods. It is somewhat surprising that steepest descent did not converge more quickly than Nelder-Mead on a consistent basis. Steepest descent performed well in the early phases of the optimizations but struggles to reach termination tolerance as the iterates approached the minimizer. BFGS performs somewhat similar to steepest descent in the initial phases but performs much better in the terminal phases. All three codes were written by C.T. Kelley.

We noticed something interesting when the values for m were greater than $\frac{n}{2}$. This corresponds to a system with a large amount of actuators compared to the number of sensors. In these situations there appeared to be a large, perhaps infinite, number of minimizers each with cost function values of the same order of magnitude. We performed experiments where we chose 500 different initial iterates and, if n was sufficiently large compared to m, they nearly all converged to a different minimizer. This did not occur when m was sufficiently smaller than n. The large amount of minimizers seemed physically unfeasible to our engineering collaborator. Even though the number of sensors is usually significantly larger than the number of actuators, we wanted to experiment in the case where the number of actuators was large enough to create many minimizers. To, perhaps, give the engineers a clearer control protocol in these situations, we tried using the cost function $J = e^\star e + \epsilon u^\star u$, where ϵ is a design variable. Using this cost function, the optimization methods converged to single optimal control law even in situations when m was large in comparison to n ($2m > n$). Not only does this help prescribe a single control law, it also gives engineers an opportunity to weight the cost of actuator force versus the desire to minimize the sensor signal.

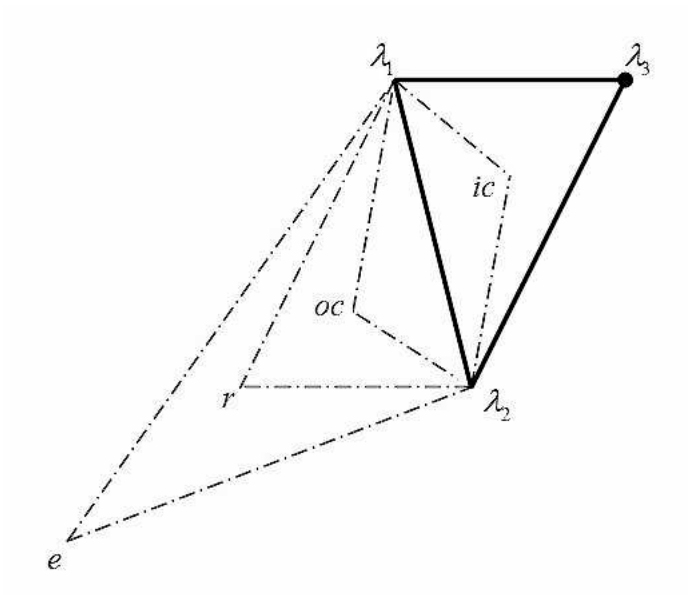


Figure 3.1: Nelder-Meade Algorithm

3.5 Conclusions

3.5.1 New Theorem

For $n > 2m$, the numerical results appear to indicate that there is a unique point where the minimum is attained, but we did not prove it. Hence, one thing that should be explored is to prove that result. One possible way to do that will be to use the equivalent formulation of the optimization problem which has polynomial equality constraints which we discussed above and use new results from Schichl and Neumaier from 2005. Their theorem gives necessary and sufficient conditions when a feasible point is a global minimum for a polynomial optimization problem with polynomial constraints. For more information, one can read "Transposition Theorems and Qualification-Free Optimality Conditions" by Hermann Schichl and Arnold Neumaier, which can be obtained at http://www.mat.univie.ac.at/neum/

3.5.2 Numerical Directions

The next step in the numerical work of this project is to experiment with the new cost function. In our preliminary work this seemed to alleviate the problem of not having a unique minimizer when the number of actuators was very large in comparison to the number of sensors. Another area which remains to be explored is the performance of the given algorithms when physical data is used.

Bibliography

[1] Kelley, C. T. (1999) *Iterative Methods for Optimization*, SIAM Frontiers in Applied Mathematics, SIAM, Philadelphia, PA.

Chapter 4

Mathematical Models of Signal Transduction: Model Reduction

PROBLEM 4: MATHEMATICAL MODELS OF SIGNAL TRANSDUCTION: MODEL REDUCTION

Ananda Bandulasiri[1], Chad Gonzales[2], Janine Haugh[3], Bing Jian[4], Sarah Lynn Joyner[5], Neslihan Saglanmak[6], Thobile Thukani[7], Guohua Zhou[8],

Problem Presenter:
Anna Georgieva, Ph.D.
Novartis, Inc.

Faculty Consultants:
Dr. Alina Chertock & Dr. Sharon R. Lubkin
North Carolina State University

Abstract

Cellular signal transduction pathways are characterized by their high complexity. Detailed mathematical description of such systems leads to very large systems of ordinary differential equations. We seek to simplify the models, but in such a way that the reduced models still capture essential dynamics. This project uses a test system, the Wnt pathway, to evaluate the utility of three methods of reduction: (1) algebraic reduction by identification of conserved quantities; (2) use of sensitivity analysis to remove terms to which the system is relatively insensitive; (3) identification and elimination of quantitatively small terms in the nondimensionalized differential equations. We find that a method using all three techniques in the context of singular perturbations may yield the desired reduction.

4.1 Introduction

Cellular signal transduction pathways are characterized by their high complexity. Mathematical models describing these processes might be of great help to gain qualitative and, most importantly, quantitative knowledge about such complex systems. In particular, model predictions will be used to contribute to better understanding of molecular mechanisms of cancer. Models of signal transduction will be useful in the context of biomarker identification for drug therapy. Detailed mathematical description of such systems leads to very large systems of ordinary differential equations. The situation becomes even more complicated when we consider two or more pathways simultaneously. Therefore, it becomes necessary to simplify the models, but in such a way that the reduced models still capture essential dynamics.

[1]Texas Tech University
[2]Arizona State University
[3]North Carolina State University
[4]University of Florida
[5]Meredith College
[6]Roskilde University
[7]Southern University and A&M College
[8]Clarkson University

A signal transduction pathway is the process whereby a cell receives some stimulus and converts it into another kind of stimulus. In the study of certain diseases, knowledge of these pathways may provide an opportunity to control some of the reactions involved using drugs. However, it is not possible to track the pathways without interfering with the reactions, so mathematical modeling can be a useful tool for understanding the mechanisms involved. Modeling provides important quantitative information about signal transduction pathways that could previously be studied only qualitatively, but as these highly complex pathways typically consist of many biochemical reactions involving several enzymes and proteins, models become quite large and involve many differential equations. Therefore it is advantageous to simplify these models in order to study the pathways more easily. In model reduction, the essential dynamics of the system must be considered and preserved to accurately reflect the biology with simplified mathematics.

Two such signal pathway models are the Wnt pathway [3] and the epidermal growth factor (EGF) pathway [5]. The Wnt pathway plays a role in the development of human cancer and is the less complex of the two. The EGF pathway consists of a much larger number of chemical reactions and is involved in embryonic and postnatal development as well as tumor growth.

The focus of this study is reduction of the Wnt pathway model, but the methods used can be applied to the EGF model as well. The system contains many chemical reactions that are modeled using Michaelis-Menten kinetics, and because the differential equations for some reactions contain more than one concentration variable multiplied together, the system is nonlinear.

Reduction of the models for these pathways will simplify the mathematics so that the biological process can be studied more easily. Three different methods have been applied to reduce the system, and each approaches the reduction differently.

4.2 Algebraic Decoupling

The input to the Wnt pathway is the Wnt molecule and the output is β-catenin. The model contains 15 differential equations, each describing the kinetics of a separate chemical species c_i.

The equations are of the form

$$\frac{dc_i}{dt} = \sum v_j, \tag{4.1}$$

where c_i is the concentration of the ith molecule and the v_js are the rates of the reactions that include c_i.

The rate equations v_j are dependent upon concentrations of several different molecules c_i as well as rate constants k_j and kd_j. The concentrations of the molecules are the variables, and some of the rate equations contain products of more than one concentration, so the system we are modeling is nonlinear. However, the nonlinear system can be studied in part using linear methods because the differential equations $\frac{dc_i}{dt}$ describing the kinetics of the reactions can be written as sums of the rates v_j, and the v_js are treated as variables in the related linear system.

The nonlinear differential equations for the Wnt pathway in terms of concentrations and rate constants are as follows:

$$
\begin{aligned}
\frac{dc_1}{dt} &= -k_1c_1 + k_2c_2 \\
\frac{dc_2}{dt} &= k_1c_1 - k_2c_2 \\
\frac{dc_3}{dt} &= k_4c_4 - k_5c_3 - k_8c_{11}c_3 + kd_8c_8 + k_{10}c_9 \\
\frac{dc_4}{dt} &= -k_3c_2c_4 - k_4c_4 + k_5c_3 + k_6c_6c_5 - kd_6c_4 \\
\frac{dc_5}{dt} &= k_3c_2c_4 - k_6c_6c_5 + kd_6c_4 \\
\frac{dc_6}{dt} &= k_3c_2c_4 - k_6c_6c_5 + kd_6c_4 + k_7c_7c_{12} - kd_7c_6 \\
\frac{dc_7}{dt} &= -k_7c_7c_{12} + kd_7c_6 - k_{17}c_{11}c_7 + kd_{17}c_{15} \\
\frac{dc_8}{dt} &= k_8c_{11}c_3 - kd_8c_8 - k_9c_8 \\
\frac{dc_9}{dt} &= k_9c_8 - k_{10}c_9 \\
\frac{dc_{10}}{dt} &= k_{10}c_9 - k_{11}c_10 \\
\frac{dc_{11}}{dt} &= -k_8c_{11}c_3 + kd_8c_8 + k_{12} - k_{13}c_{11} - k_{16}c_{11}c_{13} \\
&\quad +kd_{16}c_{14} - k_{17}c_{11}c_7 + kd_{17}c_{15} \\
\frac{dc_{12}}{dt} &= -k_7c_7c_{12} + kd_7c_6 + k_{14} - k_{15}c_{12} \\
\frac{dc_{13}}{dt} &= -k_{16}c_{11}c_{13} + kd_{16}c_{14} \\
\frac{dc_{14}}{dt} &= k_{16}c_{11}c_{13} - kd_{16}c_{14} \\
\frac{dc_{15}}{dt} &= k_{17}c_{11}c_7 - kd_{17}c_{15}
\end{aligned}
$$

where the c_is denote the concentrations of the 15 molecules involved in the pathway and the k_js and kd_js are the reaction constants.

The linear differential equations for the Wnt pathway in terms of reaction rates are as follows:

$$\begin{aligned}
\frac{dc_1}{dt} &= -v_1 + v_2 \\
\frac{dc_2}{dt} &= v_1 - v_2 \\
\frac{dc_3}{dt} &= v_4 - v_5 - v_8 + v_{10} \\
\frac{dc_4}{dt} &= -v_3 - v_4 + v_5 + v_6 \\
\frac{dc_5}{dt} &= v_3 - v_6 \\
\frac{dc_6}{dt} &= v_3 - v_6 + v_7 \\
\frac{dc_7}{dt} &= -v_7 - v_{17} \\
\frac{dc_8}{dt} &= v_8 - v_9 \\
\frac{dc_9}{dt} &= v_9 - v_{10} \\
\frac{dc_{10}}{dt} &= v_{10} - v_{11} \\
\frac{dc_{11}}{dt} &= -v_8 + v_{12} - v_{13} - v_{16} - v_{17} \\
\frac{dc_{12}}{dt} &= -v_7 + v_{14} - v_{15} \\
\frac{dc_{13}}{dt} &= -v_{16} \\
\frac{dc_{14}}{dt} &= v_{16} \\
\frac{dc_{15}}{dt} &= v_{17}
\end{aligned}$$

Each rate v_j is expressed by rate constants k and concentrations c. The rates v_{12} and v_{14} are protein synthesis reactions and are described by constant values.

In working with algebraically decoupling the Wnt pathway, we have considered two approaches/methods. One approach was to decouple the model to end up with smaller systems of equations, the other was to eliminate a number of equations from the existing system of equations.

4.2.1 Decoupling the Equations

In [1], Conzelmann *et al.* decompose the model based on the criterion of absence of retroactivity and obtain modules which are connected in most cases with a weak retroactivity. In contrast to their approach, we try to find a way to decouple the model from the mathematical point of view. Consider the following linear system:

$$\frac{d\mathbf{c}}{dt} = A\mathbf{v}, \tag{4.2}$$

where $\mathbf{c}$ denotes the vector of concentrations and A is the coefficient matrix which has been reduced to full rank. First we multiply A^T on both sides of (4.2) which gives

$$A^T \frac{d\mathbf{c}}{dt} = A^T A\mathbf{v}, \tag{4.3}$$

Note $M = A^T A$ is a symmetric matrix; hence there exists orthogonal diagonalization $M = PDP^T$ where P is an orthogonal matrix and D ia a diagonal matrix. By changing variables: $\bar{\mathbf{v}} = P^T\mathbf{v}$, $\bar{\mathbf{c}} = P^T A^T \mathbf{c}$, we get the decoupled system for $\bar{\mathbf{c}}$:

$$\frac{d\bar{\mathbf{c}}}{dt} = D\bar{\mathbf{v}}. \tag{4.4}$$

Unfortunately, each $\bar{v}_j$ and $\bar{c}_i$ is a linear or nonlinear combination of c_i, which means $\bar{c}_i$ and $\bar{v}_j$ cannot be independent at all even when $i \neq j$. Due to this reason, it is not feasible to get the decoupled system for $\bar{\mathbf{c}}$ eventually.

However, it is possible to decouple the original nonlinear system directly using a nonlinear ordinary differential equation solver like NODES[9].

In order to reduce the linear system of v_js, we must find out which differential equations can be added together to arrive at total rate of zero. Because their sum of rates is zero, the sum of their respective chemical concentrations must be a constant. The initial conditions for the concentrations are known, and quantities are conserved, so the conserved quantities can be found. Using these conservation equations, we can solve for one of the concentrations involved in each equation and eliminate it in the non-linear system [2].

For the Wnt pathway studied here, we first create a stoichiometry matrix of the reaction rates, which is the linear system of equations. The rank of this stoichiometry matrix is found so that we know how many of the original equations are independent and thereby how many can be eliminated. The null space of the stoichiometry matrix gives the linear combinations for the conserved quantities.

4.2.2 Results

The stoichiometry matrix for the Wnt pathway

$$A = \begin{pmatrix}
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

[9]http://cso.ulb.ac.be/ mcodutti/nodes/

has dimensions 15×17, and its rank was determined to be 11, implying that four equations could be eliminated. The null space of the matrix showed that the sums of the following equations were zero:

$$\frac{dc_1}{dt} + \frac{dc_2}{dt} = 0 \tag{4.5}$$

$$\frac{dc_3}{dt} + \frac{dc_4}{dt} + \frac{dc_5}{dt} + \frac{dc_8}{dt} + \frac{dc_9}{dt} = 0 \tag{4.6}$$

$$\frac{dc_{13}}{dt} + \frac{dc_{14}}{dt} = 0 \tag{4.7}$$

$$\frac{dc_5}{dt} + \frac{dc_6}{dt} + \frac{dc_7}{dt} + \frac{dc_{15}}{dt} = 0 \tag{4.8}$$

Therefore the sums of the corresponding concentrations are constants, denoted by r_i:

$$c_1 + c_2 = \quad r_1 \quad = c_1(0) + c_2(0) \tag{4.9}$$

$$c_3 + c_4 + c_5 + c_8 + c_9 = \quad r_2 \quad = c_3(0) + c_4(0) + c_5(0) + c_8(0) + c_9(0) \tag{4.10}$$

$$c_{13} + c_{14} = \quad r_3 \quad = c_{13}(0) + c_{14}(0) \tag{4.11}$$

$$c_5 + c_6 + c_7 + c_{15} = \quad r_4 \quad = c_5(0) + c_6(0) + c_7(0) + c_{15}(0) \tag{4.12}$$

Since we have four conserved quantities we can eliminate four variables, one from each of the conserved equations. We have chosen to eliminate c_2, c_9, c_{14}, and c_{15}. When substituting equations (4.9)-(4.12) into equation (4.2) we get the following reduced system of equations, which now contains 11 equations:

$$
\begin{aligned}
\frac{dc_1}{dt} &= -k_1 c_1 + k_2 \left(r_1 - c_1\right) \\
\frac{dc_3}{dt} &= k_4 c_4 - k_5 c_3 - k_8 c_{11} c_3 + kd_8 c_8 + k_{10} \left(r_2 - c_3 - c_4 - c_5 - c_8\right) \\
\frac{dc_4}{dt} &= -k_3 \left(r_1 - c_1\right) c_4 - k_4 \, c_4 + k_5 c_3 + k_6 c_6 c_5 - kd_6 c_4 \\
\frac{dc_5}{dt} &= k_3 \left(r_1 - c_1\right) c_4 - k_6 c_6 c_5 + kd_6 c_4 \\
\frac{dc_6}{dt} &= k_3 \left(r_1 - c_1\right) c_4 - k_6 c_6 c_5 + kd_6 c_4 + k_7 c_7 c_{12} - kd_7 c_6 \\
\frac{dc_7}{dt} &= -k_7 c_7 c_{12} + kd_7 c_6 - k_{17} c_{11} c_7 + kd_{17} \left(r_4 + c_5 - c_6 - c_7\right) \\
\frac{dc_8}{dt} &= k_8 c_{11} c_3 - kd_8 c_8 - k_9 c_8 \\
\frac{dc_{10}}{dt} &= k_{10} \left(r_2 - c_3 - c_4 - c_5 - c_8\right) - k_{11} c_{10} \\
\frac{dc_{11}}{dt} &= -k_8 c_{11} c_3 + kd_8 c_8 + k_{12} - k_{13} c_{11} - k_{16} c_{11} c_{13} + kd_{16} \left(r_3 - c_{13}\right) \\
&\quad -k_{17} c_{11} c_7 + kd_{17} \left(r_4 + c_5 - c_6 - c_7\right) \\
\frac{dc_{12}}{dt} &= -k_7 c_7 c_{12} + kd_7 c_6 + k_{14} - k_{15} c_{12} \\
\frac{dc_{13}}{dt} &= -k_{16} c_{11} c_{13} + kd_{16} \left(r_3 - c_{13}\right)
\end{aligned}
$$

4.3 Sensitivity Analysis

The system depends on a large number of rate parameters and initial conditions. If the system is insensitive to some of these, the model may perhaps be reduced accordingly. As there are many parameters which can be tuned for sensitivity analysis, here we only choose to focus on how the system changes with variation in the initial conditions. For this purpose, we simulate the effects of changes in the 5 nonzero initial conditions $c_1(0), c_5(0), c_7(0), c_{11}(0), c_{13}(0)$. In each case we vary the initial condition by -5% to 5% of its baseline value, and integrate the rate equations for 2000 mins, which is sufficiently long to get a stable β-catenin concentration. As an example, the effect of initial conditions on the concentration of β-catenin is shown in Figure 4.1.

