Optimization Theory Final Project
Temperature Control

Andrew Ptaszek, Duane Franz, Noah Kintner

April 26^{th} 2024

Abstract

In this project we created a linear programming model and an integer programming model to represent the conditions imposed on culture incubators. Such conditions included constant, periodic, and suddenly changing ambient temperatures as well as situations that call some changes in cost vectors. The main objective of the models were to optimize the schedule for activating and operating the heating element. Using the MATLAB functions `linprog` and `intlinprog` we found some optimal schedules for voltage use in the incubator.

Contents

1 Introduction

In the world of medicine, the study of microbial beings–germs–is currently one of the most influential fields of study in medicine. To this end, one of the most important tools to their study is the incubator–our means of mass producing subjects for study. These microbes or cultures have certain temperatures at which they reproduce at their fastest. It is the job of the incubator to maintain this rich environment to maximize the growth of previously mentioned cultures by utilizing their heating element in the most optimal way calculated. In this project, we were tasked with developing a model and solving for the most optimal schedule for the heating element's use of voltage ($u_n : n \in [1, N] \subseteq \mathbb{N}$) over the course of a 24-hour period of time such that t represents time in hours.

The primary objective of our proposed incubator is to keep the internal temperature ($x_n : n \in [1, N] \subseteq \mathbb{N}$) between $35°C$ and $39°C$. In regards to the heating element, there are two circumstances we were tasked to model—the heating element is able to have varied voltages ($u_n \in [0, \infty) \subseteq \mathbb{R}$) and the heating element is able to be turned off or on ($u_n \in \{0, 1\}$). Relating this into the objective, optimizing the solution in each scenario includes minimizing the overall consumption of voltage by the heating element. There is also a scenario in which we wish to consider minimizing the number of times the heating element is switched on and off. We also have to consider the fact that incubators do not have perfect insulation, so the ambient temperature must be taken into consideration for our model. There are three scenarios regarding the ambient temperature—constant ambient temperature ($A(t) = 21°C$), periodic ambient temperature ($A(t) = \left(21 + 4\sin\left(\frac{\pi(t-8)}{12}\right)\right)^{\circ} C$), and an ambient temperature that stays one constant temperature for the first 12 hours ($A(t) = 21°C$) then suddenly another constant temperature for the next 12 hours ($A(t) = 17°C$).

There is an entire industry for making incubators as well as other tools for science. The companies that make incubators for scientific studies compete amongst each other for making the better product, so that they can make a profit from their consumers. As a result, specific data regarding cooling coefficients as well as the models and methods they use for making their products accurately are often not released to the public. However, the concept of an incubator is well discussed in many college courses such as differential equations. Thus, the models we will use to discuss the nature of the internal temperature of the incubator (x_n) will be modified versions of Newton's Law of Cooling.

Our models each have constraints that enforce the variables representing the internal temperature of the incubator (x_n) follow the natural laws of thermodynamics. To this end we tweaked Newton's law of cooling to include a heat source to determine the temperature at the next time step. For the purpose of making a more realistic model, one team member sat in front of an oven for an hour to calculate the insulation constant of the oven to use in our model. We also made a variable that represented the difference between the the internal temperature of the incubator from the temperature $37°C$ for a simpler means of applying weights to represent the importance of maintaining the desired temperatures. With this kept in mind, there are two model types we used in this project—linear programs and integer programs. To solve these programs, we utilized relaxation methods, the function `linprog` for our linear programs, and `intlinprog` for our integer programs.

For our linear programs–which represent the voltage of the heating element (u_n) being a positive real number–, the optimized voltage schedules (u_n) allowed the internal temperature of the incubator (x_n) to reach $37°C$ quickly and stay there for a long time for each of the ambient temperature scenarios. The integer programs–which represent a heating element only being able to be turned on and off– had internal temperatures that fluctuated a lot even in constant ambient temperature.

2 Model Formulation

2.1 Initial Model

Let Δt be the time step in hours. Let u_n be the voltage used in the heating element in the linear program at the n^{th} time step. Let u_n be the decision variable for the usage of the heating element in the integer program at the n^{th} time step where $u_n = 1$ means that the heating element is on and $u_n = 0$ means that the heating element is off. Let x_n, A_n be the internal temperature of the incubator and the ambient temperature at the n^{th} time step, respectively. Let k be the thermal resistance constant. Let c be the voltage to heat correlation constant of the heating element in the linear program and the temperature of the heating element when turned on in the integer program. Let d_n be the difference between the desired temperature, $37°C$, and the internal temperature of the incubator (x_n). Let w_1, w_2 be the cost associated with d_n and u_n respectively. With all of these variables kept in mind, this is our initial model:

$$\begin{array}{lccc} \min & \sum_{n=1}^{N} w_1 d_n + w_2 u_n & & \\ s.t. & x_{n+1} + (k\Delta t - 1)x_n - c\Delta t u_n & = & \Delta t k A_n \\ & x_0 & = & A_0 \\ & -d_n + x_n & \leq & 37 \\ & d_n - x_n & \leq & -37 \end{array}$$

The first constraint is a modified form of Newton's Law of Cooling. As discussed in the prompt for this project, heat energy exchange can be modeled with the modified version of Newton's Law of Cooling mentioned in the online textbook

$$\frac{dx}{dt} = k\left[kA(t) - x(t)\right] + cu(t)$$

where $A(t)$ is the ambient temperature at time t, $x(t)$ is the internal temperature of the incubator at time t, $u(t)$ is the voltage ran through the heating element at time t, k is the constant of heat resistance, and c is the constant of voltage and heat correlation[FRY21] (note that the difference between this model from the prompt and the one from the online book is the term representing the heating element). We discretize this using Euler's method where $\frac{dx}{dt} \approx \frac{x_{n+1}-x_n}{\Delta t}$ to get this version of the model [SH16]

$$\frac{x_{n+1} - x_n}{\Delta t} = k\left[A_n - x_n\right] + cu_n.$$

With some basic algebra we get our constraint

$$x_{n+1} + (\Delta t - 1)x_n - c\Delta t u_n = \Delta t k A_n.$$

The second constraint merely implies that the incubator starts as the same temperature and the ambient temperature. The third and last constraints are meant to enforce an equality constraint

$$d_n = 37 - x_n$$

by separating it into the inequality constraints we give in the model (note that the same can be done with any of the other equality constraints in the model as needed as discussed in the textbook regarding linear optimization[BT97a]). The reason for introducing the variable d_n in the first place was to allow x_n to be outside of the desired temperature range. We wanted this ability for the model because of the fact that incubators almost never start off in the desired temperature. If we alternatively made x_n be bounded between $35°C$ and $39°C$, our solutions would always be infeasible.

