Energy Systems and Control
[CE295]

Book of Projects 2018

Instructor: Professor Scott Moura
Grad. Stud. Instructor: Bertrand Travacca

Preface

We are extremely proud to present the first *book of projects* made for the CE295 class (Energy Systems & Control). CE95 is a class given every year in the spring at UC Berkeley in the department of Civil & Environmental Engineering. This class gives an introduction to control system tools for students interested in energy system applications. Applications of interest include batteries, electric vehicles, renewable energy, power systems, smart buildings/homes and more! Technical tools include system modeling, state-space representations, stability, parameter identification, state observers, feedback control, and optimization.

Each year students engage in a semester-long course projects. For many years we have noticed the quality of the projects. The class of 2018 displayed such excellence that we decided to collect all the projects here. This book is aimed at future students, who will find essential material to get inspiration and build upon previous work made in the class. Note that this book could also be of interest to people from industry or academia as truly innovative ideas are presented.

We organized the book in five different thematic parts: Microgrids (part I), Electric Vehicles and Transportation (part II), Building Energy Management (part III), Batteries and Energy Storage (part IV), Health and Environment (part V).

We wish you a pleasant reading,

Professor S.Moura,
B.Travacca

Table of Contents

Part I

Microgrids

Microgrid Control Platform

Ramon Crespo, Ioanna Kavvada, Guo Jun Li and Dieter Smiley

Abstract

This study designed, implemented and analyzed an optimization program as a Microgrid Control Platform (MCP). The MCP takes into account the state of each of the Microgrid's assets, Photovoltaic (PV) generation, electric vehicle (EV) demand and building load demand, as well as future costs to electricity. The objective of the optimization program is to minimize operational costs. The forecast states of PV generation, EV demand and building load demand are based on forecasts made from individual data sets, using Markov chains, ARX machine learning models and random forest regression in each of the respective modeling processes. Ultimately, results show that utilizing these forecasting methods in a moving-time-horizon linear optimization program may yield additional revenue beyond a simplified rules-based control scheme. The results also indicate that the optimal time horizon for standard operations is 12 hours. Future research should focus on shortcomings of the MCP designed, which includes accounting for seasonal variation and improving robustness to forecast errors.

Introduction

Motivation and Background

The world energy demand is projected to rise through 2050 with an average annual growth about 0.9 percent, as reported in [1], and is expected to increase energy-related carbon dioxide emissions. Energy generating units based on renewable energy sources are major components of the strategy to reduce harmful emissions and deal with depleting energy resources. However, they are less reliable as compared to the conventional fossil fuel-based power generating systems due to their intermittent nature, as stated in [2]. Integration of different renewable energy sources coupled with energy storage system can add reliability in the power systems.

In light of this trend, a significant shift in the electricity structure of communities is needed. These communities will require grid resiliency and reliability, consideration of carbon-free energy resources while also taking into account the economics and profitability of buying and selling electricity in a shared grid. One such vision is a decentralized electricity structure where communities are semi-self sufficient through distributed energy resources locally while still possessing a connection to the grid, also known as smart grids.

Smart grids consist of numerous components and controls that are still being developed and tested. According to [3], the full implementation of the smart grid will happen in the next decade, when technology is mature enough to be implemented in a commercial scale. As the technology and efficiency of the components of the smart grid improves, the need for the right control tools increases. The MPC seeks to fill this gap. The platform will calculate the optimal use of resources to maximize energy efficiency and minimize CO2e emissions by taking into account forecasts of pricing, building load demand, other load demands of a community as well as variability in renewable generation.

Focus of this Study

The focus of the study is the design and implementation of a MCP that takes into account PV generation forecasts, building energy demand and EV energy demand to optimize the use of electricity, from and economic and environmental point of view. In particular, emphasis is placed on cost performance, the time horizon required for adequate performance and the MCP's robustness to error.

Literature review

This project will use the design of the EcoBlock project in Oakland, California to test the MPC. EcoBlock is an existing project that seeks to design, build and test a solar-powered urban system that uses PV energy generation and flywheel storage to meet the electric demand of the buildings and EV fleet. Though the MCP will be designed using an existing project, it will be easily scalable for implementation on other systems.

The definition of a Microgrid may vary dependent upon the article, but generally tends to contain three important characteristics: a group of interconnected load and distributed energy resources (DERs); a clear point of common coupling with the larger electric grid; and an ability to connect and island itself from the grid. More generally, [4] describes that Microgrids contain sources of generation, load, and components capable of energy storage that are connected to the macro grid at a single point. Energy storage is an important consideration for the optimal performance of a Microgrid, as it enables an energy system to balance supply and demand, and enables the shifting of dispatchable energy in time.

In addition to the necessary physical equipment for generation, storage and dispatching electricity, other components are required to design an optimal Microgrid. An essential ingredient of Microgrids and in optimal energy management is the utilization of energy controllers, or sensors capable of providing accurate information in real time and making decisions. According to [5], the capability of electricity networks to "facilitate the fluent interaction of all users connected to it" enables bi-directional power flow and is a vague definition of the smart grid concept. Controller and sensors allow for consumption-driven adjustment by monitoring energy use performance in real time or in discrete time intervals. This data is also used to generate predictions for future energy consumption and generation.