4.3.1 Results

We also observe how other equations change under the perturbation to these 5 initial conditions. The general result is summarized in Table 4.1.

We see that most of the final concentrations are insensitive to most of the initial conditions.

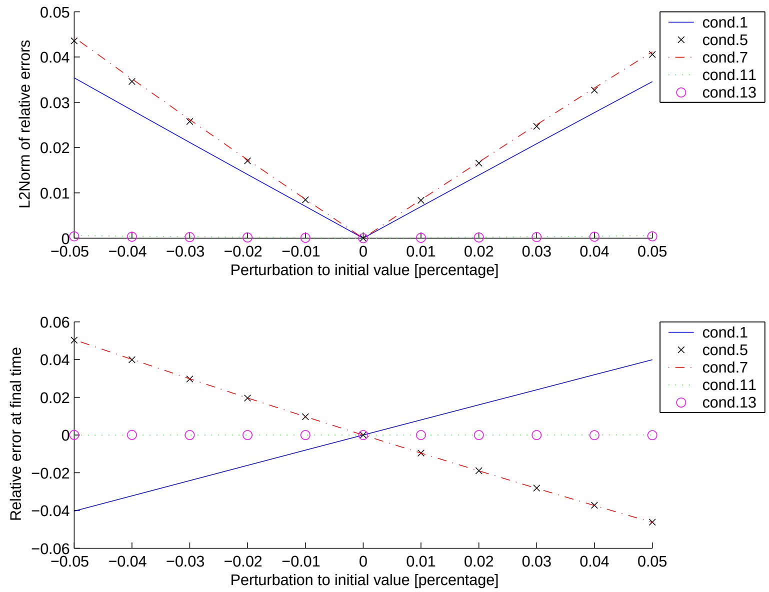


Figure 4.1: Effects of the initial conditions on equilibrium β-catenin concentration $c_{11}(2000)$. For the perturbation of initial conditions $c_1(0)$, $c_5(0)$, and $c_7(0)$, the resulting relative errors are within 5% at 2000 mins. For the initial conditions $c_{11}(0)$ and $c_{13}(0)$, from the β-catenin equation, the errors tend to zero as time increases.

eqn.	$c_1(0)$	$c_5(0)$	$c_7(0)$	$c_{11}(0)$	$c_{13}(0)$
c_1	5%	10^{-14}	10^{-14}	10^{-14}	10^{-14}
c_2	5%	10^{-14}	10^{-14}	10^{-14}	10^{-14}
c_3	5%	5%	5%	0	0
c_4	5%	5%	5%	0	0
c_5	10^{-6}	5%	10^{-6}	0	0
c_6	10^{-3}	10^{-3}	5.6%	0	0
c_7	10^{-3}	10^{-3}	5.6%	0	0
c_8	1%	1%	1%	0	0
c_9	1%	1%	1%	0	0
c_{10}	1%	1%	1%	0	0
c_{11}	4%	5%	5%	0	0
c_{12}	0	0	0	0	0
c_{13}	3.6%	4.1%	4.1%	0	5%
c_{14}	0.7%	0.8%	0.8%	0	5%
c_{15}	3.7%	4.5%	0.8%	0	0

Table 4.1: Effects of initial conditions $c_1(0)$,$c_5(0)$,$c_7(0)$,$c_{11}(0)$,$c_{13}(0)$ on outputs of concentrations c_1, c_2 etc. Entries in table represent maximum values of the relative errors of concentrations at the steady state which are defined as $(\tilde{c_i}(finaltime) - \bar{c_i}(finaltime))/\bar{c_i}(finaltime)$ where $\tilde{c_i}$ is the perturbed solution and $\bar{c_i}$ is the baseline solution.

4.4 Non-dimensionalization

In the majority of cases mathematical models of signal transduction are too complex for intuitive understanding. Therefore, it is a very important task to reduce these mathematical systems to simpler ones to identify the key components and gain some insight into the dynamics of the system. If we knew which terms in the model were quantitatively small, we could reduce the model by eliminating them. But smallness is relative, so identification of small terms must be done on a nondimensionalized version of our system.

4.4.1 Method

In order to nondimensionalize our system, we must first examine the units of our variables and rate constants. The concentrations c_i, are measured in units of nM. The units for the rate constants, k_i and kd_i, are given in Table 4.2. Time, t, is measured in min.

We then rewrite all variables as a product of a dimensional scalar and a dimensionless variable [2], [4]. For example:

$$t = t^* \hat{t}, \tag{4.13}$$

where t^* is a scalar with the same units as t, and $\hat{t}$ is a dimensionless variable.

Similarly,

$$c_i = {c_i}^* \hat{c}_i, \tag{4.14}$$

follows the same convention for all $i \in [1, 15]$

4.4.2 Results

Since initial concentrations, $c_i(0)$, are of roughly the same order of magnitude (Table 4.3), we write ${c_i}^* = c^*$

parameter	units	baseline value
k_1	$\min^{-1}$	0.182
k_2	$\min^{-1}$	.0182
k_3	$\min^{-1}\text{nM}^{-1}$	.05
k_4	$\min^{-1}$	0.267
k_5	$\min^{-1}$	.133
k_6	$\min^{-1}\text{nM}^{-1}$	.0909
kd_6	$\min^{-1}$	.0909
k_7	$\min^{-1}\text{nM}^{-1}$	0.01
kd_7	$\min^{-1}\text{nM}^{-1}$	$50k_7$
k_8	$\min^{-1}\text{nM}^{-1}$	1000
kd_8	$\min^{-1}\text{nM}^{-1}$	$120k_8$
k_9	$\min^{-1}$	206
k_{10}	$\min^{-1}$	206
k_{11}	$\min^{-1}$	0.417
k_{12}	nM/min	0.423
k_{13}	$\min^{-1}$	0.000257
k_{14}	nM/min	8.22×10^{-5}
k_{15}	$\min^{-1}$	0.167
k_{16}	$\min^{-1}\text{nM}^{-1}$	0.1
kd_{16}	$\min^{-1}\text{nM}^{-1}$	$30k_{16}$
k_{17}	$\min^{-1}\text{nM}^{-1}$	0.0095
kd_{17}	$\min^{-1}\text{nM}^{-1}$	$1200k_{17}$

Table 4.2: Rate constants for Wnt pathway

Variable	Initial Condition(nM)
c_1	100
c_2	0
c_3	0
c_4	0
c_5	50
c_6	0
c_7	100
c_8	0
c_9	0
c_{10}	1
c_{11}	34
c_{12}	0.02
c_{13}	15
c_{14}	0
c_{15}	0

Table 4.3: Initial Conditions for Wnt Pathway

for all i. We obtain the following system of dimensionless equations:

$$\frac{d\hat{c}_1}{d\hat{t}} = (-k_1 t^* \hat{c}_1) + (k_2 t^* \hat{c}_2) \tag{4.15}$$

$$\frac{d\hat{c}_2}{d\hat{t}} = (k_1 t^* \hat{c}_1) - (k_2 t^* \hat{c}_2) \tag{4.16}$$

$$\frac{d\hat{c}_3}{d\hat{t}} = (k_4 t^* \hat{c}_4) - (k_5 t^* \hat{c}_3) - (k_8 c^* t^* \hat{c}_{11} \hat{c}_3) + (kd_8 t^* \hat{c}_8) + (k_{10} t^* \hat{c}_9) \tag{4.17}$$

$$\frac{d\hat{c}_4}{d\hat{t}} = (-k_3 c^* t^* \hat{c}_2 \hat{c}_4) - (k_4 t^* \hat{c}_4) + (k_5 t^* \hat{c}_3) + (k_6 c^* t^* \hat{c}_6 \hat{c}_5) - (kd_6 t^* \hat{c}_4) \tag{4.18}$$

$$\frac{d\hat{c}_5}{d\hat{t}} = (k_3 c^* t^* \hat{c}_2 \hat{c}_4) - (k_6 c^* t^* \hat{c}_6 \hat{c}_5) + (kd_6 t^* \hat{c}_4) \tag{4.19}$$

$$\frac{d\hat{c}_6}{d\hat{t}} = (k_3 c^* t^* \hat{c}_2 \hat{c}_4) - (k_6 c^* t^* \hat{c}_6 \hat{c}_5) + (kd_6 t^* \hat{c}_4) + (k_7 c^* t^* \hat{c}_7 \hat{c}_{12}) - (kd_7 t^* \hat{c}_6) \tag{4.20}$$

$$\frac{d\hat{c}_7}{d\hat{t}} = (-k_7 c^* t^* \hat{c}_7 \hat{c}_{12}) + (kd_7 t^* \hat{c}_6) - (k_{17} c^* t^* \hat{c}_{11} \hat{c}_7) + (kd_{17} t^* \hat{c}_{15}) \tag{4.21}$$

$$\frac{d\hat{c}_8}{d\hat{t}} = (k_8 c^* t^* \hat{c}_{11} \hat{c}_3) - (kd_8 t^* \hat{c}_9) - (k_9 t^* \hat{c}_8) \tag{4.22}$$

$$\frac{d\hat{c}_9}{d\hat{t}} = (k_9 t^* \hat{c}_8) - (k_{10} t^* \hat{c}_9) \tag{4.23}$$

$$\frac{d\hat{c}_{10}}{d\hat{t}} = (k_{10} t^* \hat{c}_9) - (k_{11} t^* \hat{c}_{10}) \tag{4.24}$$

$$\frac{d\hat{c}_{11}}{d\hat{t}} = (-k_8 c^* t^* \hat{c}_{11} \hat{c}_3) + (kd_8 t^* \hat{c}_8) + (k_{12} t^* / c^*) - (k_{13} t^* \hat{c}_{11}) - (k_{16} c^* t^* \hat{c}_{11} \hat{c}_{13}) + (kd_{16} t^* \hat{c}_{14}) \tag{4.25}$$

$$- (k_{17} c^* t^* \hat{c}_{11} \hat{c}_7) - (kd_{17} t^* \hat{c}_{15}) \tag{4.26}$$

$$\frac{dc_{12}}{d\hat{t}} = (k_7 c^* t^* \hat{c}_7 \hat{c}_{12}) - (kd_7 t^* \hat{c}_6) + (k_{14} / c^*) - (k_{15} t^* \hat{c}_{12}) \tag{4.27}$$

$$\frac{d\hat{c}_{13}}{d\hat{t}} = (-k_{16} c^* t^* \hat{c}_7 \hat{c}_{12}) + (kd_{16} t^* \hat{c}_{14}) \tag{4.28}$$

$$\frac{d\hat{c}_{14}}{d\hat{t}} = (k_{16} c^* t^* \hat{c}_{11} \hat{c}_{13}) - (kd_{16} t^* \hat{c}_{14}) \tag{4.29}$$

$$\frac{d\hat{c}_{15}}{d\hat{t}} = (k_{17} c^* t^* \hat{c}_{11} \hat{c}_7) - (kd_{17} t^* \hat{c}_{15}) \tag{4.30}$$

We chose to examine the system for three different values of c^*. We used a benchmark value of $c^* = 10$ nM. We then looked at the system as $c^* \to 0$ and as $c^* \to \infty$.

<u>(a) $c^* \to 0$</u> The simplest situation turned out to be the one in which $c^* \to 0$. In this case, the effect of the nonlinear terms was negligible because these terms were all multiplied by c^*. After dropping these terms, we are left with a system of fifteen linear differential equations, and can obtain an exact solution.

<u>(b) $c^* = 10$</u> As the baseline case we take $c^* = 10$. Eliminating numerically small terms reduces the Wnt

system to the following simpler system.

$$\frac{d\hat{c}_1}{d\hat{t}} \sim -k_1\hat{c}_1 t^* \tag{4.31}$$

$$\frac{d\hat{c}_2}{d\hat{t}} \sim k_1\hat{c}_1 t^* \tag{4.32}$$

$$\frac{d\hat{c}_3}{d\hat{t}} \sim kd_8\hat{c}_8 t^* \tag{4.33}$$

$$\frac{d\hat{c}_8}{d\hat{t}} \sim (k_8 c^*\hat{c}_{11}\hat{c}_3 - kd_8\hat{c}_9)\, t^* \tag{4.34}$$

$$\frac{d\hat{c}_9}{d\hat{t}} \sim (k_9\hat{c}_8 - k_{10}\hat{c}_9)\, t^* \tag{4.35}$$

$$\frac{d\hat{c}_{10}}{d\hat{t}} \sim (k_{10}\hat{c}_9 - k_{11}\hat{c}_{10})\, t^* \tag{4.36}$$

$$\tag{4.37}$$

Where we may separately solve the remaining rate equations. Thus with $c^* = 10$ we are able to reduce the number of differential equations from 15 to 7 in which only 5 are coupled. This reduces the complexity of our system and allows for the calculation of an exact solution.

(c) $c^* \to \infty$ When $c^* \to \infty$, we can split the large system into a series of smaller subsystems. From this we are left with 6 subsystems in which we can solve two of the systems equations (25)-(26) and (27)-(31), numerically and/or analytically.

$$\frac{d\hat{c}_1}{d\hat{t}} = (-k_1\hat{c}_1 + k_2\hat{c}_2)\, t^* \tag{4.38}$$

$$\frac{d\hat{c}_2}{d\hat{t}} = (k_1\hat{c}_1 - k_2\hat{c}_2)\, t^* \tag{4.39}$$

$$\frac{d\hat{c}_3}{d\hat{t}} = -c^* t^* k_8\hat{c}_{11}\hat{c}_3 \tag{4.40}$$

$$\frac{d\hat{c}_7}{d\hat{t}} = c^* t^* (-k_7\hat{c}_7\hat{c}_{12} - k_{17}\hat{c}_{11}\hat{c}_7) \tag{4.41}$$

$$\frac{d\hat{c}_{11}}{d\hat{t}} = c^* t^* (-k_8\hat{c}_{11}\hat{c}_3 - k_{16}\hat{c}_{11}\hat{c}_{13} - k_{17}\hat{c}_{11}\hat{c}_7) \tag{4.42}$$

$$\frac{d\hat{c}_{12}}{d\hat{t}} = c^* t^* k_7\hat{c}_7\hat{c}_{12} \tag{4.43}$$

$$\frac{d\hat{c}_{13}}{d\hat{t}} = -c^* t^* k_{16}\hat{c}_7\hat{c}_{12} \tag{4.44}$$

We can then substitute these solutions into the following 4 subsystems to obtain solutions for the whole system.

$$\frac{d\hat{c}_{14}}{d\hat{t}} = c^* t^* k_{16}\hat{c}_{11}\hat{c}_{13} \tag{4.45}$$

$$\frac{d\hat{c}_{15}}{d\hat{t}} = c^* t^* k_{17}\hat{c}_{11}\hat{c}_7 \tag{4.46}$$

$$\frac{d\hat{c}_4}{d\hat{t}} = c^*t^*\left(-k_3\hat{c}_2\hat{c}_4 + k_6\hat{c}_6\hat{c}_5\right) \tag{4.47}$$

$$\frac{d\hat{c}_5}{d\hat{t}} = c^*t^*\left(k_3\hat{c}_2\hat{c}_4 - k_6\hat{c}_6\hat{c}_5\right) \tag{4.48}$$

$$\frac{d\hat{c}_6}{d\hat{t}} = c^*t^*\left(k_3\hat{c}_2\hat{c}_4 - k_6\hat{c}_6\hat{c}_5 + k_7\hat{c}_7\hat{c}_{12}\right) \tag{4.49}$$

$$\frac{d\hat{c}_9}{d\hat{t}} = t^*\left(k_9\hat{c}_8 - k_{10}\hat{c}_9\right) \tag{4.50}$$

$$\frac{d\hat{c}_{10}}{d\hat{t}} = t^*\left(k_{10}\hat{c}_9 - k_{11}\hat{c}_{10}\right) \tag{4.51}$$

$$\frac{d\hat{c}_8}{d\hat{t}} = c^*t^*k_8\hat{c}_{11}\hat{c}_3 \tag{4.52}$$

We tried to numerically solve the reduced nonlinear systems with Matlab, but our approximation caused the equations to be much stiffer even than the original stiff system. Even with the use of the odes15 solver, we were unable to obtain a solution for $c^* \to \infty$.