2.2 Linear Programming Model

We shall use the variable naming conventions discussed in the development of our initial model for the linear program below

$$\begin{array}{lccc} \min & \sum_{n=1}^{N} w_1 d_n + w_2 u_n & & \\ s.t. & x_{n+1} + (k\Delta t - 1)x_n - c\Delta t u_n & = & \Delta t k A_n \\ & x_1 & = & A_1 \\ & -d_n + x_n & \leq & 37 \\ & d_n - x_n & \leq & -37 \\ & x_n, d_n, A_n \in \mathbb{R}, \quad u_n \geq 0 & & \end{array}$$

In addition to the conditions and assumptions made for the initial model, we assume that the variables regarding voltage and heat are all real numbers. Considering the temperature unit Celsius, we cannot merely assume that the variables are positive as the domains of both dimensions have negative values as well as nonnegative ones. In the case of the voltage schedule, we shall assume that the voltages calculated are positive since applying a negative voltage on the heating element does not cause it to cool down its surroundings.

Notice that there are but two independent variables in this model—the voltage schedule (u_n) and the ambient temperature (A_n). Of these two variables, it is the voltage schedule that we have control over to optimize the voltage consumption and temperature regulation of the incubator. This is what we assumed to be true for the incubator, and thus, this model achieves its purpose.

2.3 Integer Programming Model

We shall use the variable naming conventions discussed in the development of our initial model for the integer program. In addition to those variables, let v_n^+ represent the act of turning on the heating element when equal to 1 and the act of not doing so when 0. Similarly, let v_n^- represent the act of turning the heating element off when equal to 1 and the act of not turning the heating element off when 0. We also added this variable into the objective with a cost of w_3 for the cases where we care how often the heating element is switched on and off. If we are not considering that, we can simply have a cost $w_3 = 0$.

$$\begin{array}{lccc} \min & \sum_{n=1}^{N} w_1 d_n + w_2 u_n + w_3 \left(v_n^+ + v_n^-\right) & & \\ \text{s.t.} & x_{n+1} + (k\Delta t - 1)x_n - c\Delta t u_n & = & \Delta t k A_n \\ & u_n - u_{n+1} + v_n^+ - v_n^- & = & 0 \\ & v_n^+ + v_n^- & \leq & 1 \\ & x_1 & = & A_1 \\ & -d_n + x_n & \leq & 37 \\ & d_n - x_n & \leq & -37 \\ & u_n, v_n^+, v_n^- \in \{0,1\}; \quad x_n, d_n \geq 0 & & \end{array}$$

As can be seen in the model, there are two new constraints. The second equality constraint in this integer program ensures that the we quantify the act of changing the switch from off to an and vice versa. For example, if $u_n = 0$ and $u_{n+1} = 1$ for any value of $n \in [1, N]$ we say that the heating element was switched on during the n^{th} time step. If you plug this into the equation, given that we are using boolean algebra, then we are enforcing $v_n^+ = 1$ and $v_n^- = 0$ which is what we want. As mentioned in the discussion of the initial model, we know how to turn equality constraints into inequality constraints and vice versa if we need them to be[BT97a]. The new inequality constraint near the top of the order ensures that we do not

have both values equal to 1 at the same time as you cannot both turn on a heating element and turn it off at the same instance.

Notice that, just as in the linear programming problem, the two independent variables of the problem are the ambient temperature (A_n) and the voltage schedule (u_n). Also, similar to the linear program, u_n is the variable that we have control over to optimize the temperature, voltage consumption, and number of times the heating element is switched on and off. Thus, the incubator follows the real-world behaviors we wished to emulate, and thus any solution to the program should yield the best voltage schedule.

3 Data

As can be seen from the linear programming and the integer programming models, the only external data we are concerned with are the ambient temperature and the constants that govern the dynamics of heat exchange in both models. With regard to the ambient temperature, we were tasked with three main scenarios to model and find optimal solutions for (remember that A_n is the ambient temperature in degrees Celsius). The first is a constant ambient temperature of $21°C$ or symbolically

$$A_n = 21, \quad \forall n \in [1, N] \subseteq \mathbb{N}.$$

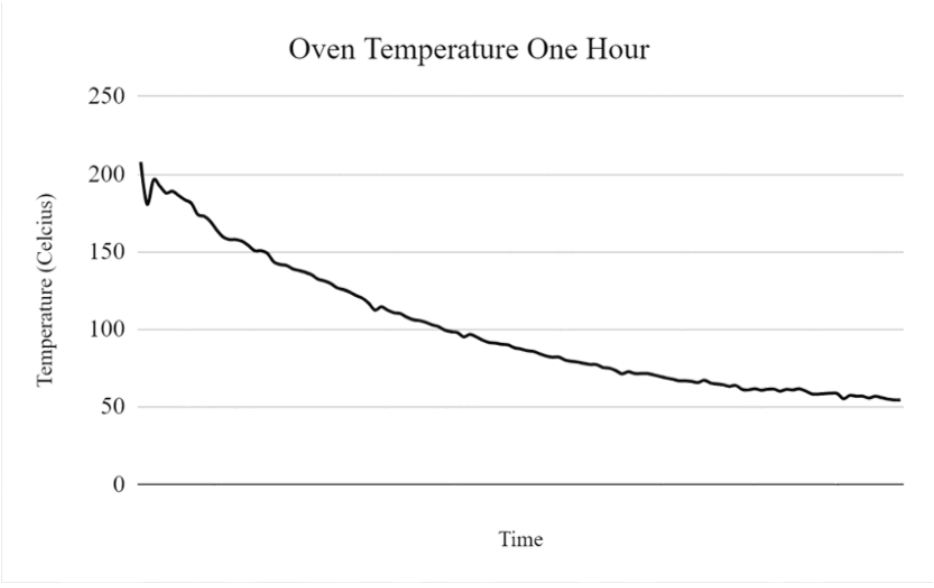
In real life, we can think of this as the inside of a building at night. The second is a periodic ambient temperature which can be characterized by the equation

$$A_n = 21 + 4\sin\left(\frac{\pi(\Delta t(n-1) - 8)}{12}\right), \quad \forall n \in [1, N] \subseteq \mathbb{N}.$$

The last scenario that each model needs to solve for is an ambient temperature that suddenly decreases due to the air conditioning activating. For this scenario, we were given the following function

$$A_n = \begin{cases} 21, \text{ if } \Delta t(n-1) \leq 12 \\ 17, \text{ if } \Delta t(n-1) > 12 \end{cases}, \quad \forall n \in [1, N] \subseteq \mathbb{N}.$$

As mentioned in the introduction, we had used real-world data to calculate our values of k and c. One of our team mates sat in front of an oven for around an hour to collect its data. The following graph is the fruit of his labor.