Many methods of forecasting electricity demand and generation exist and the optimal method may vary dependent upon application and prediction horizon. As in [6], some common forecasting methods reviewed include time series, regression, and autoregressive integrated moving average. Beyond attaining data and controls, to provide optimal energy

management there exists a need for real-time computational intelligence methods such as an optimization program, as in [7]. The inclusion of correct data for the programs application are critical to the utility of the program. Appropriate information may include historic and forecasted weather data, expected load profiles for the microgrid, and knowledge or predictions of the pricing schedule for retail electricity rates. The objective of the optimization algorithm influences what may be considered relevant data. For example, other considerations and goals of a Microgrid which are not in the scope of this report may include peak load shaving or voltage regulation, as described in [8]. Different objective functions will, necessarily, use different optimization algorithms.

Cost reduction and DER power generation maximization are two main objectives when planning for the deployment of renewable energy sources. Different optimization techniques have been proposed to achieve these two objectives. [9] describes various methods of optimization techniques for various renewable energy sources. Other papers, as in [2], show various optimization models, such as renewable energy models, emission reduction models, energy planning models, energy supply and demand models, forecasting models and control models for efficient utilization of the renewable energy sources. Ultimately, the real time demand response model over a twenty-four hour planning horizon found in [5] served as a framework for our team's energy management controls.

Renewable energy sources can be used in grid connected as well as in stand-alone modes. Grid connected hybrid renewable energy sources are promising areas of research as these are expected to provide the same level of power supply reliability that can be achieved from the conventional fossil fuel based energy systems.

Key contributions

Our research has contributed to the understanding of time horizon selection when attempting to minimize computational power, as well as a better understanding on how to forecast Microgrid components and apply them to a simple MCP program.

1 Technical Description of MCP

Here we will outline in detail (i) the technical components of the Microgrid studied; (ii)how our MCP optimizes for cost-minimizing Microgrid operations; and, (iii) how the forecast of each Microgrid component of interest is generated.

1.1 Descpription of Microgrid

The MicroGrid (Figure 1) consists of 63 residential buildings, 24 electric vehicle supply equipment (EVSE) each serving one electric vehicle, PV area of 27,750 ft^2 and 480 kWh of flywheel storage. Solar power generated by the solar panels is collected at a single point and can be either stored in the flywheel, sold to the grid or used to satisfy EV and house load demand. The MicroGrid acts as a unit with a single connection to the grid.

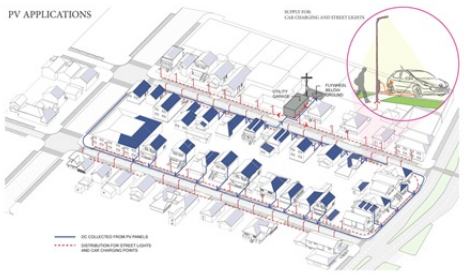


Figure 1: Schematic description of the microgrid

1.2 Optimization Program Formulation

The feasibility and environmental-friendliness of a MicroGrid is ultimately based upon the cost-effectiveness of the Microgrid, or how much energy needs to be bought or sold from the electrical grid. Hence, the MCP designed and implemented in this study focuses on minimizing the overall operations cost over a set time horizon. This is done through a moving-time-horizon linear optimization program. The program setup is listed below.

Objective function:

$$\min \sum_{i=k}^{k+L} c_I(i) G_I(i) \Delta t - \sum_{i=k}^{k+L} r_E(i) G_E(i) \Delta t \tag{1}$$

Subject to: $for \;\; i = k, k+1, ..., k+L$

$$S(i) + B_d(i) + G_I(i) = L(i) + B_c(i) + G_E(i) + C_{2,1}(i) \tag{2}$$

$$0 \leq G_I(i) \leq G_{I,max}, \quad 0 \leq G_E(i) \leq G_{E,max} \tag{3}$$

$$E(i+1) = E(i) + [\eta_c B_c(i) - (\frac{1}{\eta_d}) B_d(i)] \Delta t \tag{4}$$

$$E(k) = E_{meas} \tag{5}$$

$$E(k+L) = E_{final} \tag{6}$$

$$0 \leq E(i) \leq E_{max} \tag{7}$$

$$0 \leq B_c(i) \leq B_{max}, \quad 0 \leq B_d(i) \leq B_{max} \tag{8}$$

$$Z_{2,1}(i+1) = Z_{2,1}(i) + \eta_2 C_{2,1}(i) \Delta t \tag{9}$$

$$Z_{2,1,}(k) = Z_{2,1,k,meas} \tag{10}$$

$$Z_{2,1,min}(i) \leq Z_{2,1}(i) \tag{11}$$

$$0 \leq C_{2,1}(i) \leq C_{2,1,max} \tag{12}$$

A list of variables and their definitions are provided in the Appendix.