For $c^* \to 0$ and $c^* = 10$, we compared the resulting graphs for c_{11} (β-Catenin) with the numerical solution of the original system. This plot is showing in Figure 4.2 The appropriate c^* value to use for a situation would have to be determined on a case-by-case basis, based on the biology of the system. In some cases, it may be useful to use one of these new systems, but in others it would be necessary to use the full system.

4.5 Discussion

Our main goal in this paper is to reduce the original model to a simpler one through various mathematical techniques.

First, based on the rate equations, we obtain the stoichiometry matrix. By computing its rank and determining the null space, we can get information about the conserved equations. Again, we look into the reduced full rank stoichiometry matrix and perform a linear transformation to decouple the rate equations. However, we cannot obtain a decoupled system for $\hat{c}_i$. Since the transferring matrix in our Wnt system is not a block diagonal matrix it is also difficult to decouple it into block subsystem.

Second, we nondimensionalize our nonlinear system, by choosing different scalar c^* i.e. $c^* = 10, 0, \infty$. We try three different approximating systems, all of which led to a reduced system. However, the simulation for each of the reduced systems led to much different dynamics than the original system. In the case $c^* \to \infty$, the numerical solution failed because of stiffness of the system.

Third, we conducted sensitivity analysis. We tested the effects of initial conditions on the numerical solution of the system. From table 4.1, we can see that the system is not sensitive to the initial values of β-catenin. This leads us to believe that we can decouple some of the equations by setting β-catenin to a constant value in some equations. This still needs to be tested.

A method which may yield more robust reductions would use the sensitivity analysis of section 3 with the reduction method of section 4 in a matched asymptotic scheme, combined with the conservation approach of section 2.

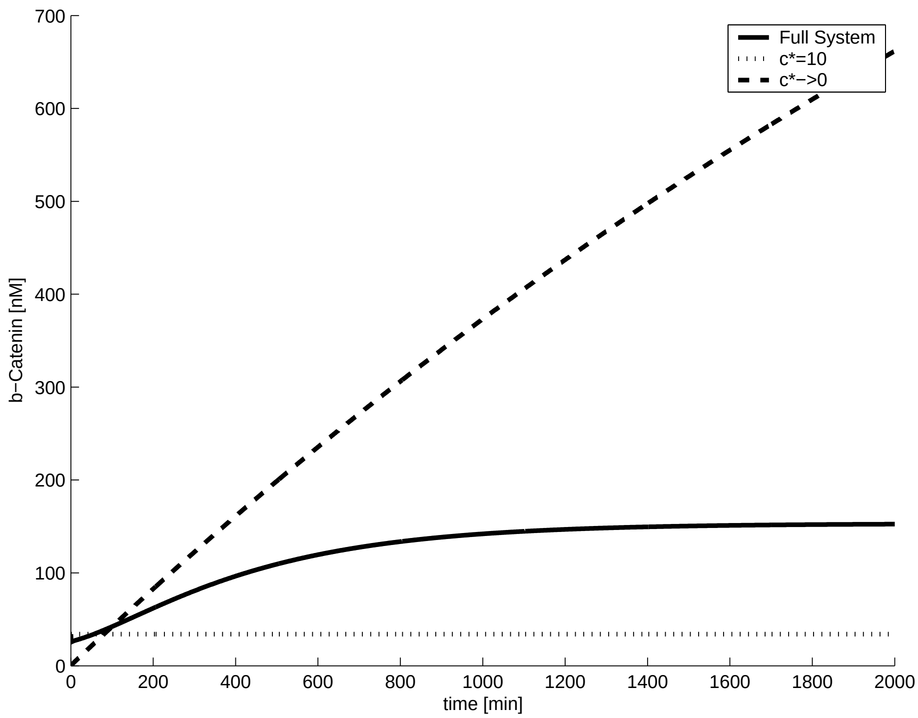


Figure 4.2: Comparison of solution of Wnt model with solution of reduced model with $c^* = 10$ and $c^* \to 0$.

Bibliography

[1] Conzelmann, H., Saez-Rodriguez, J., Sauter, T., Bullinger, E., Allgower, F. and Gilles, E. D. (2004) Reduction of Mathematical Models of Signal Transduction Networks: Simulation-based Approach Applied to EGF Receptor Signalling, *IEE Systems Biology*, 1:159-169.

[2] Edelstein-Keshet, L. (1988) *Mathematical Models in Biology*, McGraw-Hill.

[3] Lee, E., Salic, A., Krger, R., Heinrich, R. and Kirschner, M. W. (2003) The Roles of APC and Axin Derived from Experimental and Theoretical Analysis of the Wnt Pathway, *PLOS Biology*, 1:116-132.

[4] Lin, C. C. and Segel, L. A. (1988) *Mathematics Applied to Deterministic Problems in the Natural Sciences*, SIAM, Philadelphia.

[5] Schoeberl, B., Eichler-Johnsson, C., Gilles, E. D. and Mueller, G. (2002) Computational Modeling of the Dynamics of the MAP Kinase Cascade Activated by Surface and Internalized EGF Receptors, *Nature Biotechnology*, 20:370-375.

Chapter 5

Power Preallocation and Control in a Heavy Traffic Queueing Model for Wireless Systems

PROBLEM 5: POWER PREALLOCATION AND CONTROL IN A HEAVY TRAFFIC QUEUEING MODEL FOR WIRELESS SYSTEMS

Norou Diawara[1], Xiaoying Han[2], Bret Hanlon[3], Chuan Lin[4], Menaka Navaratna[5], Amith Vijayat[6], Anastasia Wong[7]

Problem Presenter:
Priya Ranjan, Ph.D.
Intelligent Automation, Inc.

Faculty Consultants:
Dr. Robert Buche & Dr. Jim Zhang
North Carolina State University

Abstract

The wireless industry is still developing techniques for power control for base stations. We model a wireless network as a system of queues under heavy traffic conditions. The primary concern is the allocation of nominal power (to handle the mean dynamics) and reserve power (to handle stochastic variation about the mean.) Several optimization problems based on different objectives and constraints were studied for the pre-allocation of nominal power. We suggest a method for quantifying the tradeoff between reserve and nominal power.

5.1 Motivation

The demand for wireless systems continues to increase worldwide. The industry is especially interested in the efficient use of power under the physical restraints of technology and cost. Much research has been conducted over the last two decades on efficient power allocation to meet the objective performance criterion for wireless systems. Total power to support a single base station (this is a simplified but useful model) serving a fixed number of remote units may be divided among:

- nominal preallocated power to serve the mean arrivals
- and reserve power to handle stochastic variations about the mean

Since there are few protocols for allocating the power efficiently for nominal and reserve usage, we will provide optimization methods with sensitivity analysis for pre-allocating nominal power to the base station under different objective functions. Heuristics and quantifications for assigning power are obtained via cost functions that assume the system is operating near its full capacity.

Suppose a system of k mobiles is served by a single base station with a dedicated queue for each mobile. Each queue can be in any one of the j vector channel states. Data movement can be represented as a process

[1]Auburn University
[2]University of Buffalo
[3]Texas Tech University
[4]NC State University
[5]Texas Tech University
[6]Clarkson University
[7]Mills College

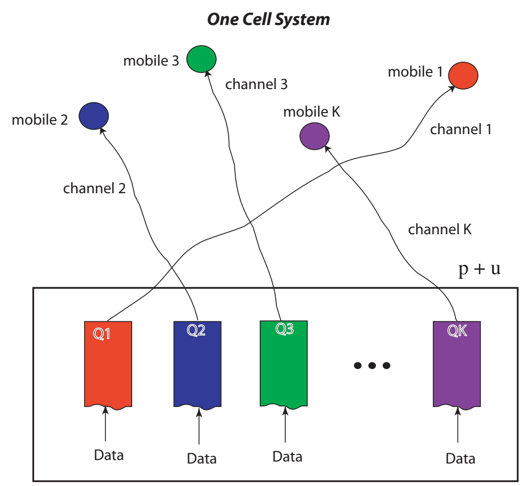


Figure 5.1: Wireless system.

of cumulative $A_i(t)$ arrivals and $D_i(t)$ departures where $1 \leq i \leq k$. Throughout we have assumed a fixed mean arrival rate destined for each mobile. The actual arrival rate can be considered as stochastic variation about the mean rates:

Departure rate is based on channel state and nominal power:

- $\bar{\lambda}_i^d$ is the specified mean departure rate per unit power for queue i
- π is the channel state stationary distribution
- $\bar{p}_i(j)$ is the nominal power needed to support the mean arrival rate for queue i in channel state j

The assumption in [1] requires that the system (base station) operate close to capacity, i.e., heavy traffic condition. The intuition for this assumption is that the underlying control problem (allocation of reserve

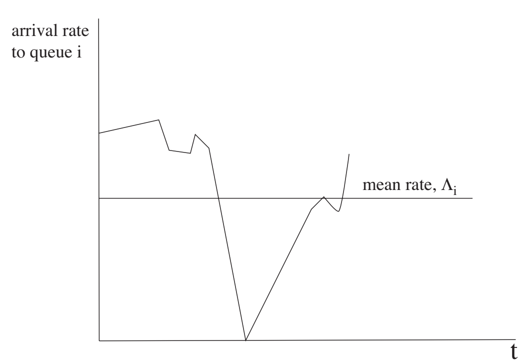


Figure 5.2: Arrival rate for queue i, $1 \leq i \leq K$.

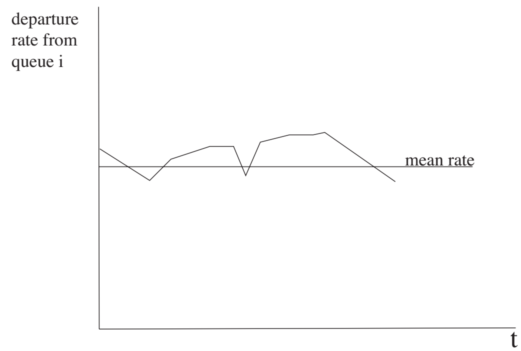


Figure 5.3: Departure rate for queue i, $1 \leq i \leq K$.

power) is too complicated for a direct solution. We need to average out some inessential details in the wireless communication system, such as multiple time scales, fast fading, etc., and approximate its queue dynamics using a limiting process. The heavy traffic approximation is a good candidate to take. Throughout the system, heavy traffic condition can be defined by the equality of mean arrival and mean departure rates:

$$\Lambda_i^a = \sum_j \bar{\lambda}_j^d \bar{p}_i(j)\pi(j)$$

5.2 Model 1

5.2.1 Efficient Allocation of Power for Mean Arrival Rates

In our first model, we search for an upper limit to the mean arrival rate given fixed total nominal power while operating under heavy traffic conditions. We add the term δ_i, the allowable increase above the baseline mean arrival rate, to the left side of the heavy traffic condition equation,

$$\Lambda_i^a + \delta_i = \sum_j \bar{\lambda}_i^d \bar{p}_i(j)\pi(j).$$

The allocation of δ_i to each queue i depends on the importance of the i^{th} mobile. So, we introduce a weight parameter d_i to signify the priority of the data of the i^{th} user. The solution to the optimal allocations of the δ_i's depends on the significance or weight parameter d_i. Thus, we maximize the following objective function(*profit*):

$$\sum_{i=1}^{k} d_i\delta_i = d_1\delta_1 + d_2\delta_2 \ldots + d_k\delta_k,$$

over $(\delta_1, \ldots, \delta_k)$ such that

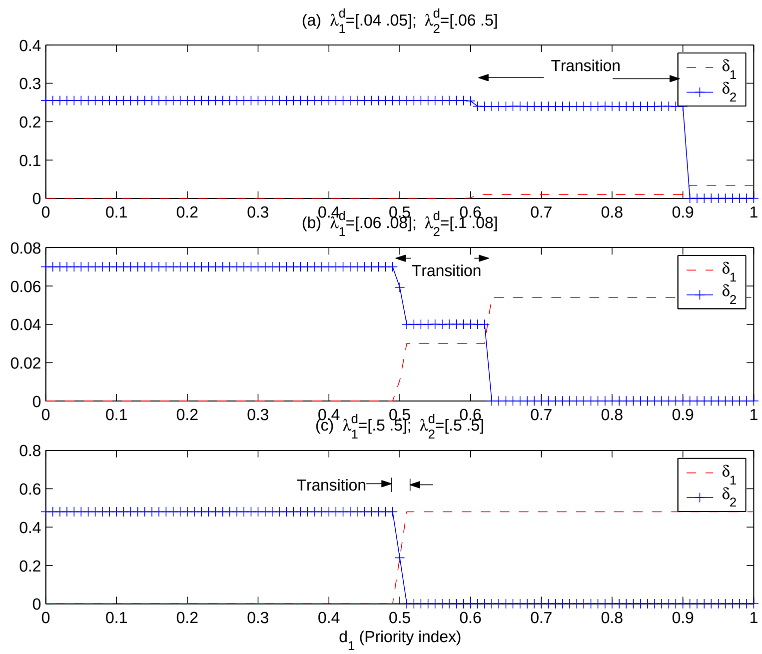


Figure 5.4: Sensitivity Analysis (δ_i w.r.t d_1.)

$$\sum_{i=1}^{k} \bar{p}_i(j) = c, \qquad \text{each } j$$

$$\sum_j \bar{\lambda}_i^d(j)\bar{p}_i(j)\pi(j) = \Lambda_i^a + \delta_i, \qquad \text{each } i$$

5.2.2 Results and Sensitivity Analysis

The simulations of Model 1 were carried out for 2 queues and 2 channel states with the following parameter values

$$\lambda^d = (\lambda_{ij}^d) = \begin{pmatrix} 0.06 & 0.08 \\ 0.1 & 0.08 \end{pmatrix}$$

$$\Lambda^a = (\lambda_i^a) = (0.01, 0.01) \quad \pi = (\pi_1, \pi_2) = (0.5, 0.5), \quad d = (0.5, 0.5) \quad \text{and} \quad c = 0.5$$

The optimal profit occurs when $\delta_1 = 0.0107$, $\delta_2 = 0.0593$, $p_1(1) = 0.0000$, $p_1(2) = 0.5179$, $p_2(1) = 1.0000$, and $p_2(2) = 0.4821$. The maximum profit under the above parameter values is 0.0350.

Figure 5.4 displays the effect of changing the priority index d_i on the values of δ_i. δ_1 increases with increasing d_1. Both queues are kept under heavy traffic conditions. Part (a): The departure rate per unit

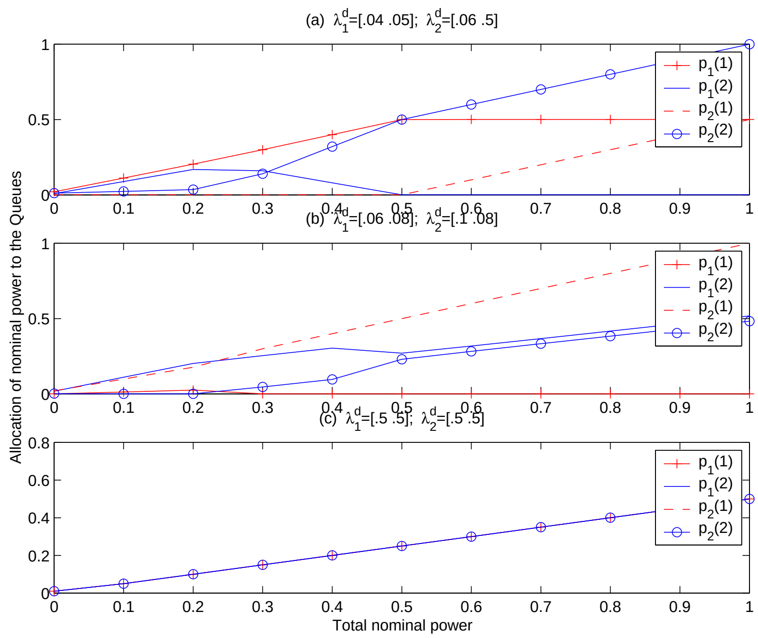


Figure 5.5: Sensitivity of power distribution with respect to nominal power .

power of the second queue in state 2, $\bar{\lambda}_2^d(2)$ is very large compared to the other queue states. Profit increases when δ_2 increases and d_1 is small (and d_2 is large). But as d_1 increases, there is a point where any further increase of δ_1 becomes more profitable. The transition point is delayed as the departure rate of the second queue increases. In part (b) and part (c), we reduced the difference between departure rates to illustrate the variations in the transition points. When two queues are weighted equally the transition occurs at $d_1 = 0.5$.

Figure 5.5 shows how the allocation of nominal power varies as the total nominal power changes. The relative values of the power distribution is most affected by departure rate. In graph (a), for channel state 1, there is little difference between the departure rates. As nominal power increases the power distribution varies to satisfy the conditions of heavy traffic and constant total power. In graph (b), the departure rate in the second queue in state 1, $\bar{\lambda}_2(1)$, is significantly greater than in the first queue in state 1, $\bar{\lambda}_1(1)$. So queue 2 requires less power than queue 1. In graph (c), for all values of total nominal power, equal rates of departure causes power to be allocated equally to support each queue.