As can be seen from the graph, the data does not follow a linear path, but it is nearly linear. From this data he utilized the function

$$k_n = \frac{x_{n+1} - x_n}{\Delta t[A_n - x_n]}$$

which can be derived from the modified version of Newton's Law of Cooling mentioned in the referenced textbook[FRY21] (not the one with a heat source we developed in the model) with Euler's method[SH16] applied to it. He then took these values into a separate matrix for statistical analysis that we have displayed in the table below.

Experiment Statistics		
Mean	Standard Deviation	Range
0.68803	1.24600	11.81446

From this analysis we decided to use the value $k = 0.68803$. Note that k is the heat resistance of the incubator, or in this case the oven.

In theory, calculating values for proportion of voltage and heat constant, c, is more difficult of a process to do accurately via experimentation. However, it was observed that it takes the oven approximately 15 minutes to heat the oven from room temperature to $350°F$ and the oven typically operates on 240 V. Using the experimental data on our calculated k value, we approximated our c value as $c \approx 2.6$. We feel that it is safe to use this value for our current use of the the model since changing the maximum voltage of the heating element does not effect the structure of our model in any way.

4 Solution Method

4.1 Linear Program

Looking at the model we developed earlier

$$\begin{array}{lrcl} \min & \sum_{n=1}^{N} w_1 d_n + w_2 u_n & & \\ s.t. & x_{n+1} + (k\Delta t - 1)x_n - c\Delta t u_n & = & \Delta t k A_n \\ & x_1 & = & A_1 \\ & -d_n + x_n & \leq & 37 \\ & d_n - x_n & \leq & -37 \\ & x_n, d_n, A_n \in \mathbb{R}, \quad u_n \geq 0 & & \end{array}$$

we can see that the objective is merely a linear combination of variables whether they are independent or dependent. Similar can be said regarding the constraints. Thus, the entire model is a pure linear problem[BT97a].

To solve the problem in each ambient temperature scenario, we decided to use MATLAB's function `linprog`. The code starts off with initializing the constants—ambient temperatures (A_n from model), time interval in hours ($0 \leq t \leq 24$ discretized with Δt), the heat resistance of the incubator (k in the model), the heat and voltage proportion constant (x in the model), and the weights or costs (w_1, w_2 in the model)[24b]. We then follow the solution algorithm.

4.2 Integer Program

Looking at the other model we developed earlier

$$\begin{array}{lccc} \min & \sum_{n=1}^{N} w_1 d_n + w_2 u_n + w_3 \left(v_n^+ + v_n^-\right) & & \\ \text{s.t.} & x_{n+1} + (k\Delta t - 1)x_n - c\Delta t u_n & = & \Delta t k A_n \\ & u_n - u_{n+1} + v_n^+ - v_n^- & = & 0 \\ & v_n^+ + v_n^- & \leq & 1 \\ & x_1 & = & A_1 \\ & -d_n + x_n & \leq & 37 \\ & d_n - x_n & \leq & -37 \\ & u_n, v_n^+, v_n^- \in \{0,1\}; \quad x_n, d_n \geq 0 & & \end{array}$$

we can see that this objective is also a linear combination of dependent and independent variables. Furthermore, when we look at the constraints, we see that all the dependent variables are linear combinations of the independent variables much in a similar way to the linear programming model. We know that the combination of linear functions is linear[BT97a]. However, in this model, we have variables that are binary decision variables which are a type of integer variables. Thus, the problem is a linear integer program.

To solve the problem in each ambient temperature scenario, we decided to use MATLAB's function `intlinprog`. Much like with the pure linear program, the code starts off with initializing the constants—ambient temperatures (A_n from model), time interval in hours ($0 \leq t \leq 24$ discretized with Δt), the heat resistance of the incubator (k in the model), the heat and voltage proportion constant (x in the model), and the weights or costs (w_1, w_2, w_3 in the model).[24a] Then we follow the solution algorithm.

4.3 Solution Algorithm

To solve our problem, we decided to use MATLAB's `linprog` and `intlinprog`. The initial section of the code defines all the constants, such as the time interval, the various ambient temperatures, k, c, and the weights. After that, the sparse matrices were set up. Here was process for making A_{eq} (from the integer model) sparse as an example:

First, we had to notice that A_{eq} is made up of 5 submatrices corresponding to the coefficients of d_n, u_n, x_n, v_n^+, and v_n^-. The matrix associated with the coefficients of d_n is characterized as

$$\mathbf{d} = \begin{bmatrix} 0 & & & & \\ & 0 & & & \\ & & 0 & & \\ & & & \ddots & \\ & & & & 0 \end{bmatrix}$$

The matrix associated with the coefficients u_n is characterized as

$$\mathbf{u} = \begin{bmatrix} -c\Delta t & 0 & 0 & \\ 1 & -1 & 0 & \\ 0 & -c\Delta t & 0 & \\ 0 & 1 & -1 & \\ 0 & 0 & -c\Delta t & \\ 0 & 0 & 1 & \ddots \end{bmatrix}$$

The matrix associated with the coefficients x_n is characterized as

$$\mathbf{x} = \begin{bmatrix} 1 & 0 & 0 & \\ 0 & 0 & 0 & \\ kt-1 & 1 & 0 & \\ 0 & 0 & 0 & \\ 0 & kt-1 & 1 & \\ & 0 & 0 & \ddots \end{bmatrix}$$

The matrix associated with the coefficients v_n^+ is characterized as

$$\mathbf{v}^+ = \begin{bmatrix} 0 & 0 & 0 & \\ 1 & 0 & 0 & \\ 0 & 0 & 0 & \\ 0 & 1 & 0 & \\ 0 & 0 & 0 & \\ 0 & 0 & 1 & \\ & & & \ddots \end{bmatrix}$$

The matrix associated with the coefficients v_n^- is characterized as

$$\mathbf{v}^- = \begin{bmatrix} 0 & 0 & 0 & \\ -1 & 0 & 0 & \\ 0 & 0 & 0 & \\ 0 & -1 & 0 & \\ 0 & 0 & 0 & \\ 0 & 0 & -1 & \\ & & & \ddots \end{bmatrix}$$

Together these 5 matrices become the matrix

$$A_{eq} = \left[\begin{array}{c|c|c|c|c} \mathbf{d} & \mathbf{u} & \mathbf{x} & \mathbf{v}^+ & \mathbf{v}^- \end{array} \right]$$

The matrix A_{eq} is a $2N \times 5N$ matrix.