In essence, this optimization program minimizes the total cost (cost minus revenue) of importing or exporting electricity over the next L time periods, as shown in (1). In (1), we see that cost or revenue at each time step is determined by term $G(i)$, which, in (2), we see is simply the difference between supply of energy and consumption of energy within the Microgrid. In order to control the outflow and inflow of energy based on forecast price, generation and consumption, the flywheel storage, or B_c and B_d terms in (2), are used as the control for the system.

The remainder of the optimization program is limited by power and energy constraints, as well as the dynamics of each component. The power limits of the grid are included in the optimization through (3). The flywheel storage dynamics are shown through (4), where B_c and B_d are the charging and discharging mechanisms of the storage system. The initial and final conditions for the flywheels are shown through (5) and (6), respectively, and the limits of the energy state of the flywheel storage is shown through (7). Charging constraints, or the maximum charging can occur at each time step, is represented through (8). Finally, EV demand is characterized by a cumulative EV energy demand, represented by $Z_{2,1}$, and a charging power, represented by $C_{2,1}$. From the constraints, (9) represents the dynamics of EV charging, while (10) shows the initial condition of EV charging. The energy and power limits constrain the optimization problem via the inequality constraints (11) and (12).

Note that there is no need to model the conditions of the flywheel in our study, as flywheel losses are assumed to be very low compared to the energy being transferred on an hourly basis. The flywheel storage is simply computed iteratively at each time step given the value at previous time step and charging or discharging commands. The charging and discharging efficiencies are 0.95 and 0.95, respectively, and are found in [10].

1.3 Microgrid Component Forecasting

In order for the supply and consumption of energy by the Microgrid to be balanced as in (2), each component must be modeled separately. Here, we show how we used prior data to model each of the Microgrid's assets.

1.3.1 Electric Vehicle Demand

Electric vehicle demand is taken from [11]. The year-long data acquired was used to create a model that predicts the future EV demand based on previous data. Ultimately, two models were used and compared to see what differences could be observed through using a variation of an Average Model and a Machine-Learning model.

For the average model, the average EV demand for the month in question (May) was used. Then, a normally-distributed error is injected into each hour. This is shown in the formulation below.

$$E(i+1) = E_{avg}(i) + N\left(\frac{E_{avg}(i) - E(i)}{2}, \frac{|E_{avg}(i) - E(i)|}{8}\right) \tag{13}$$

Essentially, as shown in (13), the model attempts to correct for deviations from the average by compensating with a normally distributed error that is centered closer to the average. The standard deviation of the normally-distributed component in (13) was chosen arbitrarily.

The resultant graph of predictions for a 24-hour time-horizon is shown in Figure 2.

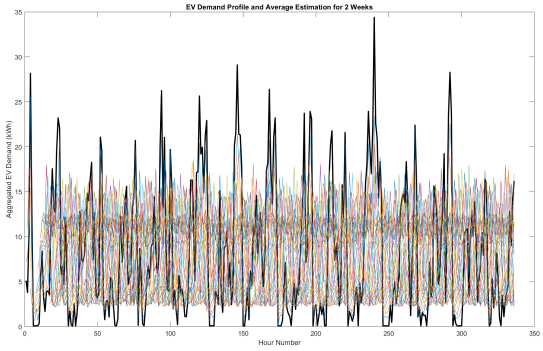


Figure 2: Average Model forecasting of EV Demand (all colored lines) as compared to actual EV Demand (solid black line)

The second model used was a 3-hour previous data input, Autoregressive with Exogenous Inputs (ARX) Model. Here, year-long data from [11] was used as training data to fit variables to a linear function that forecasts the next time step by linearly combining the monthly average with the three previous data points. This is formulated below.

$$E(i) = E_{avg}(i) + \alpha_1 E(i-1) + \alpha_2 E(i-2) + \alpha_3 E(i-3) \tag{14}$$

From (14), we can then use ordinary least squares regression to find the α coefficients in front of each previous term injected into the model. The resultant coefficients generated from the regression is shown in (15).

$$\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 0.3386 \\ -0.1353 \\ -0.1014 \end{bmatrix} \tag{15}$$

The resulting 24-hour forecast from this machine learning model is shown in Figure 3.

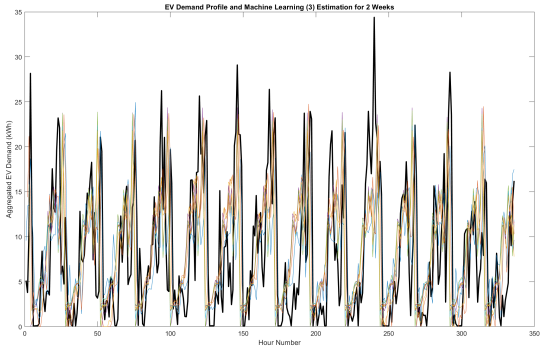


Figure 3: ARX Model forecasting of EV Demand (all colored lines) as compared to actual EV Demand (solid black line)

From Figures 2 and 3, we observe that the Average Model creates predictions that have high random noise and does not reflect the true nature of EV demand as often as the machine learning model does. This is reflected by the RMSE for each hour predicted by each model. The Average Model's RMSE values hovers around 0.35 kWh constantly, while the RMSE for the machine learning model begins at 0.34 and extends higher as time progresses. The slightly lower RMSE values for the ARX model, combined with the overall better shape of the predictions as shown in Figure 3 shows that the ARX model is a better forecast mechanism for the data at hand.