Figure 5.6 shows the variation of profit $\sum_i d_i\delta_i$ with the total nominal power. As expected the profit function is increasing in total nominal power. More power allows for larger increases in mean throughput. The variations of the curve reflects the effects of the heavy traffic condition and the constant nominal power.

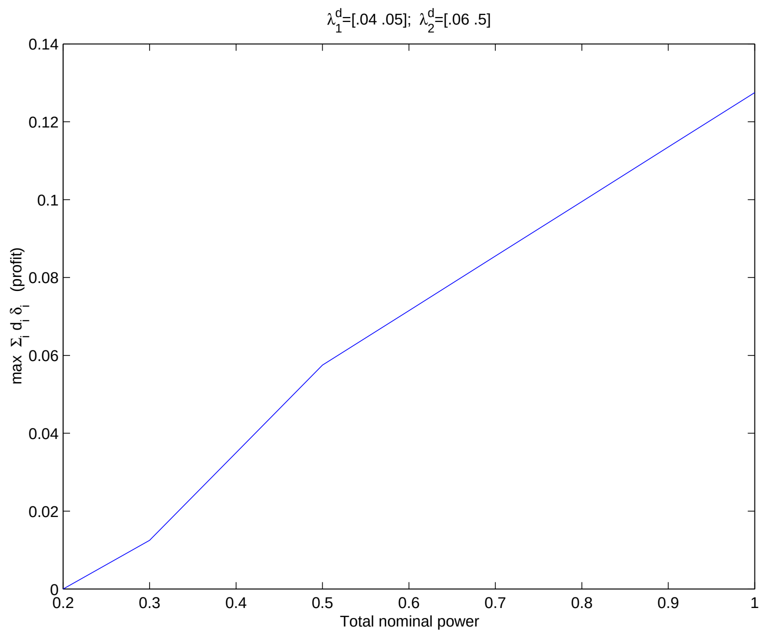


Figure 5.6: Sensitivity of "*Profit*"

5.3 Model 2

5.3.1 Description

In the first problem we restricted the total amount of nominal power, $\sum_i \bar{p}_i(j) = c$, for each j. In the second model we consider lower and upper bounds on the power assignment for queue, $\bar{p}_i(j)$. The lower bound can be viewed as a minimum service or quality requirement for each queue; the upper bound can be viewed as a physical limit on the available power. More specifically, the second model optimally allocates the power to each queue based on maximizing a given utility function f (the averaged utility over stationary distribution of the channel). We impose the heavy traffic condition as well as the upper and lower bounds. Formally, the optimization problem is described as

$$\max_{\bar{p}} \sum_{i,j} f(\bar{p}_i(j))\pi(j) \quad \text{s.t.}$$

$$\Lambda_i^a = \sum_j \bar{\lambda}_i^d(j)\bar{p}_i(j)\pi(j) \quad \forall i$$

$$L \leq \bar{p}_i(j) \leq U \quad \forall i, j.$$

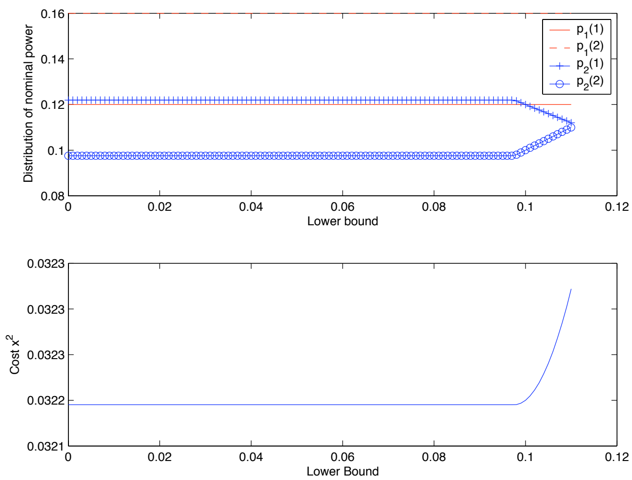


Figure 5.7: Lower bound of Problem 2

5.3.2 Results

Throughout this section we consider the case of two queues and two channel states and, unless otherwise stated, the following values are used:

$$\lambda^d = \begin{pmatrix} .06 & .08 \\ .1 & .08 \end{pmatrix}, \quad \Lambda^a = (.01, .01) \quad \text{and} \quad \pi = (.5, .5).$$

We consider two different functions f. First, we consider the cost function $f(x) = a(x-b)^2$ (in this case the goal is minimize cost). Of course both the shape parameter a and the shift parameter b affect the optimal value; only b affects the optimal solution, i.e. the allocation of the $\bar{p}_i(j)$. This makes sense since values of $\bar{p}_i(j)$ which are close to b in absolute value will minimize cost.

It is also useful to consider the effect of changing L and U, the lower and upper bounds. For this analysis, we consider $f(x) = x^2$ ($a = 1$, $b = 0$). In the unrestricted case, where there are no upper or lower bounds, the optimal solution is $\bar{p} = (.1200, .1600, 0.122, .0976)$ which corresponds to a cost of .0322. Figure 5.7 plots the effect of changing L on the allocation of nominal power and the minimum cost (here $U = .2$ and L ranges from 0 to .11). As expected the cost function is increasing in the lower bound L; as L increases the feasible region is smaller. In fact, the lower bound does not affect the decision variables until L is about .1 at which point the cost is strictly increasing in L (the optimal unrestricted solution has each of the decision variables above .0976). There is a similar response in the allocation of nominal power. Queue 1's allocation is unaffected (the unrestricted optimal allocation is above .11). In the case of queue 2, $p_2(2)$ is forced to increase by the lower bound and, to maintain the heavy traffic condition, $p_2(1)$ decreases. With $U = .2$, L cannot be increased above .11 or the problem becomes infeasible. Figure 5.8 is the same for changing U and the analysis is analogous (here $L = .005$ and U ranges from .15 to 1). Figure 5.7 and 5.8 concern problem 2 in the case of the cost function $f(x) = x^2$. The plots show the effect of changing the lower bound L and upper bound U respectively on the allocation of nominal power and on the minimum cost.

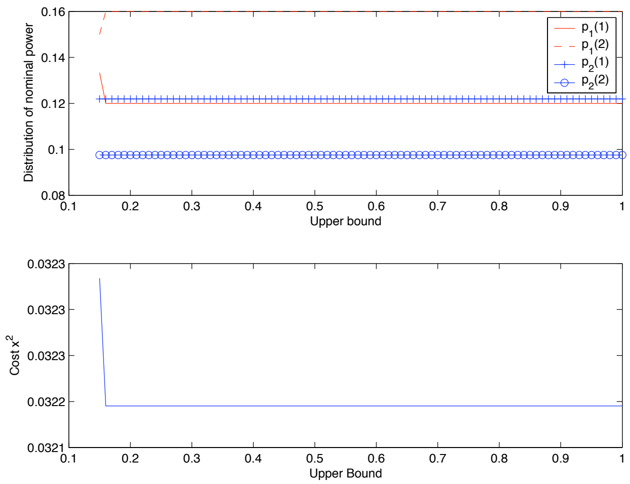


Figure 5.8: Upper bound of Problem 2

Next we consider maximizing the function $f(x) = log(1 + x) - \frac{x}{b}$, $b > 1$. In this case, f is a net utility function, where $\log(\bar{p}_i(j))$ represents the utility obtained from using the power and $\frac{\bar{p}_i(j)}{b}$ represents the cost of using the power. Throughout this analysis $b = 10$. The table below shows the effect of decreasing the possible values of $\bar{p}_i(j)$ (i.e. increasing L and decreasing U). In the table $\bar{p} = (\bar{p}_1(1), \bar{p}_1(2), \bar{p}_2(1), \bar{p}_2(2))$. Of course the optimal value is increasing in $U - L$. Also notice that $U - L$ becomes relatively small before the region becomes infeasible.

(L, U)	$\bar{p}$	optimal value
(0,100)	(0.3100, 0.0175, 0.0113, 0.2358)	0.2265
(.015, .22)	(0.2200, 0.0850, 0.0240, 0.2200)	0.2240
(.025, .21)	(0.2100, 0.0925, 0.0320, 0.2100)	0.2234
(.035, .2)	(0.2000, 0.1000, 0.0400, 0.2000)	0.2226
(.045,.19)	(0.1900, 0.1075, 0.0480, 0.1900)	0.2217
(.055, .18)	(0.1800, 0.1150, 0.0560, 0.1800)	0.2206
(.065, .17)	(0.1700, 0.1225, 0.0650, 0.1688)	0.2194
(.075, .16)	(0.1600, 0.1300, 0.0750, 0.1563)	0.2180
(.085, .15)	(0.1500, 0.1375, 0.0850, 0.1437)	0.2164
0.0950 0.14	infeasible	infeasible

5.4 Model 3: Tradeoff between Nominal and Reserve Power

5.4.1 Description

We consider the tradeoff between the nominal power and reserve power, which are introduced in [1]. While nominal power supports the mean arrival rate, reserve power is applied to stochastic variations above the mean, such as burst arrivals.

In section one, we maximized the utility for fixed nominal power. We increased the arrival rates for each queue as much as possible via a linear programming model. For fixed reserve power, the optimization model turns out to be a stochastic control problem. In Buche and Kushner [1], the reserve power becomes the control variable for allocating the reserve power to queues in different channel states. We refer the interested reader to [1].

Roughly speaking, if our queueing system is operating under a heavy traffic condition (near the system capacity), the system can be imbedded in a parameterized sequence indexed by n. As $n \to \infty$, one can apply weak convergence techniques to the queue dynamic equation to obtain the limiting process, which is a stochastic differential equation with reflection:

$$x_i(t) = x_i(0) + \int_0^t b_i(u, x(s))ds + w_i(t) + z_i(t) \tag{5.1}$$

where $b_i(u,x) = -\sum_j \bar{\lambda}_i^d(j)u_i(j,x)\Pi(j)$, $z_i(t)$ is called reflection term and, simply speaking, it is the correction term for the "fictitious" departure in the idle time for the queueing system.

Solving the continuous time stochastic control problem gives the optimal reserve power subject to (5.1). So we minimize the cost function:

$$\begin{aligned} &W(x,u) = E_x \int_0^\infty e^{-\beta s} \left[\sum_i \alpha_i x_i^{p_i}(s)ds + c'(x(s))dz(s) \right] \\ &\beta > 0 \text{ and small, } \alpha_i > 0, p_i > 1 \end{aligned} \tag{5.2}$$

where $c'(\cdot)$ specifies the boundary cost. In our case, since the queue dynamics are described by SDER (Stochastic Differential Equation with Reflection), it is difficult to find an analytical solution for this kind of continuous-time stochastic control problem. The well known numerical method for the continuous time stochastic control problem was developed by Kushner and his coworkers [3]. The basic idea is that the original controlled diffusion process (i. e. the queue dynamics equation) is approximated by a discrete Markov chain with parameter h, where each h corresponds to a time interval $\Delta t^h(u,x)$, giving the time between the jumps of the Markov chain, and an associated cost function V^h. This approximating Markov chain is chosen such that certain "*local*" properties of it are analogous to those of the original controlled process. A very direct method to achieve this goal is substituting the finite-difference approximation into the differential operator of the (unreflected) process to get the state transition probabilities for the Markov chain and $\Delta t^h(u,x)$. Then a discrete-time dynamic programming equation for which the desired computation can be carried out for the cost function V^h is obtained.

Now define total available power in the base station is P_{total}, Hence

$$\bar{P}_{\text{nominal}} + u = P_{\text{total}}$$

5.4.2 Results

Our numerical experiment is based on Table 5.1:

As we can see in Figure 5.9 and Figure 5.10, when we increase the reserve power, the cost will decrease at the beginning, and the profit increases when we increase the nominal power. However, these two criteria are not same. It is hard to choose an actual nominal power and reserve power for our system, but we can see some trends here.

Table 5.1: Tradeoff of Nominal and Reserve Power

Nominal Power	Profit	Reserve Power	Cost
0.5	.0125	1.0	3813.8
1	.035	1.5	3410.0
1.5	.0575	2.0	2774.7
2	.08	2.5	2184.3
2.5	.1025	3.0	1869.8
3	.1250	3.5	1877.2
3.5	.1475	4.0	2105.6
4	.17	4.5	2449.6
4.5	.1925	5.0	2853.2
5	.215	5.5	3292.7

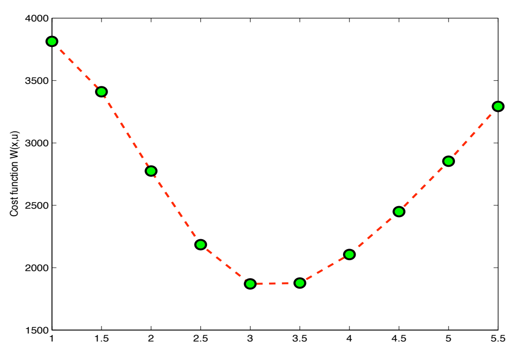


Figure 5.9: Cost vs. Reserve Power

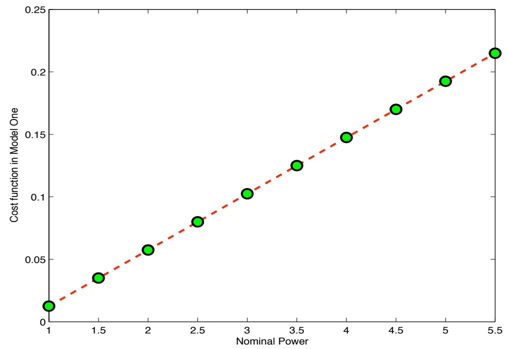


Figure 5.10: Profit vs. Nominal Power

5.5 Extension Model 4: Admission Control Problem

Here we are interested in determining the list of the users to be added given the existing power resources and the heavy traffic conditions.

Consider the scenario of k existing users in the system with an associated Λ_i^a such that

$$\Lambda_i^a < \sum_j \bar{\lambda}_i^d(j)\bar{P}_i(j)\pi(j) \quad i \leq K$$

$$\sum_i p_i(j) \leq c \quad \forall j$$

and N possible users in the admission pool, having predefined priorities or weights α_i where $i \in \{1, 2, \ldots N\}$. To minimize the cost of allocating power to the new users, we decide which user(s) to add so that profit will be maximized. This problem could be viewed as utilizing the unused nominal power to accommodate new users optimally without disturbing the constraints.

The cost function can be defined or modeled as follows:

$$\sum_i \delta_i \alpha_i \sum_j f(p_i(j))\pi(j) \; i \in (1, 2, \ldots, N) \; j \in (1, 2, \ldots, J),$$

where α_i is the priority weight of the new user to be added, δ_i is the decision variable of user i which is a indicator function, $g(.)$ can be thought as the utility function.

The constraints to be applied are

- Heavy Traffic Condition

$$\delta_i \Lambda_i \leq \sum_j \lambda_i(j) p_i(j) \pi(j) \quad \forall i \in (1, 2, \ldots, K + N)$$

- Upper limit of available nominal power

$$\sum_i \delta_i p_i(j) \leq C \quad i \in (1, 2, \ldots, K+N)$$

This turns out to be a mixed-integer programming problem. We don't solve it here due to the time limitation. However, it is interesting to consider it in future work along this direction.

5.6 Conclusions

To satisfy a wireless company's different optimization goals, we assumed for simplicity heavy traffic conditions, modeling base station traffic as a system of arrivals and departures. We formed different objective functions for our four models. We found upper and lower bounds for nominal power to support the allowable increase in mean arrival rates. In seeking the tradeoff between nominal power and reserve power, we discovered a need for a new model that combines both into a single objective function. Further analysis and modeling of the brief description of the approach to admission control is still needed.

5.7 Acknowledgments

We owe thanks to Dr. Robert Buche, Dr. Priya Ranjan and Dr. Jim Zhang, our faculty consultants for suggesting improvements, and clarifying the message. We appreciate the support of NSF, SAMSI, IMSM and all the North Carolina State University faculty and staff.

Bibliography

[1] Buche, R. and Kushner, H. J. (2002) Control of Mobile Communications With Time-Varying Channels in Heavy Traffic, *IEEE Transactions on Automatic Control*, 47:992-1003.

[2] Hooker, J. (2000) *Logic-Based Methods for Optimization,* Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley.

[3] Kushner, H. J. and Dupuis, P. (2001) *Numerical Methods for Stochastic Control Problems in Continuous Time*, $2^n d$ Edition, Springer.

[4] Luenberger, D. (1973) *Introduction to Linear and Nonlinear Programming*, Addison-Wesley Publishing Company, New York.