Rather than defining the entire matrix as sparse and then filling in the entries, it was more feasible to define the component matrices separately. To define a sparse matrix in MATLAB, you need to know the vectors i, j, and v, corresponding to the i, j (row, col) indices of the matrix and its value in v. For instance, u_n can be defined with $i = [1, 2, 2, 3, 4, 4, 5, \ldots]$ where the pattern is that every number from 1 to N is in the vector, but even numbers are entered twice. However, the last even element only has one entry. The j vector looks like $[1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, \ldots]$ where every number from $[2, N] \subseteq \mathbb{N}$ repeats 3 times and two 1's are appended onto the beginning of the vector. The values at each of these i, j pairs is defined by a repeating sequence of `-c*t`, `1`, and `-1`. These vectors are implemented by the code shown below:

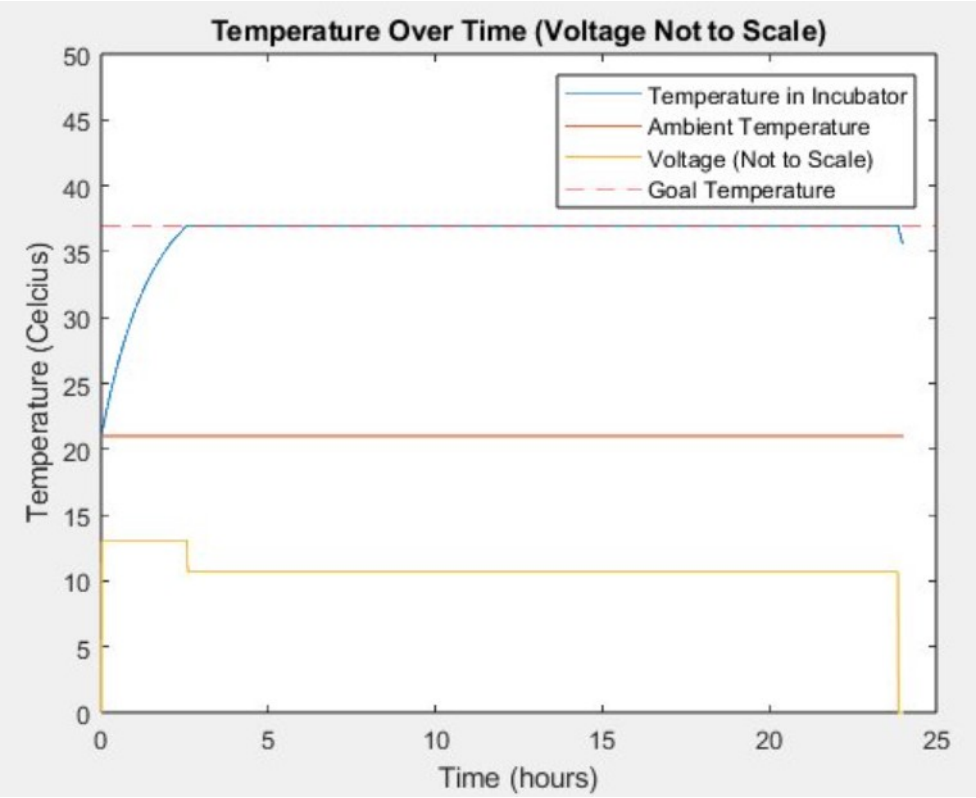
```
%uneq setup
rows = [];
for i = 1:2*N
    if mod(i, 2) == 1
        rows = [rows i];
    else
        rows = [rows i i];
    end
end

rows(end) = [];
cols = repelem(2:N, 3);
cols = [1 1 cols];

uneqVals = repmat([-c*t 1 -1], 1, N);
uneqVals(end) = [];
uneq = sparse(rows, cols, uneqVals, 2*N, N);
```

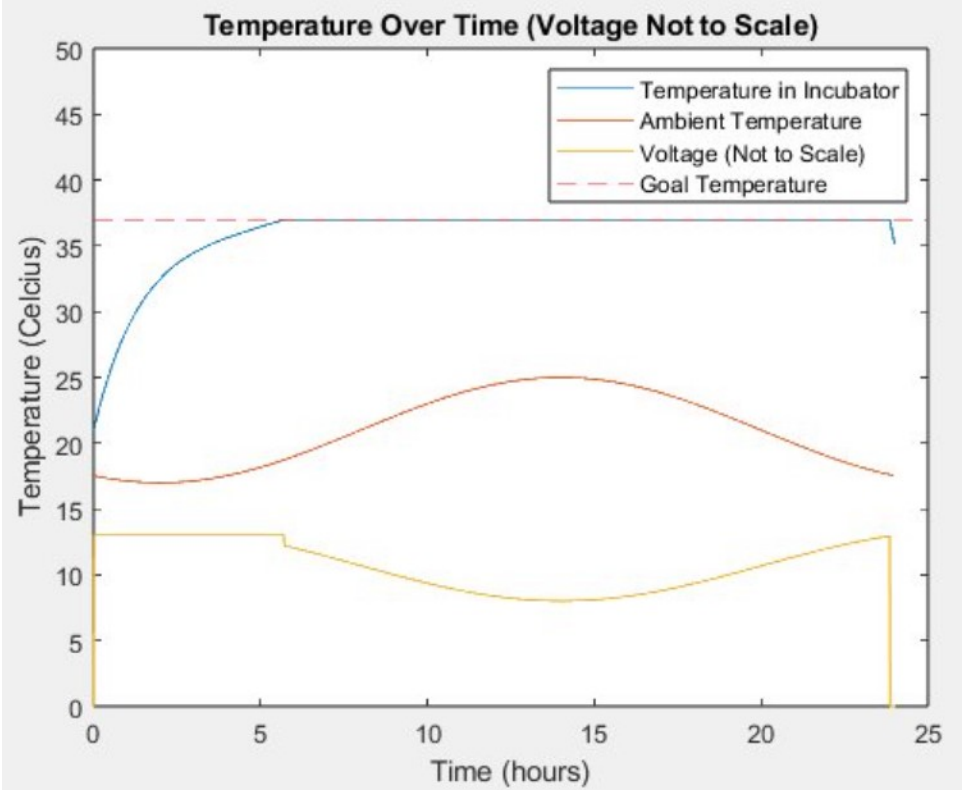
A similar process is done for each of the other submatrices except for d_n, which can be defined with a sparse matrix of all zero's. The benefit of using these sparse matrices is that they only allocate memory for the non-zero components of our matrix, and in the case of d_n, we save a $2N \times N$ matrix's worth of memory. After all of the components are defined, we used MATLAB's `horzcat` to combine them into the single A_{eq} matrix, and we repeat the process for A_{ineq}. Finally, the b_eq and b_{ineq} vectors are defined (non-sparsely, as they are mostly non-zero elements) and `intlinprog` can be called[24a].