Ultimately, because of these reasons, the ARX model was used as the input data for the MCP in this study.

1.3.2 Photovoltaic Generation

Solar energy, which comes from the sun in the form of solar irradiance, can be directly converted to electricity by using photovoltaic (PV) technology. PV technology uses solar cells made of semiconductors to absorb the irradiance from the sun and convert it to electrical energy. The typical solar PV cell has an efficiency of about 15–20% [12].

The characteristics of a PV module can be demonstrated by power–voltage or current–voltage curves. Figure 4 shows the power–voltage curve of a PV module for different conditions of solar irradiance and cell temperature. The PV output power is dependent on solar irradiance and cell temperature. Low irradiance leads to low power, and high temperature causes a reduction in output power. The point on the curve at which the PV module delivers maximum power to the load is known as maximum power point (MPP). In the PV system, we assume that a maximum power point tracker will be used.

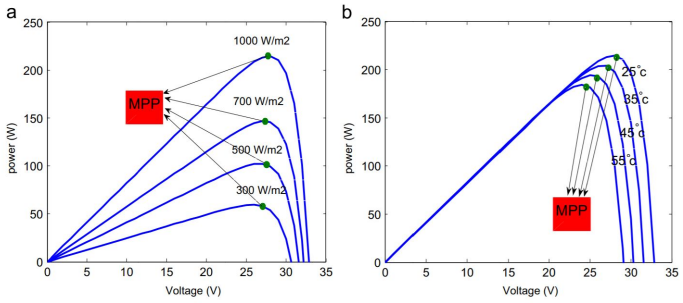


Figure 4: Power–voltage curve of a PV module for different conditions of solar irradiance and cell temperature [12].

The maximum power output is presented by [13]:

$$P = \eta \times S \times I \times (1 - \beta_{ref} \times (T - T_{ref})) \tag{16}$$

Where η is the conversion efficiency of the solar cell array (%) at the reference temperature T_{ref} at solar radiation flux of 1000 W/m^2 as seen in [14], S is the array area (m^2), β_{ref} is the temperature coefficient at reference conditions, I is the solar radiation (kW/m^2) and T is the outside air temperature ($^\circ C$).

The average value of β_{ref} reported in [14] is 0.0045 $(^{\circ}C)^{-1}$ at $T_{ref} = 25^{\circ}C$. In this analysis, the conversion efficiency of the solar cell array was considered to be 15%.

Hourly data regarding the Global Horizontal Irradiance (GHI), the total amount of short-wave radiation received from above by a surface horizontal to the ground, and the outside temperature for a typical meteorological year was retrieved from the National Renewable Energy Laboratory (Sacramento Municipal Utility District station) for the years 1998-2016. Using equation 16, historical solar power generation data was generated.

Each year was split into 4 seasons that exhibited similar solar power generation profiles (Season 1: May, June and July, Season 2: November, December, January and February, Season 3: October and March, Season 4: April, August and September). For each season, the hourly solar power generation values are not normally distributed about the hourly averages (Figure 5) and the hourly PV generation values are correlated to power generation of the previous time steps. Therefore, to take into account the correlation with previous time steps, a discrete-time Markov Chain model was selected as the most suitable forecasting method for PV power generation.

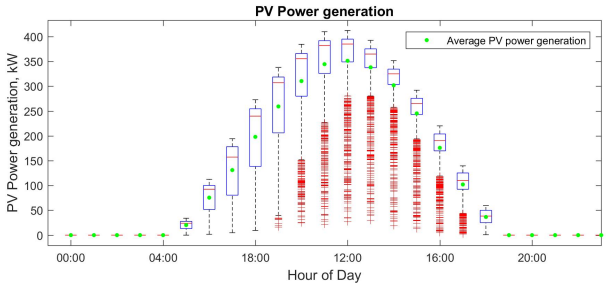


Figure 5: Non-normal solar power generation distribution (summer-time)

The transition probabilities for each time k were defined as follows:

$$p_{ijk} = Pr[S_{k+1} = S^j | S_k = S^i, k] \qquad \forall S^i, S^j \in S, k = 0, ..., 23 \tag{17}$$