Chapter 6

Modeling of a Fluid Pump Driven by a Piezoceramic Disk Actuator

PROBLEM 6: MODELING OF A FLUID PUMP DRIVEN BY A PIEZOCERAMIC DISK ACTUATOR

Chadia Affane Aji[1], Tom Braun[2], Seshadri Damu[3], Thomas Deems[4], Ingunn Gunnarsdottir[5], Sidharth Rupani[6], Necibe Tuncer[7]

Problem Presenter:
William Ott
PAR Technologies LLC

Faculty Consultants:
Dr. William Oates & Dr. Ralph Smith
North Carolina State University

Abstract

A system dynamic model of a miniature piezoelectric pump designed by PAR Technologies was developed to predict flow performance at multiple frequencies. The pump consisted of a circular piezoelectric diaphragm, Reed valves and inlet and outlet tubes. Lumped parameters were determined for the electro-mechanical-fluid coupled system using experimental results and theoretical analyses. The piezoelectric diaphragm was modeled as a lumped element mass-spring damper with external forces generated by the voltage input and pump chamber fluid pressure. The edge constraints on the piezoelectric diaphragm were included in the lumped element model to assess changes in flow rate under simply supported or fully clamped boundary conditions. Fluid inertial effects were evaluated in the pump tubing which resulted in enhanced performance at certain operating frequencies. Reasonable flow rate predictions were obtained in comparison to experimental results of the pump developed by PAR Technologies.

6.1 Introduction and Motivation

Piezoelectric materials develop a potential when subjected to a mechanical stress, and conversely deform when exposed to an electric field. This material behavior has been exploited in a myriad of aeronautical, biomedical, industrial and military applications. Here we consider lead zirconate titanate (PZT), which provides an approximately linear response for low to moderate drive levels and precise positioning over operating frequencies ranging from fractions of Hertz (Hz) to tens of kHz. This material is utilized in a wide variety of applications such as accelerometers, acoustic emission transducers, actuators, flow meters, medical imaging, sonar transducers, and many others. In pumps, PZT is typically employed as either a stack actuator or a diaphragm. The stack actuators are used in hydraulic applications requiring large force such as morphing air

[1]Auburn University
[2]North Carolina State University
[3]University of Tennessee
[4]Slippery Rock University
[5]Roskilde University
[6]Worcester Polytechnic Institute
[7]Auburn University

craft control surfaces, while diaphragms are used in applications where larger flow rate and lower pressure are important such as electronic cooling systems.

A system dynamic model was developed to predict flow rate of the miniature piezoelectric pump developed by PAR Technologies. The pump utilizes Ruggedized Laminated Piezo (RLP) actuators which are employed in a broad range of applications including piezoelectric valves, electronic cooling, drug infusion, and vibratory fans for laminar air flow. In the miniature pump, the RLP disk actuator is used as the driving diaphragm for pumping fluids. The diaphragm is the only actively controlled component of the pump, which also consists of a pair of passive one-way valves controlling fluid flow into and out of the pump chamber. These pumps have a number of applications including cooling systems and drug infusion. While considerable success has been achieved in developing high performance flow rates, a strong understanding of the underlying dynamics is lacking. Therefore in this work, a system dynamic model is developed to identify key parameters associated with flow rate in the piezoelectric pump.

6.2 System Dynamic Model

A schematic illustrating the lumped parameter model of the miniature pump system is shown in Figure 6.1. A pumping chamber containing the PZT diaphragm (which is held within the pumping chamber by o-rings, although not shown in the figure) as well as inlet and outlet valves is attached by tubes to source and sink reservoirs. The source and sink reservoirs serve as constant pressure sources at atmospheric pressure. An electric potential applied to the actuator deforms the piezoelectric diaphragm, which induces a bending force in the composite actuator. An alternating voltage applied to the diaphragm causes the plate to deform and return to its original shape, displacing a volume of fluid per stroke. A pair of Reed valves rectify fluid flow by pulling fluid into the pressure chamber through the inlet valve and out of the pressure chamber through the outlet valve.

For modeling purposes, this pump is divided into the following subsystems:

- The electromechanical subsystem models the displacement of the piezoelectric actuator using a mass-spring-damper system. The volume of fluid displaced per stroke is determined from classic linear elasticity solutions of a flat circular plate. This approach allows for an assessment of clamping effects of the o-rings.

- The fluid subsystems model flow through the inlet and outlet tubes, within the pump chamber, and across the valves by employing analogies from electrical circuit theory. Each lumped element component is assigned an effective flow resistance and fluid capacitance. Additionally, the effects of fluid inertia is considered in the tubing.

These two subsystems are coupled (the output of one system forms an input to the other) in the following manner: the applied electric potential deforms the piezoelectric actuator which forces fluid into and out of the pump chamber, and the pressure in the pump chamber exerts a resistive force on the piezoelectric acuator. These two subsystems and their governing equations are discussed in detail in the following sections.

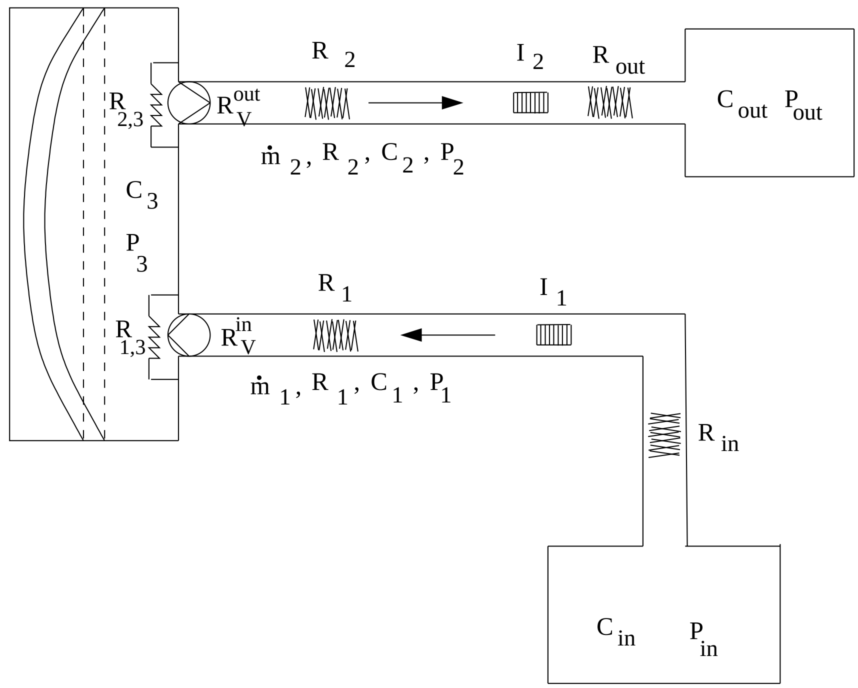


Figure 6.1: System dynamic model of the piezoelectric pump.

6.2.1 Electromechanical Subsystem

The electromechanical subsystem of the pump consists of a circular actuator fixed between a pair of o-rings, as shown in Figure 6.2(a). The actuator consists of a PZT wafer laminated to a stainless steel substrate. For the model developed in this paper, the effects of the laminate, substrate, and PZT are lumped together to provide a single effective element for the actuator. The motion of the actuator is modeled as a flat circular plate in the unforced state. This is an approximation, since mismatch in the thermal expansion coefficients of each composite layer introduces some curvature, but the curvature is assumed to be small compared to the overall actuator displacement. In addition, the o-rings will introduce clamping on the actuator, but are also compliant which allows some movement of the plate edges. To incorporate this behavior, we introduce the variable γ to assess the clamping effect of the o-rings, where $0 \leq \gamma \leq 1$. A value of $\gamma = 0$ signifies the plate is simply supported, while $\gamma = 1$ means the plate is perfectly clamped. All forces on the plate are assumed to be uniformly distributed and act only in the normal direction.

According to Timoshenko [5], the displacement at radial point r of a uniformly loaded simply supported plate is given by

$$w(r) = \frac{q(a^2 - r^2)}{64D} \left(\frac{5+\nu}{1+\nu} a^2 - r^2 \right), \tag{6.1}$$

where a is the radius of the plate, D is the flexural rigidity, q is the load intensity, and ν is Poisson's ratio.

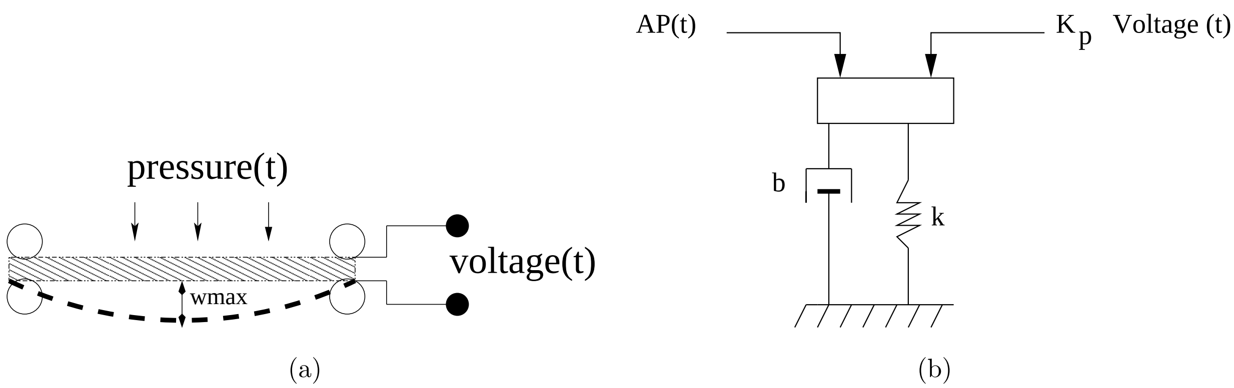


Figure 6.2: Diagram of electromechanical subsystem. (a) Cross-sectional schematic of the circular piezoelectric actuator. (b) Effective lumped element representation of the actuator.

Thus,

$$w_{max} \;=\; w(0) \;=\; \frac{(5+\nu)qr^4}{64(1+\nu)D} \tag{6.2}$$

gives the maximum displacement for static loading. Substituting (6.2) into (6.1) gives

$$w(r) \;=\; w_{max}\left[1 - \frac{r^2}{a^2} - \frac{(1+\nu)r^2}{(5+\nu)a^2} + \frac{(1+\nu)r^4}{(5+\nu)a^4}\right]. \tag{6.3}$$

Integrating this relation over the surface of the actuator gives the volume displaced for the simply supported case as

$$\Delta V_{ss} \;=\; \frac{\pi a^2(7+\nu)}{3(5+\nu)} w_{max}.$$

Considering a perfectly clamped plate and following the same integration gives the volume displaced as

$$\Delta V_{clp} \;=\; \frac{\pi a^2}{3} w_{max}.$$

For the volume displaced by the actuator, a weighted average is taken of these two volumes, using γ as the weight. This gives

$$\Delta V \;=\; \left(\gamma + (1-\gamma)\frac{7+\nu}{5+\nu}\right)\frac{\pi a^2}{3} w_{max} = H w_{max}. \tag{6.4}$$

Whereas the volume given in (6.4) is derived for the static case, the shape of the deformations is assumed to be the same in both dynamic and static loading, which implies only the center point needs to be considered under dynamic loading. The center point is modeled as the harmonic oscillator

$$m\ddot{w}_{max}(t) + b\dot{w}_{max}(t) + kw_{max}(t) \;=\; F(t), \tag{6.5}$$

where m is the effective mass of the disk, b is the Kelvin-Voigt damping coefficient of the disk, k is the effective stiffness, and F gives the forces acting on the actuator. This gives the effective system depicted in Figure 6.2(b). The force due to applied voltage is assumed to be linear across the operating regime considered here. This gives the total force acting on the disk as

$$F(t) \;=\; -K_p\phi(t) - \pi a^2 P(t), \tag{6.6}$$

where K_p is the linear piezoelectric coefficient relating voltage to force, ϕ is the voltage, and P is the pressure acting on the disk. The material dependent parameters m, b, k, K_p, and ν were estimated from known data as detailed in Section 6.3. Note that a positive electric potential will deform the actuator away from the valves, which correlates with the pump developed by PAR.

6.2.2 Fluid Subsystems

The fluid subsystems are modeled using an electrical circuit analogy with resistance, capacitance, and inertia (inductance) elements. The potential in the system is a pressure differential, which is analogous to voltage. The equivalence to RLC circuit components are given as follows:

- Resistance is defined as the change in potential required to make a unit change in flow rate.
- Capacitance is the change in quantity of liquid required to cause a unit change in pressure.
- Inertia is the change in potential required to make a unit rate of change in flow rate.

Thus, capacitance and inertia store energy, while resistance dissipates energy. We derive the governing equations for the fluid subsystem by using fundamental fluid equations and related analogies from electrical circuits.

The fluid dynamics within the pump chamber is modeled using the continuity equation

$$\dot{m}_i - \dot{m}_o \;=\; \frac{d(\rho V)}{dt}, \tag{6.7}$$

where $\dot{m}_i$ is the mass flow rate through the inlet valve, $\dot{m}_o$ denotes the mass flow rate through the outlet valve, ρ is the density of the fluid, and V is volume of the pump chamber. The bulk modulus of the fluid β_f is defined as

$$\beta_f \;=\; \rho_o \frac{dP}{d\rho}, \tag{6.8}$$

where ρ_o gives the average fluid density in the chamber and P is the pressure in the pump chamber. Combining (6.7) and (6.8) gives

$$\frac{\rho_o V_o}{\beta_f}\dot{P} \;=\; C\dot{P} \;=\; \dot{m}_i - \dot{m}_o - \rho_o \dot{V}, \tag{6.9}$$

where the mass flow term $\rho_o \dot{V}$ acccounts for the rate of volume change in the pump chamber as the piezoelectric actuator deforms.

The ambient pressure is defined as zero, therefore the mass flow through the outlet valve is

$$\frac{P}{R} = \dot{m}_o,$$

where an effective resistance R accounts for viscous effects within the pump chamber and across the Reed valves. Similar relations hold for the inlet valve. Substituting the mass flow rate relations into (6.9), we get the following first order differential equation governing the fluid dynamics in the pump chamber

$$RC\dot{P} + P = R\dot{m}_i. \tag{6.10}$$

In the present analysis, the fluid capacitance is based on bulk modulus and density of water. This neglects compliance within the pump chamber structural housing and o-rings. In comparison, elastic effects in the tubes were included in the effective fluid capacitance associated with the inlet and outlet tubes. The fluid capacitance in the tubes is defined by

$$C_1 = \frac{\rho V_1}{\beta_{1,2}}$$

and

$$C_2 = \frac{\rho V_2}{\beta_{1,2}},$$

where $V_1 = \pi r_1^2 l_1$, $V_2 = \pi r_2^2 l_2$, ρ is the density of the fluid, r_1 is the radius of the inlet tube, r_2 is the radius of the outlet tube, l_1 is length of the inlet tube, and l_2 is the length of the outlet tube. The effective bulk modulus of the tubes $\beta_{1,2}$ is given by [3]

$$\beta_{1,2} \quad \frac{E}{2}\left[\frac{d_i^2 + d_o^2}{d_o^2 - d_i^2} + \nu\right]^{-1}.$$

Here E is the modulus of elasticity, ν gives Poisson's for the ratio tube, d_i is the inner diameter, and d_o is outer diameter. This allows the effective bulk modulus of the tubes and fluid

$$\frac{1}{\beta_{tube}} = \frac{1}{\beta_{1,2}} + \frac{1}{\beta_f}$$

to be computed, where β_f is the bulk modulus of the fluid.

In the hydraulic tube, flow resistance is characterized by tube geometry, viscosity, and density [2] as given by

$$R = \frac{128\mu l}{\pi\rho D^4},$$

where R is the resistance, μ is the viscosity of the fluid, ρ is the fluid density, l is the length of the tube, and D is the diameter of the tube. In our case, we label the inlet tube resistance R_1 and the outlet tube resistance R_2.

Part of the flow resistance of the Reed valves is modeled as drag across a flat plate. The total drag force is given as

$$\begin{aligned}\text{Total Drag force} &= \text{Coefficient of Drag} \times \text{Dynamic Pressure} \times \text{Valve Plate Area}\\ &= C_d\left(\frac{1}{2}\rho U^2\right)S,\end{aligned}$$

where U is the velocity of the flow and S is the valve plate area. The total drag force can also be written as pressure difference across the valve (ΔPS). We combine the previous two equations and introduce volume flow rate $Q = SU$ to get

$$\Delta P \;=\; \frac{C_D\rho}{2S^2}Q^2.$$

The flow resistance in the Reed valves (R_v^{in} and R_v^{out}) is given by the slope of $\frac{\Delta P}{Q}$.

Additional flow resistance is included in the pump chamber which is associated with flow loss from the change in diameter of the inlet/outlet tubes and the pump chamber. The pressure drop between the pump chamber and the inlet and outlet valves is

$$\Delta P \;=\; \frac{\rho Q_1^2}{2}\left((1+K_L)\frac{1}{A_1^2} - \frac{1}{A_p^2}\right),$$

where Q_1 is the volumetric flow rate, K_L is a loss coefficient, A_1 and A_p give the cross-sectional area of the valve opening and pump chamber, respectively. This is derived from Bernoulli's equation to account for head loss from flow around the internal pump geometry. The resistance is determined from the slope of $\Delta P - Q$. The flow resistance on the inlet side is denoted by $R_{1,3}$ and on the outlet side $R_{2,3}$.

Whereas only first order effects are included in the pump chamber, the mass of the fluid in the tubes requires considering inertial effects. These effects in the input and output tubes are governed by

$$CI\ddot{P} + RC\dot{P} + P \;=\; R\dot{m}_i, \tag{6.11}$$

where the inertial term $I = \frac{\rho L}{A}$ is a function of the fluid density ρ, the tube length L, and tube cross-sectional area A.