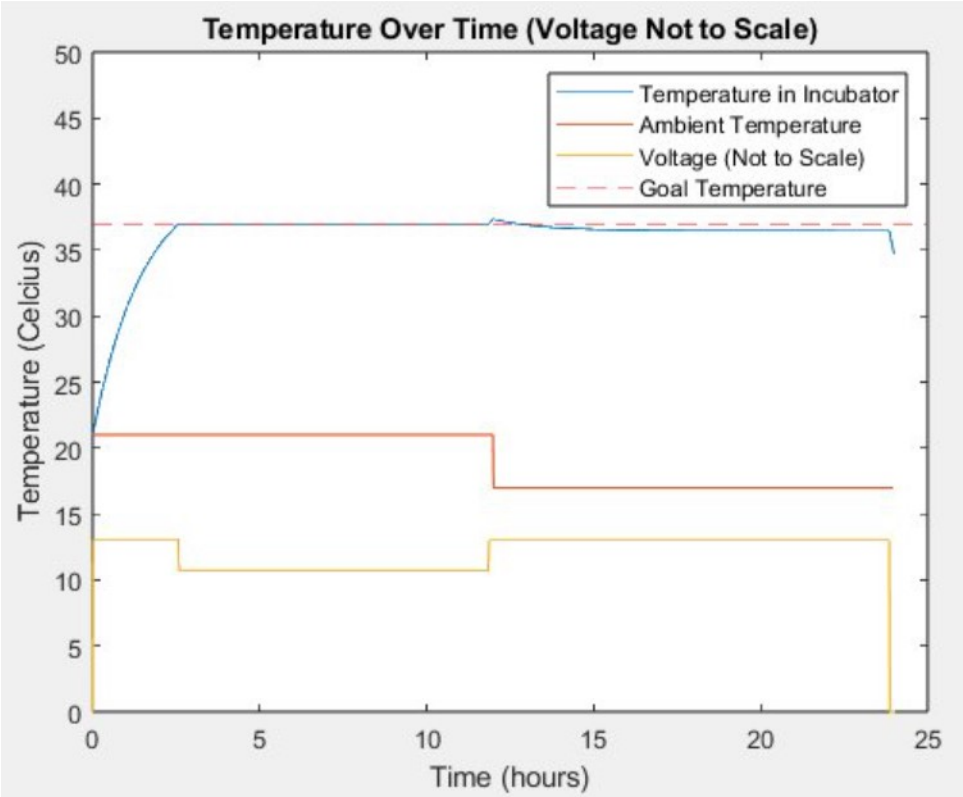
5 Results

5.1 Graphs

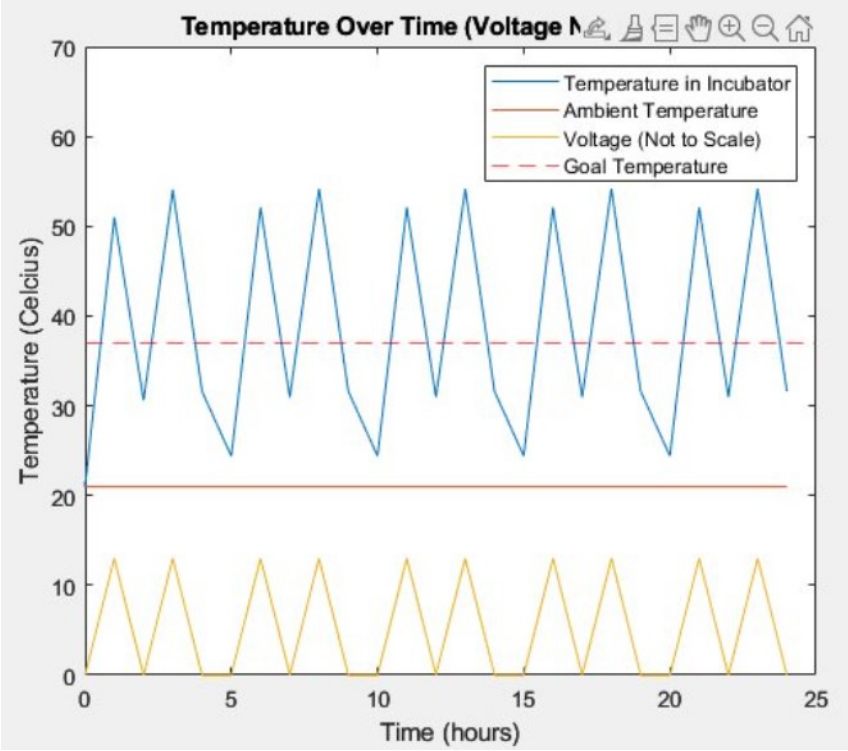
5.1.1 Linear Program

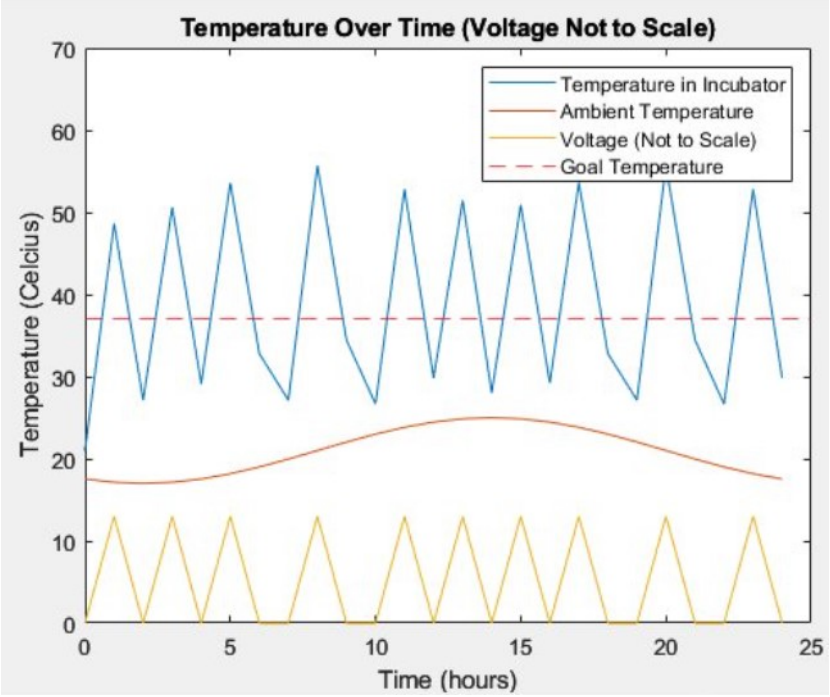


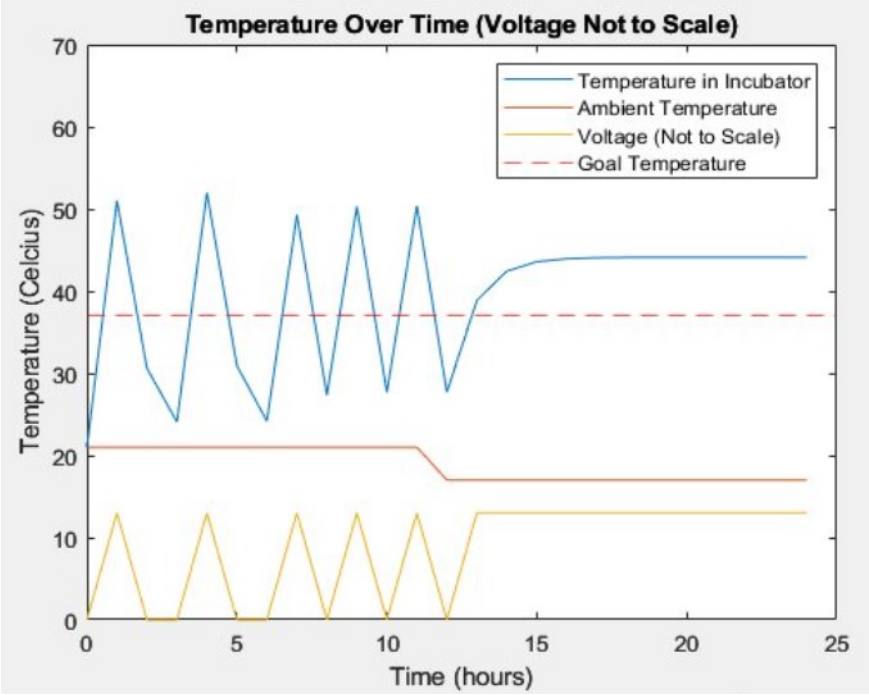

Temperature Over Time (Voltage Not to Scale)
Temperature in Incubator
Ambient Temperature
Voltage (Not to Scale)
Goal Temperature
Temperature (Celcius)
Time (hours)
0
5
10
15
20
25
30
35
40
45
50


Temperature Over Time (Voltage Not to Scale)
Temperature in Incubator
Ambient Temperature
Voltage (Not to Scale)
Goal Temperature
Temperature (Celcius)
Time (hours)
0
5
10
15
20
25
30
35
40
45
50

5.1.2 Integer Program







5.2 Analysis

For both the linear and integer models, the objective value does not have any specific meaning, other than that it should be as small as possible. For the Linear program with periodic temperature, we were pleased to see that the voltage