Variables S^i, S^j are different levels of solar power generation within a set S. Set S was created by discretizing the solar power generation values into 14 evenly spaced levels as follows:

$$\begin{aligned} S = \{&S_{min} + 0 \times (S_{max} - S_{min})/12, \\ &S_{min} + 1 \times (S_{max} - S_{min})/12, \\ &..., \\ &S_{min} + 11 \times (S_{max} - S_{min})/12, \\ &S_{min} + 12 \times (S_{max} - S_{min})/12\} \end{aligned} \tag{18}$$

The transition matrices were then used to create the cumulative transition probability matrices for each time of day k. An example of the PDF and CDF for a specific initial solar power generation level and time of day is presented in Figure 6. Using the CDFs and a uniform random number generator in the range 0 to 1, given the current solar power generation level and the current time step k, the solar power generation level of the next time step k+1 was predicted. Each time step has an initial solar power generation level and

a uniformly distributed pseudorandom number that has been generated in the range from 0 to 1. For the initial power generation lever, the pseudo random number is mapped onto the CDF to find the corresponding power generation level of the next time step by using linear interpolation. The predicted solar power generation level then became the current one for the next iteration. By performing N iterations, solar power generation levels can be predicted for a N-hour time horizon. Figure 7 presents the forecasted solar power generation values using a 6h-time-window for a 4 day period.

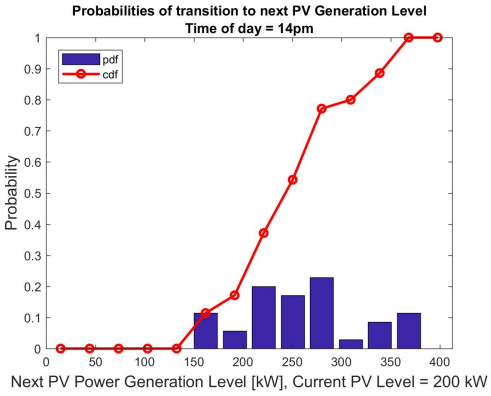


Figure 6: Transition probabilities for a specific solar power generation level and specific time of day

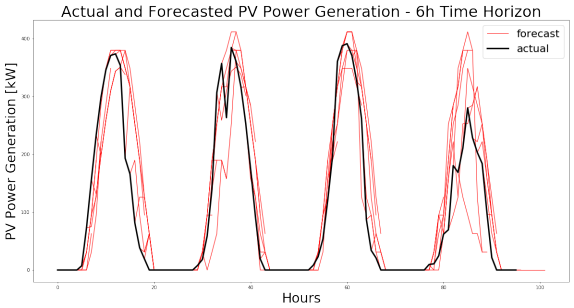


Figure 7: Forecasted solar power generation values using a 6h-time-window (red) and actual solar power generation values (black)

1.3.3 Building Load Demand

The data compiled to create prediction models and forecasts for building energy consumption was obtained by the EcoBlock team. This data was gathered from historic electric consumption recorded by smart meters and accessed through [15]. The historic electric demand readings varied among residents, depending on when their service with PG&E started and when their smart meter was installed, with some data sets dating back to the first quarter of 2016.

From [15], energy consumption of 23 individual housing units were obtained and filtered to ensure feasibility. In our study, we assume that at no time can a building realistically have

zero demand, so any hours containing zero energy consumption were removed. Initially, for every hour, an arithmetic average and median were determined to provide an initial estimate load profile, per unit, for any given day. The average and median of the raw data is shown in 8, and the accompanying mean and standard deviation are shown in the corresponding table in the Appendix. The data indicates that these residential building experienced two peak loads in a given day, with the maximum of the two occurring in the evening hours.

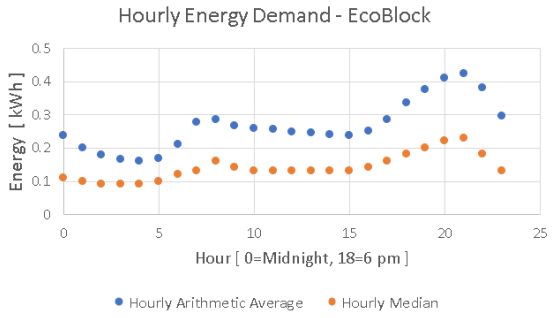


Figure 8: Average and Median Hourly Energy Demand for EcoBlock

Random forest regression was used in creating a prediction model for estimating building energy demand. According to [16], Random Forest Regression (RFR) is a tree-based method of supervised machine learning that fits prediction models to training data by using decision trees, random sampling, and a randomly selected subset of available parameters from which to sample. In building the decision tree, RFR stratifies the predictor space, the set of all possible values for parameters X_1, X_2,..., X_p into J smaller, distinct subsets, R_1, R_2,...,R_J. For each subset, RFR evaluates the mean response variable y_R, which in the case of building energy demand forecasting is kilowatts consumed in an hour.The RFR methodology iterates through the available parameters and regions at each decision node and determines the parameter, X, and splitting point, s, within the parameter, that provides the lowest total error in prediction, in the form of residual sum squared (RSS).