The dynamic equations governed by (6.5), (6.10) and (6.11) are formulated in a matrix system that corresponds to the piezoelectric actuator, pump chamber, and tubing subsystems. In addition, the fluid reserviors on the inlet and outlet sides of the system were governed by (6.5) with large capacitance values to ensure constant pressure sources.

The overall system is

$$m\ddot{w}_{max} + b\dot{w}_{max} + kw_{max} \;=\; -K_p\phi - P_3A \tag{6.12}$$

$$C_3\dot{P}_3 \;=\; \rho\dot{V} + \frac{P_1 - P_3}{R_1 + R_v^{in} + R_{1,3}} - \frac{P_3 - P_2}{R_2 + R_v^{out} + R_{2,3}} \tag{6.13}$$

$$C_1I_1\ddot{P}_1 + C_1(R_1 + R_v^{in} + R_{1,3})\dot{P}_1 + P_1 \;=\; \frac{R_1 + R_v^{in} + R_{1,3}}{R_1}(P_{atm} - P_1) - (P_1 - P_3) \tag{6.14}$$

$$C_2I_2\ddot{P}_2 + C_2(R_2 + R_v^{out} + R_{2,3})\dot{P}_2 + P_2 \;=\; (P_2 - P_{atm}) + (P_3 - P_2) \tag{6.15}$$

$$C_{in}\dot{P}_{in} \;=\; \frac{-(P_{in} - P_1)}{R_{in}} \tag{6.16}$$

$$C_{out}\dot{P}_{out} \;=\; \frac{P_2 - P_{out}}{R_out}. \tag{6.17}$$

The system of equations in matrix form is

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\phi, \tag{6.18}$$

where

$$\mathbf{x} = \begin{bmatrix} w_{max} & \dot{w}_{max} & P_3 & P_1 & \dot{P}_1 & P_2 & \dot{P}_2 & P_{out} & P_{in} \end{bmatrix}^T, \tag{6.19}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-K}{m} & \frac{-b}{m} & \frac{-A}{m} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\rho H}{C_3} & \frac{-R_{eq}}{C_3} & \frac{1}{C_3 R_{v1}} & 0 & \frac{1}{C_3 R_{v2}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{C_1 I_1} & \frac{-(R_1+R_{v1})}{C_1 I_1 R_1} & \frac{-R_{v1}}{I_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{C_2 I_2} & 0 & 0 & \frac{-(R_2+R_{v2})}{C_2 I_2 R_2} & \frac{-R_{v2}}{I_2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{C_{out} R_{out}} & 0 & 0 & 0 & \frac{-1}{C_{in} R_{in}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{C_{out} R_{out}} & 0 & 0 & \frac{-1}{C_{out} R_{out}} \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 0 & \dfrac{-K_P}{m} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T, \tag{6.20}$$

and

$$\begin{aligned} \dot{V} &= H\dot{w}_{max}, \\ R_{eq} &= \frac{1}{R_1 + R_v^{in} + R_{1,3}} + \frac{1}{R_2 + R_v^{out} + R_{2,3}}, \\ R_{v1} &= R_1 + R_v^{in} + R_{1,3}, \\ R_{v2} &= R_2 + R_v^{out} + R_{2,3}. \end{aligned}$$

6.3 Testing and Simulation

Data for the pump designed and built by PAR Technologies is used to verify the model. The radius of the actuator is 8.5 mm, which accounts for the o-ring at the edges. For the electromechanical subsystem, the parameter k is estimated from measurements of dome height of the PZT disk versus pressure load. The natural frequency (given by PAR as 3.5 kHz) and the value of k is used to approximate the mass. The parameter b is taken from these parameters by assuming a damping ratio of 0.2. This gives values of $k = 7.479 \times 10^5$ N/m, $b = 13.398$ Ns/m, and $m = 0.0015$ kg. The constant K_p is estimated by considering (6.5) in static equilibrium and with negligible pressure, which implies $kw(t) = K_p\phi(t)$. The value of $\frac{w(t)}{\phi(t)}$ is approximated experimentally, giving $K_p = 0.1977$ N/V. For ν, the Poisson's ratio of PZT is used, which is 0.34. These values are refined by the Nelder-Mead optimization algorithm [1] to obtain values that more closely fit the measured voltage versus displacement data. This process allows the effective parameters to accommodate measurement error in the data used for the initial estimation, as well as some modeling error. The value of k is held constant, and the other values given by Nelder-Mead are $b = 14.514$ Ns/m, $m = 0.0014$ kg, and $K_p = 0.1861$ N/V. These

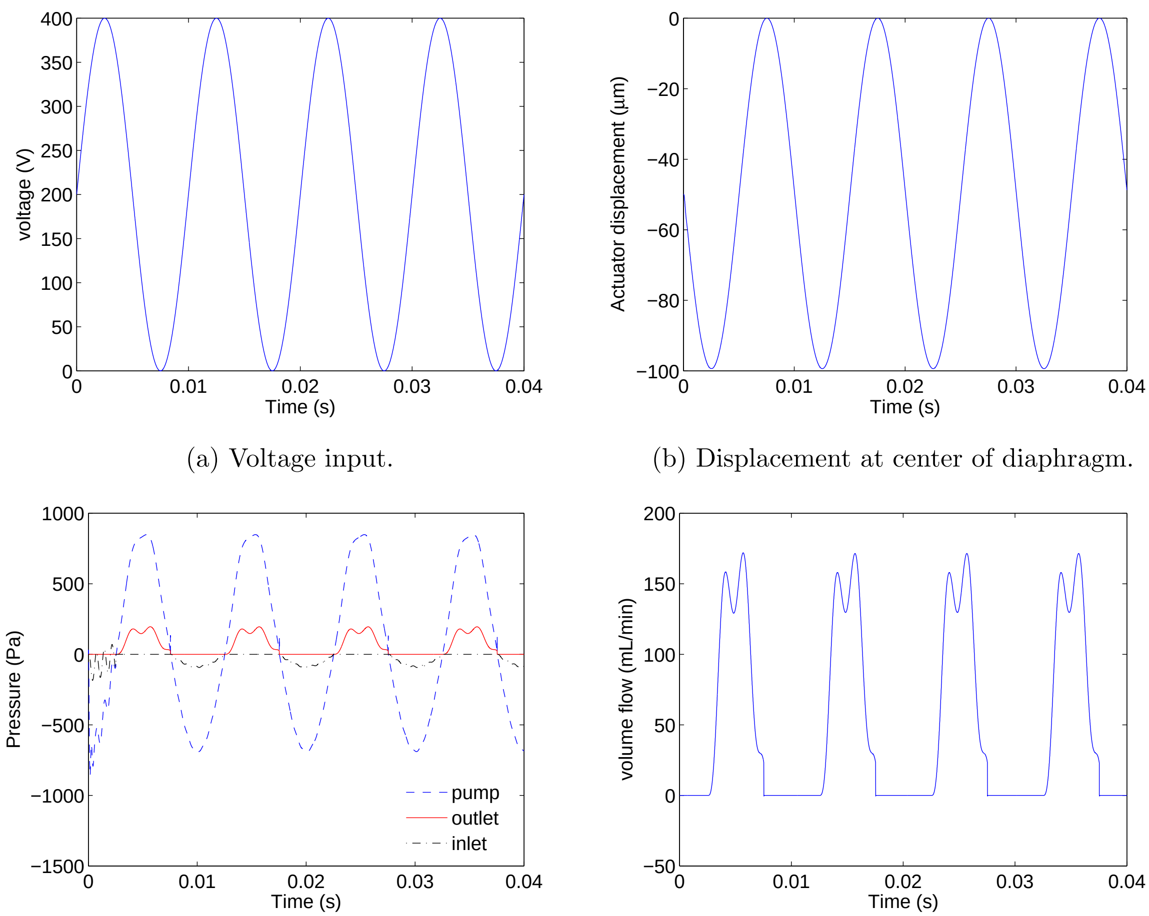
Parameter	Value
Actuator stiffness	74790 N/m
Actuator damping	14.514 Ns/m
Actuator mass	1.4 g
Poisson's ratio for actuator	0.34
K_p	0.1861 N/V
Tube diameters	3.175 mm
Inlet tube length	76 mm
Outlet tube length	152 mm
Modulus of elasticity (tube)	1 GPa
Poisson's ratio for plastic tube	0.5
Nominal pump chamber height	0.23 mm
Valve length	4.42 mm
Valve width	2.24 mm
Coefficient of drag for valve	1.2
Head loss, K_L	0.5
Fluid Bulk Modulus	2.15 GPa

Table 6.1: Parameters used to in the system dynamic model.

were found to reduce the error between predicted and measured displacements by approximately a factor of 2. For the fluid subsystem, some parameters were estimated from data provided by PAR, while others required theoretical analysis. The parameters used in the simulations are given in Table 1.

A simulation illustrating internal pump dynamics is illustrated in Figure 6.3 using a sinusoidal input of 400 V peak to peak at 100 Hz. This resulted in the actuator displacement shown in Figure 6.3(b). Since the actuator was assumed to be linearly related to voltage, we see that the actuator follows the voltage curve with a peak to peak displacement of 100 microns. The pressure changes caused by the actuator are shown in Figure 6.3(c). The pressure in the outlet tube is never negative, but increases as the diaphragm pushes fluid out. Likewise, the input tube pressure drops temporarily as fluid flows into the pump chamber, but never increases to a positive value. This clearly illustrates the effects of the valves. Finally, Figure 6.3(d) shows flow rate versus time, which follows the output tube pressure as expected.

The flow rate versus frequency is shown in Figure 6.4. The effect of a fully clamped actuator demonstrates loss in flow rate as expected. The unclamped simulation refers to the simply supported edge condition. Additionally, the system dynamic model is compared to flow rates based on volume displaced times the operating frequencies (linear approximations in Figure 6.5). This neglects flow loss, system compliance and inertial effects. The inertial effects are responsible for the increased flow over certain operating frequencies.



(a) Voltage input.

(b) Displacement at center of diaphragm.

(c) Pressure of pump chamber and tubes.

(d) Volumetric flow rate.

Figure 6.3: Displacement, pressure, and volume flow predicted by the system dynamic model.

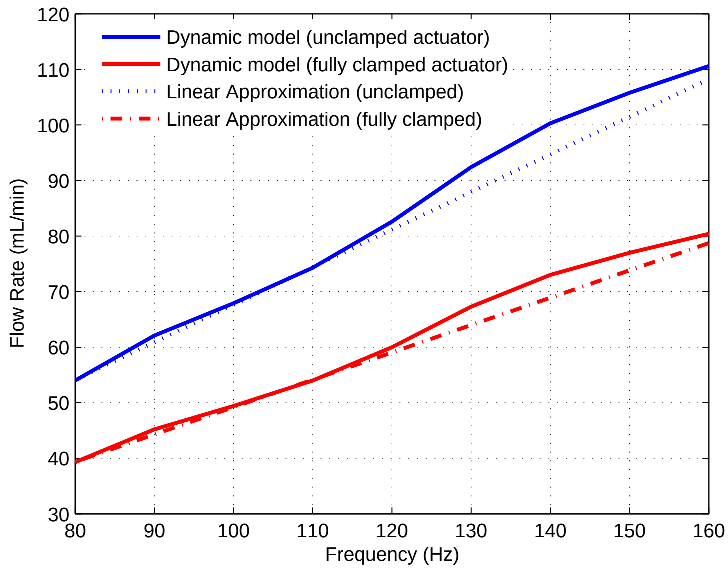


Figure 6.4: Average flow rate versus frequency at the applied peak to peak voltage 400 V.

In general, a reasonably qualitative estimate of flow rate is achieved in comparison to data provided by PAR Technologies. For example, measured flow rate at 80 Hz was 70 mL/min and at 160 Hz measured flow rate was 170 mL/min. This results in an average error of 28% in comparison to the simply supported actuator model. There are several potential reasons for the discrepancy in model performance and experimental results. Fluid inertial effects in the pump chamber were neglected which may enhance flow rate at the operating frequencies considered. In comparison, the lowest resonant frequency for the system was theoretically calculated to be 270 Hz which was associated with resonant effects in the outlet tube. The effect of this resonant frequency on flow enhancements at lower frequency requires additional work. Other unmodeled dynamics such as nonlinear ferroelectric consitutive behavior and variations in the deformed shape of the piezoelectric actuator under dynamic load may contribute to discrepancies between the model and experiment.

6.4 Future Work

Reasonable predictions of fluid flow rate were achieved using a system dynamic model that coupled lumped element piezoelectric material behavior and fluid dynamics. Additional data on critical system components are anticipated to provide further insight into model parameter estimates. For example, the model under predicts flow rate which may be due to fluid resonant effects in the pump chamber. Transient and harmonic computational fluid dynamic simulations and comparisons to experimental results would be useful in developing lumped parameter estimates of this behavior. Additionally, most compositions of lead zirconate titanate are only approximately linear, particularly at field levels in the MV/m range at multiple drive frequencies and temperature. Thus, hysteresis effects should be considered in the model to obtain a better estimate on flow performance. Furthermore, variable flow resistance across the valves typically plays a critical role in fluid flow although only theoretical approximations of valve fluid dynamics and mechanical coupling have been considered here. Finally back-pressure and back-flow can significantly degrade performance in some regimes, and these effects should be added to the model.

Acknowledgments

We would like to thank the NCSU Department of Mathematics, the Center for Research in Scientific Computation, and the Statistical and Applied Mathematics Studies Institute for organizing the IMSM workshop, and the many individuals at these organizations who gave their time and effort. We also offer our thanks to PAR Technologies for presenting their work to us, and providing us with data with which to validate our model. Finally, we offer our thanks to the National Science Foundation for their support of the workshop.

Bibliography

[1] Lagarias, J. C., Reeds, J.A, Wright, M. H. and Wright, P. E. (1998) Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions, *SIAM Journal of Optimization*, 9:112-147.

[2] Munson, B. R., Young, D. F. and Okiishi, T. H. (1994) *Fundamentals of Fluid Mechanics*, Second edition, John Wiley and Sons, Inc., Iowa.

[3] Oates, W. S. and Lynch, C. S. (2001) 'Piezo electric Hydraulic Pump System Dynamic Model, *Journal of Intelligent Material Systems and Structures*, 12:737-744.

[4] Ogata, K. (1978) *System Dynamics*,' Second edition, Prentice-Hall, Inc., New Jersey.

[5] Timoshenko, S. and Woinowsky-Krieger, S. (1959) *Theory of Plates and Shells*,' McGraw-Hill, New York.

Chapter 7

Modeling and Forecasting of Electric Power Load

PROBLEM 7: MODELING AND FORECASTING OF ELECTRIC POWER LOAD

Jennifer Geis[1], Zheng Li[2], Danny Modlin[3], Min-Hyong Park[4], Jyothsna Prabhakaran[5], Troy Tingey[6], Wei Zhang[7]

Problem Presenters:
Mike Settlage & Glen Snider
Progress Energy

Faculty Consultants:
Dr. Tao Pang & Dr. Denis Pelletier
North Carolina State University

Abstract

Progress Energy is a Fortune 250 company serving the east coast by providing residential and industrial customers with power when they require it. However, developing an accurate forecast model for the amount of power that will be used is an on-going challenge. Factors that impact the power load usage include time of day, the day of the year, and the weather, among many others. Based upon these given factors, can an accurate model be created to predict the power load usage in a state such as Georgia? For the purpose of business operations, this model should be accurate for predicting the power load usage for up to a week into the future. This problem requires a statistical analysis that is thoughtful of knowledge outside of the data which includes the behavior of power consumers and other factors that are often difficult to measure. Using regression analysis with time series data, we aimed to build a model that reflects systematic movements in the data and predict them accurately so that prediction errors would be more or less random. Theoretical conceptualization paired with intuitive experimentation led to the creation of our model.

7.1 Introduction and Motivation

Progress Energy is a Fortune 250 energy company on the east coast serving customers in North Carolina, South Carolina, Georgia, and Florida. Approximately 38% of the power generated by Progress is consumed by industrial users, 37% by residential users, and the remaining is used to generate electricity. Thus accurate forecasting of demand for electricity allows for reliable and cost efficient delivery of energy to their many and varied consumers. However, building a model to minimize forecast errors is a challenging task since there are numerous measurable and non-measurable factors that drive the decisions that form the overall demand for

[1]Augsburg College
[2]Brown University
[3]University of North Carolina at Wilmington
[4]Cornell University
[5]University of Alabama at Birmingham
[6]Arizona State University
[7]Tufts University

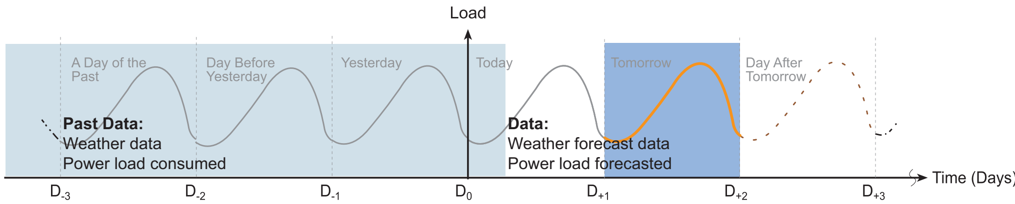


Figure 7.1: Forecast model uses past data for weather and power load until today, weather forecasts, and predicted values for the rest of the day in order to predict how much energy will be used tomorrow.

power consumption.