sinusoidally fluctuated to meet the changes in the ambient temperature. In the case of the sudden temperature drop, the most significant detail of our data was that the voltage proactively increased before the temperature change. For the integer program, the temperature would keep jumping above and below the goal temperature because of the on/off nature of the heating element. It is worth noting that in the sudden drop case, the voltage stayed on for the remainder of the time and stabilized with the internal temperature.

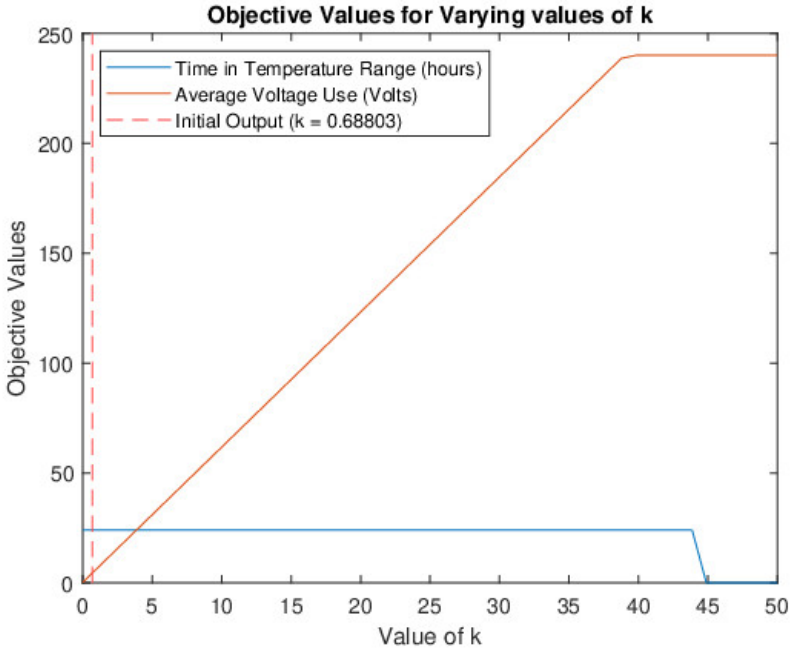
The running time of our linear model was about 47.5 seconds for 86,400 iterations (a one second time interval). For the integer model, this increased to 2170.27 seconds with 30 minute time steps. Clearly, the integer model was much slower, which is due to the nature of the branch and bound algorithm used in `intlinprog`.