This is neatly described by [16] in (19) and (20).

$$R_1(j,s) = \{X|X_j < s\}, R_2(j,s) = \{X|X_j > s\}, \forall j, s \tag{19}$$

$$\min \sum_{i:x_i \in R_1(j,s)} (y_i - y_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - y_{R_2})^2 \tag{20}$$

Where y_{R_1} is the mean response of training observations in R_1, y_{R_2} is the mean response of training observations in R_2 and y_i is the actual observed response in the data.

The process of creating predictive models using RFR followed a systematic process: assemble variables of interest from available data that were expected to have strong explanatory power of energy consumption, use the training data to fit a random forest regression model in Python, and use that model to predict energy consumption for a test data set of two weeks. Examples of parameters expected to be powerful predictors and used within the regression include several hours of previous energy consumption of a building, building index within the data set, date and the time of the data point. If temperature data was available, it

would have been included, as [17] has shown that temperature has high predictive power for building energy consumption.

After generating predictions for various models, the mean absolute error (MAE) was evaluated for each model relative to the actual energy consumed from the test data. The best model, Model 36, was selected as that model, with it having the lowest MAE, 0.16 kWh (see orange line in 9. 9 compares Model 36 to actual data and average energy consumption for hour and day. The best performing model included building index number, hour of the day, day of the week, month of the year, and importantly, the previous two hours of actual energy consumption.

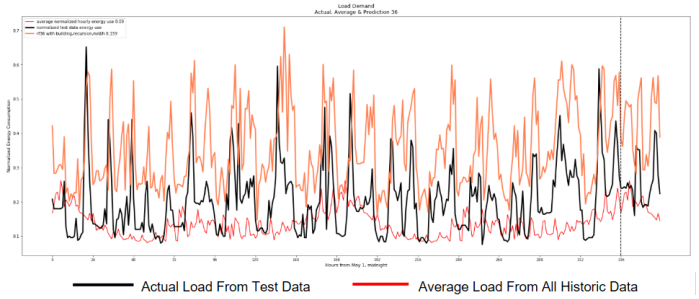


Figure 9: Forecast model output for building load demand compared to actual test data and average from all historic data.

1.3.4 Component Forecast Error

An important step in determining prediction models is establishing criteria for model evaluation. For each forecast method described, error was necessarily considered with respect to the test data set, i.e. the true values which the team is attempting to accurately predict. Typical error metrics considered are the mean absolute error, mean square error, and root mean square error.

Each component forecast method was designed with the ability to make predictions for every hour over a subsequent 24 hour time horizon. We decided to evaluate the mean absolute error of each model for every hour in the moving prediction horizon over the two week period of the test data. Figures 10, 11, and 12 describe the errors of each model. To enable easier comparison, the average mean absolute error of each predictive model is described as a percentage of its respective maximum hourly energy consumption or generation throughout the test period.

It appears that, for all models, accuracy of the predictions tend to worsen for estimates nearer the end of the prediction horizon. This indicates it may be more useful to run the optimization program more frequently than once every 24 hours. By evaluating the optimization program every hour, we expect to maintain a lower error between the decision variable implemented due to the forecast and the true energy need due to the actual energy balance from all microgrid components.

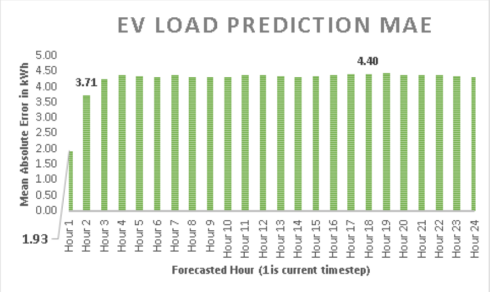


Figure 10: Forecast Error for Each Hour in the Prediction Horizon. MAE is 11.7% of Test Maximum Hourly Energy Demand

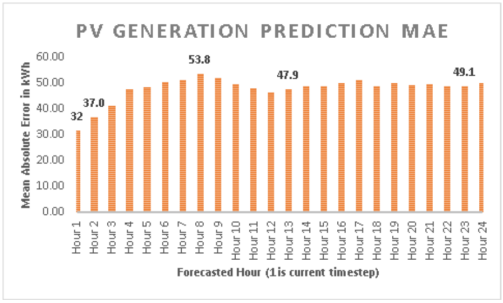


Figure 11: Forecast Error for Each Hour in the Prediction Horizon. MAE is 12.3% of Test Maximum Hourly Energy Generation

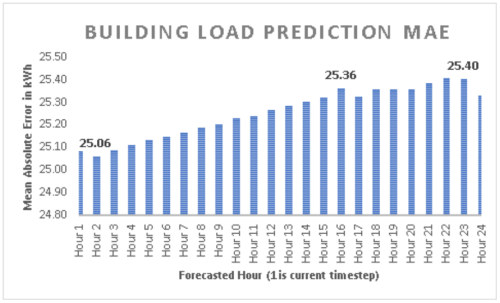


Figure 12: Forecast Error for Each Hour in the Prediction Horizon. MAE is 24.6% of Test Maximum Hourly Energy Demand

2 MCP Performance and Discussion

Here we outline the three major results of this study: (i) the performance of the MCP in general conditions, (ii) the minimum time horizon needed to reach standard operating conditions and (iii) the robustness of the MCP to error.