More specifically, the problem presented to us arises from the power consumption in Georgia. Given historical weather and power consumption data, with their geographic origin within the state, and some general information about habits and trends of populations with respect to this data, we were to build a forecast model that accurately predicts the power demand for the current and also the next day. These predictions were to be made at 6am and assist Progress Energy in their choices in energy production and purchasing.

This is a problem not only for mathematical or statistical study, it is one that may be contemplated by anyone who is personally familiar with power usage. Its relevance to our personal lives was useful in our approach in that intuitive as well as known general conditions and habits of this area and its population along with our personal experiences with power usage played a strong role in our approach to it. However, the difficulty of measuring intuition specific to populations lends to the complexity of this forecasting problem and refuses intuition as a variable for purposes of modeling. Instead, variables that affect the decision to use more electricity (such as weather forecasts) as well as historical data of power usage itself seem more appropriate and warrant careful analysis.

In using regression analysis with time series data, our aim was to build a model that reflects systematic movements in the data and predict them accurately such that prediction errors would be more or less random. Theoretical conceptualization paired with concurrent experimentation with intuitive foundations led to a highly constructive collaboration in the development of our model. It has yielded the following generalized framework for a forecast model, two example models specific to Georgia that led to its development, and recommendations for the continuing development of this model.

7.2 Description of Data

The data given to us included the hourly power load usage from power plants in Jackson County and several other counties covering all parts of Georgia. This was accompanied by hourly weather data including temperature, dew point, humidity, heat index, wind chill, wind direction, wind speed, wet bulb temperature, cloud cover, the number of sunshine minutes, and the probability of precipitation from seven weather stations. This data was collected from January 1, 1998 to July 1, 2005.

Contained within the massive amounts of data provided, there were several instances where the data entries were invalid or missing. There were occasions where power consumption was negative. These data values were deleted from the data set prior to the creation of the model. Data values that were missing within weather

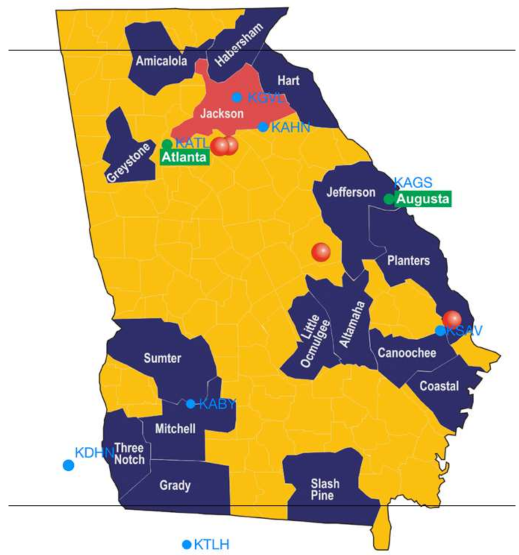


Figure 7.2: Geographic Location of Data

variables were ignored by the statistical software if that variable remained in the model.

7.3 One Model From Two Different and Complementary Modeling Approaches

In the interest of time and group size, the group was partitioned into two in order to create two models, two approaches, and then provide a comparison between the both. However, the differences in ideas and methods of each partition became increasingly complementary throughout the week. This lead to the aforementioned simultaneity of conceptualization and application, as well as a subsequently interwoven progress between the two.

7.4 Group 1: Conceptual Modeling and its Future Direction

7.4.1 Model Structure

With prior knowledge that the three greatest influences in power consumption include time of day, time of year, and weather, we were able to build a founding framework of our model. The entire forecast model would

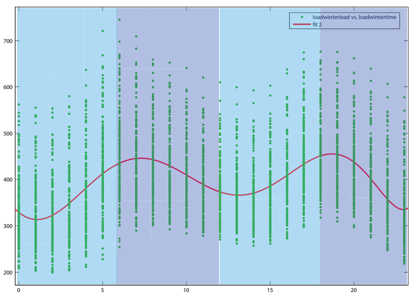


Figure 7.3: A Polynomial fit of Power Usage in Winter on 24 Hour Timeline Partitioned into Four Equal Time Periods

consist of multiple regression models for each season (time of year) and time of day. For example, there might be 96 different regression equations if we were to have a regression for each hour of a day, and four seasons (= 4 seasons $\times$24 hours of a day). That is, given a time of year, there would be a set of 24 equations that model the hourly power usage for a day in that season.

It seemed more sensible to partition a day into time periods of some number of consecutive hours and fit a regression model for each time period of the day accordingly. For instance, if a day was divided into four six-hour long periods, there would be a set of four models modeling the power usage for a day instead of 24 for a total of 16 models instead of 96. Figure 7.3 is an example of a partition that we would use for a day within a time of year which qualifies as winter. Such a division of a day seemed more sensible for several reasons. Initially, the goal of this problem was to create a forecast model that predicts an entire day, not an hour ahead, and 18 hours in advance. Also, cursory qualitative data analysis indicated that such partitioning would not create difficulties in fitting a regression model since it is reasonable to assume that any non-linear trends in power consumption over a period of time could be reflected through relevant variables (such as a time trending variable). For instance, in Figure 7.3, for the regression model in the time period beginning at midnight and ending at 6 am, variable selection may include a quadratic time trending term, while the following period beginning at 6 am and ending at 12 pm may include a cubic time trending term also. The significance of these terms would be determined through regression analysis. Variable selection will be discussed in greater detail in the following section.

What if the time periods were not equal? That is, could there be any advantages to choosing a different number and/or different length time periods within a day? While the previous partitioning seems impartial, unequal partitioning would allow for an additional degree of flexibility. That is, instead of trying to fit a curve (a model) to observations over a time period that is inconsiderate as to how the volume of power usage changes over that time, selecting time periods based on qualitative analysis of these graphs such that the time periods roughly capture sections of the graphical curves that demonstrate a certain trend as seen in Figure 7.4

In the right picture of Figure 7.4, the endpoints of time periods in this graph have been chosen at the minima, maxima, and inflection points. While there does not seem to be a strong quadratic time trend between the hours of 10 am and 4 pm, there does seem to be a noticeable one between the hours of 4 pm and 10 pm. If the summer day load curve had been partitioned into equal six hour segments as in the figure to the left, it appears unlikely that a regression model of the time periods between 6 pm and midnight would have

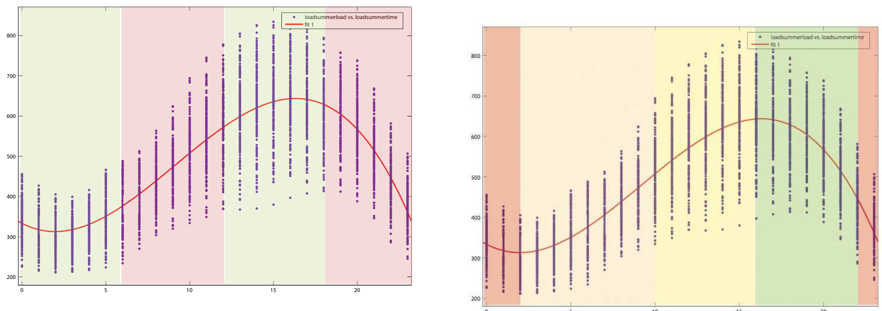


Figure 7.4: Equal Six Hour Time Divisions of a Summer Day versus Unequal Divisions

```
Call: lm(formula = Load11 ~ Temp + TimeA1 + TimeA1^2 + Load11.1)
Residuals:
    Min     1Q Median    3Q   Max
 -117.9 -20.35 -3.049 17.37 242.9

Coefficients:
                  Value Std. Error   t value Pr(>|t|)
 (Intercept) -317.2191    32.5641   -9.7414   0.0000
        Temp    4.8342     0.4106   11.7726   0.0000
      TimeA1  189.6030    15.7254   12.0571   0.0000
 I(TimeA1^2)  -45.5151     3.8498  -11.8226   0.0000
    Load11.1    0.4830     0.0265   18.2342   0.0000
```

Figure 7.5: Output from S-Plus, Time Trending

captured the quadratic trend. Moreover, the previous time periods between noon and 6 pm may have some linear time trending, but it appears doubtful that there would be a quadratic trend. While time restrictions of the workshop did now allow for thorough testing of these hypotheses, we were able to complete a regression analysis of the time period of a summer day between the hours of 4 pm and 10 pm. Indeed, as the output in Figure 7.5 suggests, there is a significant quadratic time trend.

With this example of the latter method of unequal time division, we were able to capture a significant behavior of the data that the former is likely to have missed. It is more desirable with respect to our goal, to build an accurate model. In spite of this benefit with respect to accuracy in modeling, it raised a new challenge. How could the data be organized in an efficient manner? If there are about 24 regression models within our forecast model, must the data also be partitioned into 24 respective data sets? Is there a less time consuming or more cost effective approach? These questions are considered in further detail in Section 7.3.

Another question that arose was in the selection of endpoints of seasons in addition to time periods within a day. For instance, general and intuitive knowledge suggests that in Georgia spring and fall are much shorter seasons than summer and winter. An accurate model should reflect this more carefully. This would require more data analysis and will be discussed in Section 7.5.

Mindful of these concerns, a generalized framework of our forecast model was built as follows. Given hourly data, for some season i, and a time period j of a day in season i, the ijth regression model would have the form:

$$Load_{i[j],t_j[k]} = \alpha_{i[j]} + \beta_{i[j]} Weather_{i[j],t_j[k]} + \tau_{i[j]} t + \phi_{i[j]}(L) Lags_{i[j],t_j[k]} + \epsilon_{i[j],t_j[k]} \tag{7.1}$$

$$i \in [1, s] \in \mathbb{N} \text{ where } s \text{ is the number of ``seasons'' a year has been divided into}$$

$$j \in [1, p_i] \in \mathbb{N} \text{ where } p_i \text{ is the total number of time periods of a day in season } i$$

$$t_j \in \left[1, h_{i[j]}\right] \in \mathbb{N} \text{ where } h_{i[j]} = \text{total number of hours in periods } i[j]$$

$$k \in [1, d] \in \mathbb{N} \text{ where } d = \frac{\text{total number of observed data points in season } i}{24}$$

Then,

d is the number of days in the season,

$i[j]t_j[k]$th data point is one specific observation of any variable,

$Load_{i[j],t_j[k]}$ is $i[j]t_j[k]$th observation of power load,

$\alpha_{i[j]}$ is the intercept parameter for the ijth model,

$Weather_{i[j],t_j[k]}$ is a w by 1 column vector containing weather variables where w is to be determined in variables selection(section 7.3.2),

$\beta_{j[j]}$ is a 1 by w row vector containing coefficients corresponding to relevant variables in $Weather$ vector, set to zero if regression analysis indicates that the relevant weather variable is not significant at a significance level of 5%

$$t = \begin{bmatrix} t_j \\ t_j^2 \\ t_j^3 \end{bmatrix},$$

$\tau_{i[j]}$ is a 1 by 3 row vector containing coefficient corresponding to relevant variables in t vector,

$$Lags_{i[j],t_j[k]} = \begin{bmatrix} Load_{i[j],t_j[k]} \\ Load_{i[j],t_j[k]} \\ \cdots \\ Load_{i[j],t_j[k]} \end{bmatrix} \text{ a } q \text{ by 1 column vector,}$$

$\phi_{i[j]}(L) = \begin{bmatrix} L \; L^2 \; \cdots \; L^q \end{bmatrix}$ a 1 by q row vector,

L is the lag operator where $L^q X_{i[j],t_j[k]} = X_{i[j-q],t_j[k]}$, and

$\epsilon_{i[j],t_j[k]}$ is the error associated with the $t_j[k]$th observation.

Thus the fitted model:

$$Load_{i[j],t_j[k]} = \hat{\alpha}_{i[j]} + \hat{\beta}_{i[j]} Weather_{i[j],t_j[k]} + \hat{\tau}_{i[j]} t + \phi_{i[j]}(L) Lags_{i[j],t_j[k]} \tag{7.2}$$

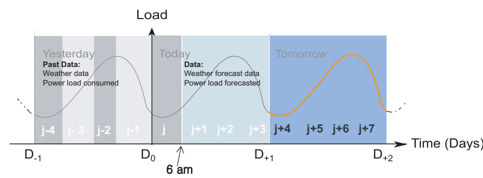


Figure 7.6: A Zoom-In Sketch of the Forecast Model

How this fitted model would be used for the purpose of prediction is best explained using an example. Consider the case where each day is divided equally into six hour periods (as in Figure 7.3), with four seasons in a year. In this case $s = 4$, $p1 = p2 = p3 = p4 = 4$, and $t1$, $t2$, $t3$, $t4$ are all a sequence of natural numbers from 1 to 6. Let season 1, 2, 3, and 4 correspond respectively to summer, fall, winter, and spring. Also, let time period 1 be the period beginning at midnight, end at 6 am, with periods 2, 3, 4 following sequentially. Then at 6 am today, in order to predict the total power load for tomorrow, the following set of seven equations are computed recursively with weather forecast data and then the four that represent forecasts of tomorrow's power load are summed. That is, the one day prediction is equal to:

$$\sum_{r=4}^{7} \widehat{Load}_{i[j+r],t_j[k]}. \tag{7.3}$$

It should be noted that there was an underlying assumption that the data was being divided into the appropriate number of models. However, the possibility of having one large equation with dummy variables that 'select' the appropriate model while keeping the data set was an idea that was considered in greater detail later on.

7.4.2 Variable Selection

The problem aims for a model with forecast accuracy. Thus, we select variables for our model based upon statistical soundness of a model. This is a sufficient condition for its forecast accuracy whereas the converse is not necessarily true.

To minimize issues of multicollinearity among explanatory variables in our model, we aim to select as few variables as possible and then to systematically examine their interactions instead of creating new variables. It is unreasonable to worsen this issue by creating derived variables (e.g. as functions of factors such as temperature, humidity, or wind speed), particularly when the dependent variables of these derived ones is suspected to have strong effect on the power load.

For example, a model that includes temperature and wind speed as explanatory variables would implicitly reflect the wind chill factor. It is a known second order function of temperature and wind speed (mph) given as:

$$WindChill = 35.74 + 0.6215 * Temperature - 35.75 * WindSpeed^{.16} + 0.4275 * Temperature * WindSpeed^{.16}$$

(http://www.srh.weather.gov/elp/wxcalc/formulas/)

Since we suspect temperature to have its own effect on the demand of power load and it also has a known contribution to the determination of the wind chill factor, we can eliminate this factor. This decision also has a secondary benefit of reducing the number of variables (and interactions to consider), thus yielding a simpler model in addition to a more accurate model from the perspective of statistical theory.

There are still more variables that may be omitted from the model for a different reason. Since the forecast model requires usage of forecasted weather data, a model should be fitted to observed weather variables that are reasonably forecasted as well. For instance, cloud coverage in minutes per hour and minutes of sunshine per hour are difficult to forecast. Furthermore, if temperature is related and dependent on these two variables, the given data is not adequate to estimate any dependable relation. The reason for this is clarified when considering the cloud coverage data, which does not reveal how much of cloud coverage there was either. That is, the effect that cloud coverage and sunshine minutes may have on temperature is already implicit in the temperature data. Even if a model could be fitted to cloud coverage minutes per hour, there is a strong likelihood of a negative correlation of sunshine minutes to cloud coverage. Since sunshine minutes tend to occur only during day time, it is not good data, even to fit a model to past data. With these considerations in mind, along with the consideration of the effect of time and our treatment of bad data, discussed above in section two, we were ready to proceed "with model fitting".'

We began regression analysis with an over-parameterized model and systematically eliminated insignificant terms beginning from higher order terms. If some interaction or parameter was considered significant, its lower order components are significant, but their p-values would no longer be valid. This purposefully excessive model included third order interactions between $Temperature$, $Humidity$, $WindSpeed$, and $WindDirection$ in hopes of capturing other interactions that we may not have considered or are counter intuitive. Furthermore, using the 'formula' for heat index from http://www.weather.gov, we included additional polynomial interactions between $Temperature$ and $Humidity$. This was also useful since another variable in our data that was omitted was $Wet-BulbTemperature$, which is closely related to $Temperature$ and $Humidity$. Also, linear, quadratic, and time variables were included. Six daily lags of power load consumption were included as well.

The consequences of this approach were highly constructive for both the design group as well as the empirical group. More specifically, the empirical results of the above trial done in the design group was promising. The variables that remained corresponded to the more intuitively grasped models of the empirical group. From this, the empirical group was able to adapt this stepwise method of fitting a model, while the design group focused on diagnostic analysis of the design of the model.

7.4.3 Improvements and Additions to Model Structure

In the model described in Section 7.4.1, the lag operator was not correctly defined; it takes the lag of an observation of power load from the previous day over the same time period. These are not interesting lags. It seemed more sensible to have lags of the power load usage in period increments periods instead of p_i period increments. Since our general model structure at this point was allowing for the inequal lengths of time periods within a day, for purposes of empirical experimentation, power load observations of previous power loads were averaged and then used as period lags instead. This application is explained in greater detail toward the end of Section 7.5. Still, from a conceptual perspective, it seemed more sensible to have hourly lags and correct the lag operator as the following:

$$L^p X_{i[j],t_j[k]} = X_{i[j],(t_j-p)[k]} \tag{7.4}$$

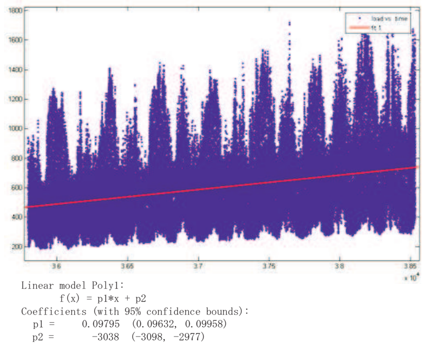


Figure 7.7: Equal Six Hour Time Divisions of a Summer Day versus Unequal Divisions

Still, the fitted model above indicates that the power consumption of the same period of time one day ago is significant for today's power load consumption. This was the insight for our decision to consider the significance of a long term growth parameter with an overall time trending variable. Indeed, some data analysis indicated a significant linear growth over the period of seven years from which the data is taken.