6 Discussion

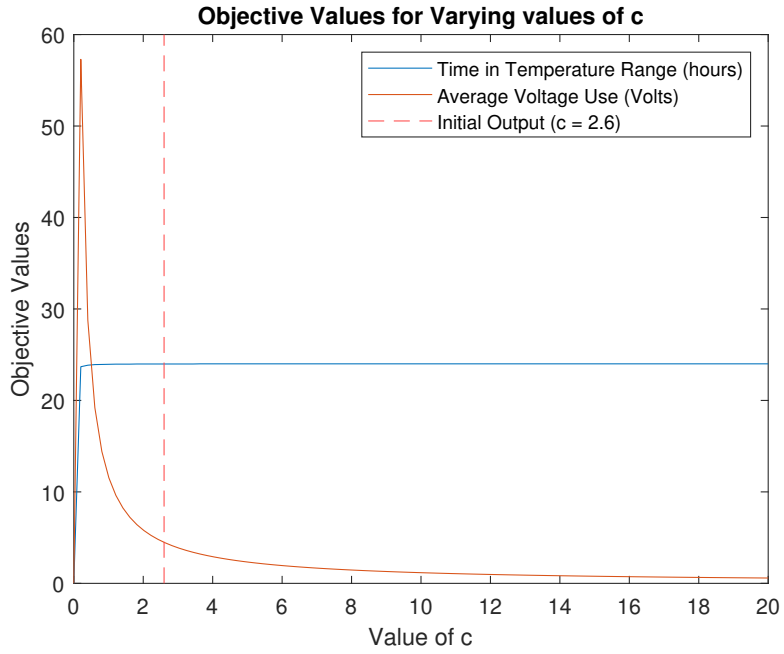
Our model was able to produce a voltage schedule that kept the internal temperature within the allowable range for as long as possible. Our model allows for varying time steps, so if the incubator needed to be controlled by a human, more reasonable time steps of a minute or half-hour could be used to calculate an optimal schedule.

A more important question in sensitivity analysis is how much the objective value changes per unit change in one of our parameters. The issue here is that our objective function, while useful for solving the model, is mostly meaningless. This is due to our model essentially being a multi-objective optimization problem that we have made simpler by assigning weights to each objective. Thus, for our sensitivity analysis, we will use our solution to generate a "new" objective that is more useful for our analysis. Our new voltage objective is simply defined as the average voltage over the time period. This acts as a more succinct and to-scale view of the voltage use. The new objective for the temperature is defined exactly as stated by the prompt. We defined it to be the total time that the temperature is in the ideal range of 35-39. Using these two new "objectives" we can gain a more meaningful understanding of how the parameters affect our solution.

Below, we plot the output values of these objectives for a range of values for k. As seen in the graph, up to a certain point, the value of k has almost no effect on the time spent in the temperature range. This makes sense as with our current value of k, the incubator needs to use very low voltage to maintain a consistent temperature. However, the average voltage use does seem to scale linearly with k. Since k corresponds to the insulation of the incubator, a higher value of k implies more heat transfer between the internal and external systems. As a result, as k increases, the incubator needs to use consistently higher voltage to stay in the ideal temperature range. This linear increase continues until k is equal to about 40, at which point the voltage is maxed out throughout the entire 24-hour period. Eventually when k is equal to about 45, the heat transfer with the environment is so extreme that our system is unable to even reach the ideal temperature range. However, a k value of 45 is quite extreme and no realistic system should ever exceed this value, so the only relevant change is in average voltage. In this case, an increase in k by 1 unit corresponds to an increase in average voltage by about 6.15.

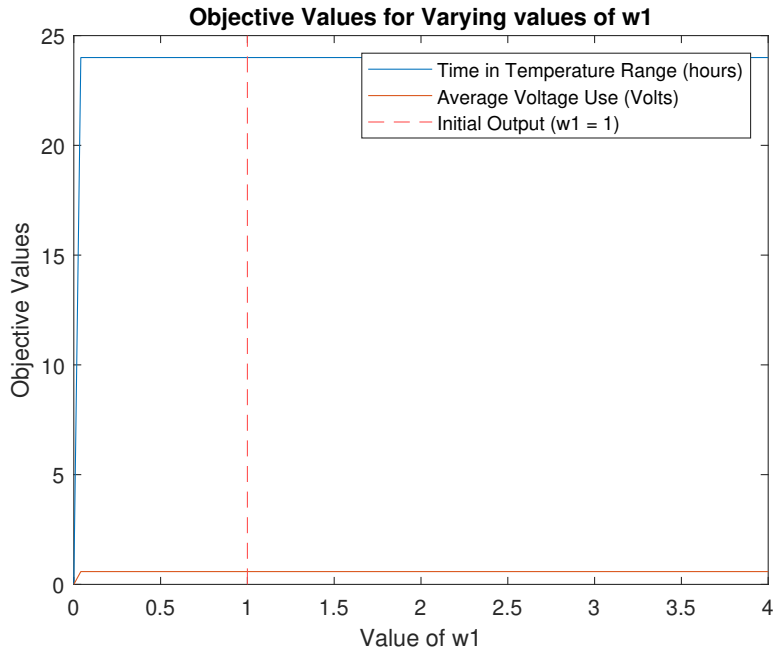


Next, we look at the effect of changing c. We know that the value of c roughly represents the relation between voltage and temperature. With a higher value of c, the voltage doesn't need to be as high to maintain the ideal temperature. This relationship can be seen in the graph below, though unlike the last one, the relationship is no longer linear. Similar to the last one, the value of c has practically no effect on the time spent in the temperature range. However, it does have a somewhat large effect on the average voltage use. The magnitude of the effect does depend on what the current value of c is though at least locally, relative to our experimental value of 2.6, a unit increase in c would correspond to a decrease in voltage by about 1.71.