2.1 Performance of MCP

The performance of the MCP was compared between two main modes of operation: (i) under the assumption of complete clairvoyance and (ii) under the forecast data that was provided using the tools described in sections 1.3.1 through 1.3.3. The results are shown in Figures 13 and 14.

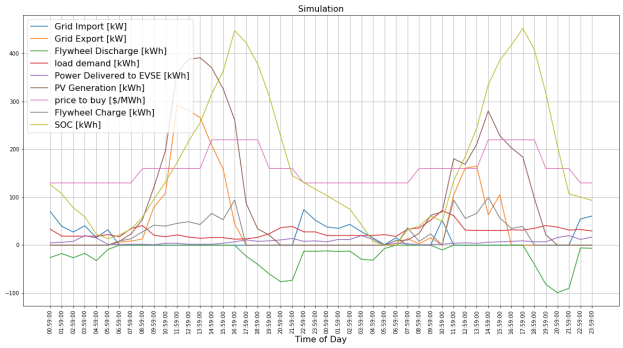


Figure 13: MCP Strategy implemented using a 12-hour time horizon and forecast tools.

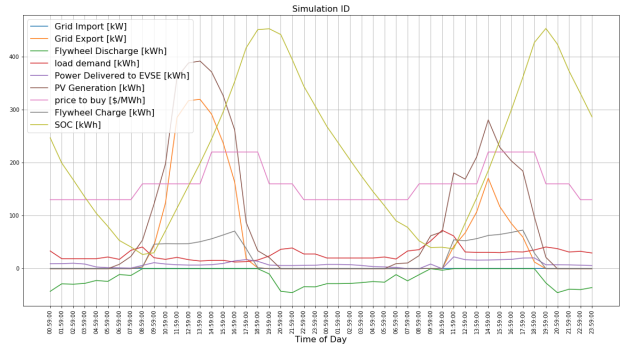


Figure 14: MCP Strategy implemented using 12-hour time horizon assuming full clairvoyance

It is worth mentioning that the optimization program never corrects for errors in the forecast, meaning that the actual values of consumption and generation are not known and the simulation is completely guided by the forecast values. Nonetheless there is an adjustment in the forecast values to the true values of consumption and generation that occurs in the forecast algorithms where the forecast are adjusted each hour to correct for actual conditions.

Figure 13 contains valuable insight into the operations and strategy choice of the MPC. First and foremost, the shape resembles that of the case of perfect clairvoyance in Figure 14. It is apparent that the strategy of the forecast system is similar to that of full clairvoyance - to charge the flywheels during periods of high solar generation while discharging them during periods of low generation and high energy cost, as illustrated by the changes in flywheel state-of-charge in the figure. This concurs with intuition as it is more economically feasible

to perform arbitrage when knowledge of the future is possible. This occurs over both days in Figure 13.

By applying this to a full 14-day period, we were able to obtain 41% of the revenues possible from full clairvoyance. A breakdown of this can be found in the Appendix.

2.2 Selection of Time Horizon

Computing power is not always a luxury, and so one performance metric assessed was the minimum forecast time horizon needed to achieve benchmark performance (at standard conditions). In our study, we varied the time horizon from 2 to 24 hours to see how performance changes. The results were normalized and produced in figure 15.

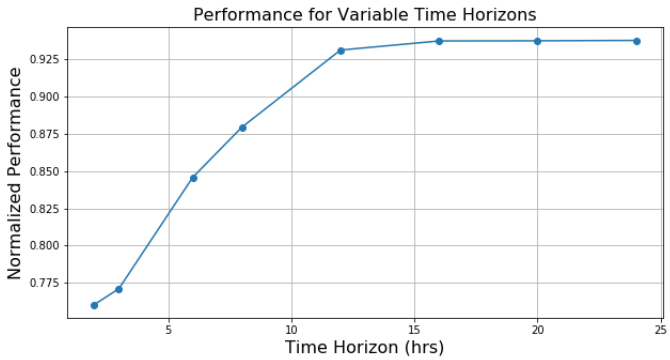


Figure 15: Normalized performance of MCP when varying time horizons used.

Here, normalized performance is computed by comparing earnings via the time horizon as a percentage of maximum theoretical earnings, assuming full clairvoyance. As figure 15 shows, performance decreases as time horizon shortens, but reaches a maximum threshold at around a 12-hour time horizon. Here, the reasoning behind this performance change matches with intuition - as time horizon decreases, the MCP is unable to fully utilize its storage to account for future changes in conditions of its assets. As time horizon increases, the MCP is able to better predict the future (it will know more) and be able to account for changes better. Further assessment of these trends can be found in the Appendix.