With the data partitioned into the sum of the number of periods in every season, we wondered how this factor could be implemented into our model. Moreover, this observation motivated the question of whether or not there were cross-seasonal or cross-periodical effects that were being overlooked by the model described in Section 7.4.1. Namely, are there significant factors for the power load usage for any time period of a day given a season, or are they effects that were true for any season given a time period? For example, could there be some factors that have explanatory power for power load usage between the hours of 10 am and 4 pm, any day of the year?

The model that could incorporate such effects, if they existed, was found in a different framework for the same model that uses crossed indicators by season and period. Instead of having a model that contains numerous sets of regression equations, all of the sub-models of our forecast model could be aggregated into one equation. The model was re-written as follows:

$$Y_{i[j],t_{j[k]}} = Load_{i[j],t_{j[k]}}$$

Also, define dummy variables:

$$s_i = \begin{cases} 1 \text{ in seasons } i \\ 0 \text{ otherwise} \end{cases} \qquad p_j = \begin{cases} 1 \text{ in time period } j \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{aligned} Y_{t_j[k]} = \\ & s_1 p_1 Y_{1[1],t_j[k]} + s_1 p_2 Y_{1[2],t_j[k]} + ... + s_i p_{j_i} Y_{i[j],t_j[k]} + ... + s_s p_{p_s} Y_{s[p-1],t_j[k]} + s_s p_{p_s} Y_{s[p],t_j[k]} + \\ & s_1 \delta_1 PeriodFactors.1_{1[j],t_j[k]} + ... + s_i \delta_i PeriodFactors.i_{i[j],t_j[k]} + ... + s_s \delta_s PeriodFactors.s_{s[j],t_j[k]} + \\ & p_1 \gamma_1 SeasonFactors.1_{1[1],t_j[k]} + ... + p_j \gamma_j SeasonFactors.j_{i[j],t_j[k]} + ... + p_p \gamma_p SeasonFactors.p_{i[p],t_j[k]} \\ & + \zeta CrossDataFactors_{i[j],t_j[k]} + \epsilon_{i[j],t_j[k]} \end{aligned} \tag{7.5}$$

δ, γ, ζ are row vectors of coefficient parameters pertaining to the variable column vectors $PeriodFactors$, $SeasonFactors$, and $CrossDataFactors$. An example of a variable in $CrossDataFactors$ is a long-term time trend variable which would be a sequence of natural numbers from 1 to the total number of data points.

Define g as the long-term time trend. Then,

$$CrossDataFactors_{i[j],t_j[k]} = \begin{bmatrix} \vdots \\ g \\ \vdots \end{bmatrix} \text{ where } g \in [1, \textit{total number of data points}] .$$

While such cross-temporal comparisons in our previous model would require cross-equation constraints on coefficients of any variable that would take such effects into account, this format does not and is simpler in this respect. In addition to this major advantage, there are other benefits. First, data can be kept intact instead of being portioned into numerous data sets. Also, changed endpoints of seasons and time periods would not require laborious re-partitioning of the data. Since each dummy variable takes on a value of 0 or 1, the endpoint changes can be made here instead of in the physical data. This is more efficient.

7.5 Group 2: Empirical Analysis and Results

7.5.1 Model Creation and Testing Procedures

Our second group took another, slightly different approach but used many similar concepts as the first group in order to create a model. As discussed before, power usage is widely known to be based upon basic human trends and habits. This line of thinking allowed us to eliminate certain weather variables immediately due to the fact that they are derived variables. Hence, dew point temperature was eliminated because it is a function of saturated vapor pressure and humidity; heat index was eliminated because it is a function of temperature and humidity; wind chill factor was eliminated because it is a function of temperature and humidity; and wet bulb temperature was eliminated because it was replication of humidity.

Furthermore, the group decided to immediately add time series variables (lags) that take into account the power load consumption for hours prior to the current hour of prediction. The group believed that having a prior knowledge of the power consumption at times up to six hours in advance could aid in our prediction.

Based upon the examination of the interactions, certain weather and lag power load data factors were either kept or eliminated. Our p-value quality indicator was used for each variable to determine elimination of a certain variable. Hence, if the p-value was larger than $\alpha = .05$, we chose to eliminate that variable from our model.

We also took an insight from the Design group to partition the data according to seasons and time periods per day. Since we have seven stations worth of weather data, we merged the data into two sets of averages: one set for the three stations closest to Jackson County and one set for the remaining five weather stations. We then used each set of data, one for Jackson and one for the rest of Georgia, to create separate models.

For the two sets of data, we systematically checked for any relationship by examining graphs of all weather variable interactions for similar line trends. The data was then regressed in order to create a tentative model. For each weather variable, we checked its p-value for significance with $\alpha = .05$ and eliminated variables should the p-value exceed .05. As certain weather variables were taken out of the tentative model, our adjusted r-square was either maintained or increased. After checking all of the weather variables, this produced an adjusted r-square of 98%. However, even though this was desirable, our mean absolute percent error (MAPE) was still around 5%, much larger than our desired 1% or less.

In order to further improve our model and hence our MAPE, we decided to divide our data another time. This was done by using the first four years worth of data in order to create tentative models and using the remaining three years to test our models. In this way, we were able to examine more closely where improvement needed to be made within the model. We then created a regression model, this time without an intercept. With the differences in the power load for each season, forcing the model to a specific intercept did not seem best.

These changes produced a model that was better able to react to a specific season. After creating a model for the GEC data, we were pleased to see an adjusted r-square value of 0.9986 and thus are able to account for 99.86% of the variation of the power load within our model. Even more gratifying was to see that we had also reduced our MAPE to 3.2% on an hourly basis. Our second model for the Jackson county data produced expected better results with an adjusted r-square of 0.9991 and a reduced MAPE of 2.5%.

Since a large portion of the complexity of this problem lies with the ability to forecast up to one week into the future, we designed a program that would simulate extended forecasting with recursive predictions. However, simulation results for forecasting 42 hours ahead in time did not produce desirable results.

In light of these results, we decided to return to the ideas at the drawing board and tackle this data in a new way. We thought that if we could get a grasp on how to model the GEC data, we could use a similar approach for the Jackson data. We again kept the averaged values for the five weather stations of the greater Georgia area.

Since we knew that both weather and lag load information were important factors, we decided to look at both but in separate models. A first model was created solely based on weather. It only used the main effects and no interactions were considered. The model resulted in explaining approximately 47% of the variation of power. The second model included only the most recent power load amounts. Intuitively, it seemed only these had any influence on future power load usage. We looked into hourly lags of one to six hours, 12 hours, and 18 hours and also daily lags from one to three days. Using just this information in the model, the variation described increased to over 70%. This gave us good feedback in understanding how each needs to be used in a combined model.

Next, we took both weather and lag time into consideration in a single model. Before creating this model, we decided that several weather variables would not be useful in an extended forecast. These variables were sunshine minutes and cloud cover. Simply thinking about a traditional weather forecast, we do not typically hear about these items and feel that including them in the model would not allow for a truly accurate prediction. Probability of precipitation was also dropped from the model due to the fact that there were excessive amounts of data values missing. Including the probability of precipitation in the model would also not allow for an accurate response. This new combined model gave an improved variation of over 90%.

However, we knew this could be taken a step further. We knew that there were several variables that interact with each other to cause a high demand in power. For example, temperature and humidity in the summer creates the heat index. When the heat index is high, air conditioners run more frequently. Likewise in the winter, temperature and wind speed create the wind chill, and heating systems are working harder when the wind chill is colder.

For this reason, we added in interactions terms such as $temperature*humidity$ and $temperature*windchill$ and other similar ones that seemed to make sense. We recalled that several of the variables showed a quadratic-like picture. Hence, we also incorporated squares of each variable into the model. We then re-analyzed by regression and checked p-values of both the variables and the squared variables. At each step of the procedure, we removed any variable whose p-value was above 0.05.

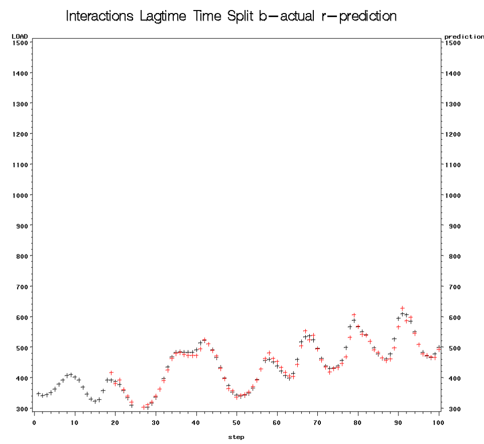


Figure 7.8: Combination of Weather Variables and Lagged Power Loads

Now we knew that some of the weather variables were only factors during certain seasons. For example, the heat index is only important in the summer and wind chill only in the winter. We also knew some of the variables were bigger factors during certain parts of the day. For example,the heat index at night is not as much of a factor as in the day. To take this into account, we introduced binary variables in our model to indicate which season each data set was in and what time of day it was. We were pleased to see our adjusted r-square value increased to over 99%.

In order to test this new model, we once again split the data into the first four years from which the model was generated and used the last three years to test it. This was so that we would not be in-sampling which could give false hope to the true power of the test. The hourly basis MAPE was around 5%. Unfortunately this did not meet our goal to be below 4%.

A re-assessment of our analysis made us realize that up to now, we assumed that the weather data was normally distributed. As a check, we normalized the variables that were left in the model and ran the regression again. The model's adjusted r-square value did not improve. However, now the one hour MAPE was 2.312%. This was a positive indicator. But we needed to be able to forecast for extended period of times. We created a program to forecast up to 42 hours ahead. This resulted in a MAPE of 12.81%. Our model was fine for predicting the immediate future but not for the extended forecast.

We believed that the problem was that the model took into account the previous hours power load. Hence, with each forecast past two hours ahead, the model was using previously forecasted values instead of an actual amount of power. This repeated use of previously predicted values was compounding the error. We needed a model that corrected this dependency. We began trying and testing many other combinations of lag times. Each time, the MAPE on a hourly basis worsened, which we expected, but the 42 hour MAPE did not improve to an acceptable level.

Considering the lack of improvement in the MAPE, we wondered if our splitting the year into two seasons and the day into two parts was the cause of the problem. We thought that maybe four seasons should be used and that the day be split into four groups dictated by either a linear, quadratic, or cubic relationship as seen in the graphs of power versus time.

We then focused our attention to only the months of July and August. These two months, in Georgia, are

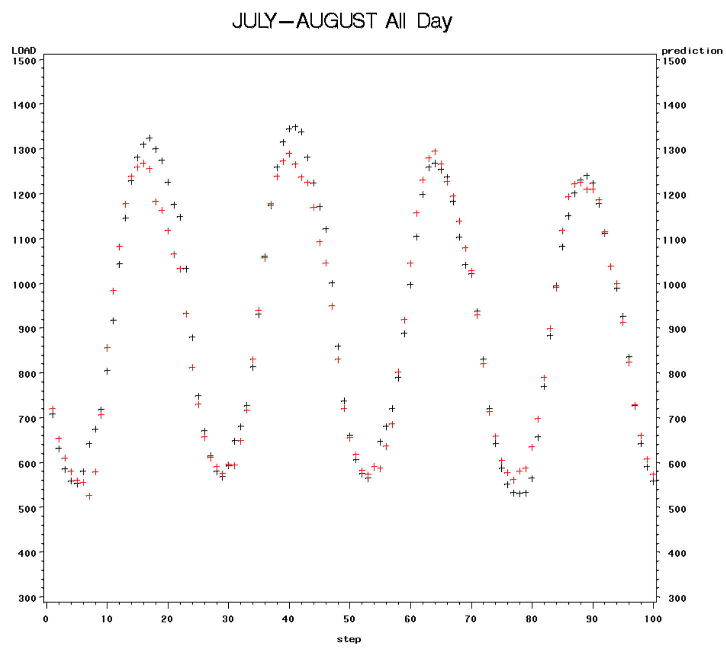


Figure 7.9: Predictions (Red) of Entire Day

definitely summer months with very consistent weather patterns. We included new time periods of six equal time periods. To improve the problem of the previous hour prediction being used, we averaged the lag of the previous six hours block. This improved our 42 hour MAPE to 8%. When the predictions were modeled against the true values, the majority of our biggest misses were at the peaks in the day. We then adjusted the time blocks again.

7.5.2 Final Empirical Model and Conclusions

This final model as given in Table 7.1 gives us an adjusted r-squared of 0.9518 which is fairly good. For forecasting an hour ahead, two hours ahead and 42 hours ahead, the MAPEs are 5%, 5% and 8% respectively.

7.6 Conclusions and Further Development of the Model

The underlying model structure of Group 2's initial models were derived directly from data, regression analysis, and experimentation. It was completely compatible with the separately conceptualized one of Group 1 Section 7.3.1 and was highly advantageous to successful development of a singular model. Notable improvements in forecast accuracy of the empirical model (as discuess in Section 7.4 coincided with application of conceptual

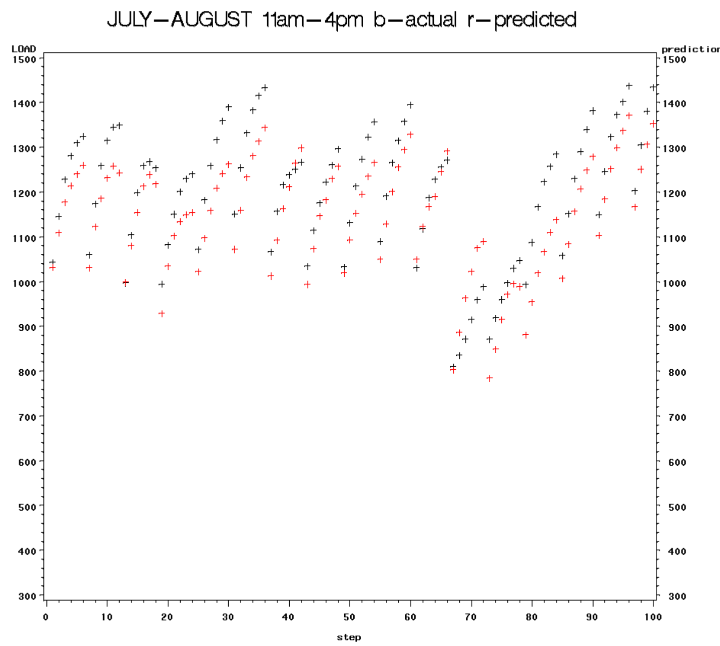


Figure 7.10: Predictions (Red) of 11am-4pm only

Table 7.1: Variables Kept For Final Model

Variable	DF	Parameter Est	se	t-value	Pr > $\|t\|$
intercept	1	−769.5810	543.23158	−8.78	<.0001
time	1	−0.8650	0.41945	−2.06	0.0392
temperature	1	93.8647	10.06274	9.33	<.0001
humidity	1	22.5036	4.19392	5.37	<.0001
wind speed	1	−4.7598	0.36033	−13.21	<.0001
temperature	1	−0.4071	0.04516	−9.02	<.0001
humidity^2	1	0.4770	0.05806	8.22	<.0001
temperature^2∗humidity	1	−0.0031	0.00062	−5.07	<.0001
temperature∗humidity^2	1	−0.0160	0.00201	−7.96	<.0001
temperature^2∗humidity^2	1	0.0001	0.00002	7.26	<.0001
load back 1 day	1	0.6627	0.01153	57.48	<.0001
load back 2 days	1	0.0643	0.01051	6.11	<.0001
load back avg of 6 hrs	1	0.0779	0.00354	21.99	<.0001

ideas (as discussed in Section 7.3. Through trial and error, the second group initially created a model that fit the data quite well. In spite of this, simulation forecasting with this fitted model indicated that it was unsatisfactory as a forecast model. With the application of conceptualized ideas, this empirically based model continued to improve. Similarly, the developments in the design of the conceptual model were inspired through the developments in the former.

For future considerations of this model to increase its accuracy and precision, it seems feasible to have an hourly self-updating forecast model given streaming data. The fitted coefficients of the regression model would be variable instead of constant. This is a reasonable goal for the model since the method of fitting a model described in Section 7.4.2 is systematic and eliminates variables based on a numerical condition. Also, there are weather variables and non-weather variables that may be interesting to examine as factors influencing power consumption, such as amount of precipitation or snow, economic data such as GDP, or oil or gas prices. Also, recall the improved format of our model that uses dummy variables. A major advantage of such a model is that additional models that reflect power consumption during holidays, unusual weather conditions, and other major events could be easily incorporated. These ideas may also provide some guidance as to how leap years and daylight savings time could become a part of our model.

Bibliography

[1] Draper, N R. and Smith, H. (1998) *Applied Regression Analysis,* Wiley, New York.

[2] Gross, J. (2003) *Linear Regression,* Springer, New York.