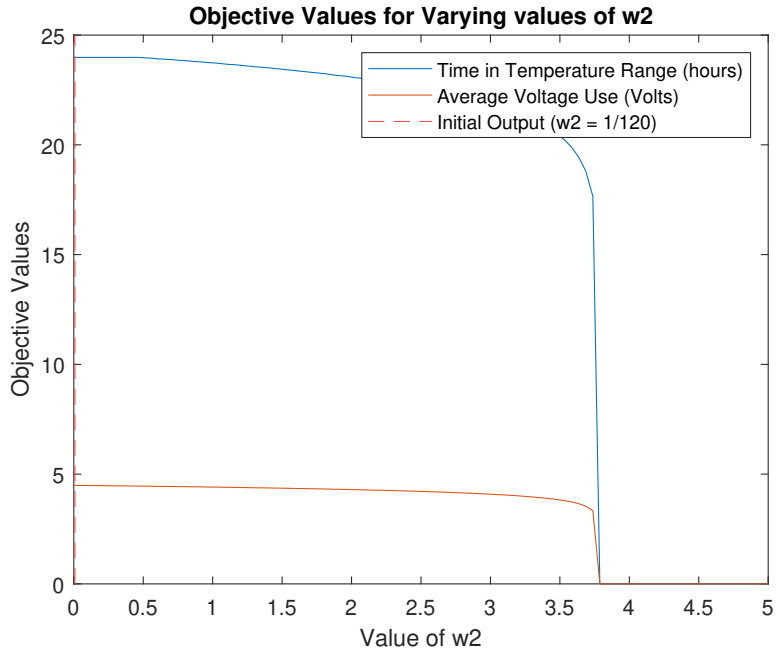


An interesting thing to note here is that results could be used in decision making for somebody utilizing an incubator. Since the time in temperature range is practically independent of the values for c and k, if somebody would like to minimize the temperature usage, these results can highlight what is more important to improve. Because the relationship between average voltage and k seems to be linear, the value of c could determine which improvement would have the greatest impact. Say a researcher has an incubator that has our exact parameters for c and k. Obtaining a better heating element would only decrease the average voltage by a rate of about 1.71. Meanwhile, finding a way to insulate the system could theoretically decrease the average voltage need by about 6.15. In this scenario, it is clear which improvement should be made to obtain the optimal increase in efficiency.

The sensitivity of w_1 is likely the least interesting of them all. Our system already reaches the optimal temperature range as fast as is physically possible, so increasing the "importance" of being in the ideal temperature range has no effect on the optimal solution. It is not visible in the graph but, decreasing w1 enough would eventually give more importance to the voltage and cause the output to change. However, because these weights are arbitrary and relative, increasing w_2 would have the same effect. As a result, it is safe to say that changing w_1 has no effect on the optimal solution of the program.



The choice of w_2 is somewhat confusing as we chose to scale our voltage values between 0 and 1. This was primarily done to make it simpler to convert to the integer solution method, but it also makes it easier to change the voltage range in our model. We are currently working under the assumption that we are using a 240 V system, so we wanted a value of w_2 that would scale with the maximum voltage (since our decision variables can only be between 0 and 1). However, for somewhat obvious reasons, we could not simply use the variable times the maximum voltage as this would have given far too much weight to the voltage. As a result, we simply chose to use $w_2 = \frac{1}{120}$ and then multiply this by our maximum voltage to use in the model. In this case, it corresponds to an actual weight equal to 2. Thus, the graph shown below uses $\frac{1}{120}$ as our value of w_2 and not 2.



Looking at the graph, we see some interesting results. Initially, increasing the importance of voltage use has very little impact on the overall solution. Even increasing the weight by 3 units only causes a decrease in voltage by about .5 V. However, this same increase causes a drop in the time spent in the temperature range of about 2 hours. There is practically no real-life scenario where this small power save is worth the drastic temperature change. Furthermore, once the weight hits about 3.78, saving voltage becomes so "important" that it just sets everything to zero and the temperature never increases.

We can make some interesting conclusions based of these last two graphs. The first is that the choice of w_1 truly does not matter. Because these weights are arbitrary, for any choice of w_1, a value of w_2 can be chosen that would produce the same results. The second conclusion we can make from this is that the temperature is clearly the more important factor here. Even if you really care about saving energy in your system, saving even a little bit of energy would correspond to a drastic change in the temperature. The better method for saving energy would be to invest in a better insulated system or a heating element that converts energy to heat more efficiently.

Another question in sensitivity is how much the parameters can change before the optimal basis changes. By using a modified bisection algorithm we can calculate the approximate range in which the optimal basis remains the same. The table below presents the results.

Constant Ambient Temperature (240 V)			
Parameter	Current Value	Maximum Increase	Maximum Decrease
k	0.68803	27.0043	0.6849
c	2.6	1.40	0.5885
w1	1	Inf	0.8054
w2	0.0083	0.0345	0.0083

From the table, we can draw many of the same conclusions as before. What is interesting however, is that while the basis may change outside of a certain range, the parameter can sometimes have a wider range where the solution is "mostly" the same. For example, the optimal basis changes once k increases by about 27. However, as seen in the graph, k can increase by about 45 before any significant change occurs.

Our models were fairly time and space efficient even given very large N values thanks to the sparse matrix implementation. Notably however, the integer version takes a much longer amount of time, though it does terminate in a somewhat reasonable window. The linear model was much quicker when you don't need to consider the frequency of turning the voltage source on and off.

One potential weakness of our model was that the voltage "predicted" the sudden ambient temperature drop, which is not realistic of course. In order to improve our model, we could make it so it only uses historic data when choosing voltage values. Another improvement we could make is incorporate stochastic modeling[BT97b], by allowing certain variables to have a level of randomness added to it. For example, the ambient temperature should have a level of variation even in a "constant temperature" environment. These changes would increase the robustness of our model to react to more realistic data.

References

[BT97a] Ditmitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[BT97b] Ditmitris Bertsimas and John N. Tsitsiklis. "Introduction to Linear Optimization". In: Athena Scientific, 1997. Chap. 6.

[SH16] Gilbert Strang and Edwin Jed Herman. *Calculus*. Vol. 2. OpenStax, 2016. URL: https://openstax.org/books/calculus-volume-2/pages/4-2-direction-fields-and-numerical-methods.

[FRY21] Joel Feldman, Andrew Rechnitzer, and Elyse Yeager. *CLP-1 Differential Calculus*. The University of British Colmumbia Department of Mathematics, 2021. URL: https://personal.math.ubc.ca/~CLP/CLP1/clp_1_dc/sec_newtonCooling.html.

[24a] "intlinprog". In: *MathWorks* (2024). URL: https://www.mathworks.com/help/optim/ug/intlinprog.html.

[24b] "linprog". In: *MathWorks* (2024). URL: https://www.mathworks.com/help/optim/ug/linprog.html.