2.3 Error Sensitivity Analysis

Finally, error sensitivity is a large part of this MCP as the optimization program used does not account for actual values. As formulated, the program sees no value in keeping charge in the flywheels in case of large errors in the forecast. This is apparent in figure 13, during the second period at time 10am when there is a higher than usual energy demand with a low energy generation. At this point it was necessary to import energy from the grid at a point when energy cost was high. To address the limitation, a penalty term would need to be included in the formulation of the optimization equation, where the term would force the flywheels to possess some charge at every point of the day that would only be used as

backup for unforseen moments of low generation with high energy demand. This could be valuable during the more variable seasons of the year.

3 Conclusions and Findings

Through this study, we have demonstrated that the design and implementation of a simple moving-time-horizon linear optimization program as an MCP can produce revenues for Microgrids with variable generation, energy storage and EV demand. Given data of PV generation, EV demand and building load demand, we have used Markov chains, ARX models and Random Forest Regression to forecast future data points. This has produced results that have reached 41% of maximum theoretical revenues.

The shortcomings of our MCP are aligned with the future work that can be done. The MCP used in our study lacks the ability to correct for high variability in actual energy consumption or production, and also does not account for variability in season. Further research can be done to account for seasonal changes in energy consumption, as well as improve on forecasting algorithms to ensure that actual data is taken into account during the forecasting process.

Executive Summary

Energy consumption is predicted to rise greatly over the next century, and the integration of renewables into the current electricity mix is needed to lower energy-related carbon emissions. Much of this can be done through Microgrids, though a cost-minimizing Microgrid Controls Platform (MCP) is needed. Here, we have designed an MCP that uses a moving time horizon linear optimization program to make decisions on whether a Microgrid should store, sell or buy energy from the grid based on generation and consumption forecasts of the Microgrid's assets. These forecasts are generated using different methods for each asset: Random forest regression for building load forecasting, ARX and machine learning models for EV demand forecasting and Markov chains for PV generation forecasting. Each method was chosen after experimentation with different models and picking the model with the smallest error and deviation from actual data.

When operating with only forecast data, the MCP produces decisions that closely align with the case of full clairvoyance. The strategy chosen matches with the optimization objective chosen - the program typically decides to perform arbitrage in order to gain economic benefit. One major shortcoming is that the program fails to address large deviations from predictions, as there is no current means of re-injecting actual data back into the model. The optimal time horizon that would balance computational power and operational stability was found to be 12 hours, and was used as the best-case operations scenario in our study. Future work should focus on accounting for seasonal variance in energy consumption as well as robustness against errors in forecasting.

References

[1] "Annual energy outlook 2018," U.S. Environmental Information Agency, 2018. [Online]. Available: https://www.eia.gov/outlooks/aeo/

[2] A. H. Fathima and K. Palanisamy, "Optimization in microgrids with hybrid energy systems - a review," *Renewable and Sustainable Energy Reviews*, vol. 45, pp. 431–446, 2015.

[3] "What is the smart grid," U.S. Department of Energy, 2018. [Online]. Available: www.smartgrid.gov/the_smart_grid/smart_grid.html.

[4] C. Sun, F. Sun, and S. Moura, "Data enabled prediction energy management of a pv-battery smart home nanogrid," *American Control Conference*, pp. 1023–1028, 2016.

[5] A. Conejo and L. Morales, J.and Baringo, "Real-time demand response model," *IEEE Transactions on Smart Grid*, vol. 1, pp. 236–242, 2011.

[6] L. Suganthi and A. Samuel, "Energy models for demand forecasting - a review," *Renewable and Sustainable Energy Review*, vol. 16, pp. 1223–1240, 2012.

[7] C. Colson, M. Nehrir, and S. Pourmousavi, "Towards real-time microgrid power management using computational intelligence methods," 2010.

[8] "IEEE 1547 standard for interconnecting distributed resources with electric power systems," IEEE Standards Association, 2018. [Online]. Available: http://grouper.ieee.org/groups/scc21/1547/1547_index.html

[9] R. Banos, F. Manzano-Agugliaro, F. Montoya, A. Alcayde, and J. Gomez, "Optimization methods applied to renewable and sustainable energy: A review," *Renewable and Sustainable Energy Reviews*, vol. 15, pp. 1753–1766, 2011.

[10] S. Schoenung, "Energy storage systems cost update," *SANDIA Report*, 2011.

[11] "Dataport API data set," Pecan Street Inc., 2018. [Online]. Available: http://www.pecanstreet.org/

[12] R. Maghami, H. Hizam, C. Gomes, A. Radzi, I. Rezadad, and S. Hajighorbani, "Power loss due to soiling on solar panel: A review, renewable and sustainable energy reviews," *Renewable and Sustainable Energy Reviews*, vol. 59, pp. 1307–1316, 2016.

[13] S. X. Chen, H. B. Gooi, and M. Q. Wang, "Sizing of energy storage for microgrids," *IEEE Transactions on Smart Grid*, vol. 3, no. 1, 2012.

[14] P. J. Skoplaki, E., "On the temperature dependence of photovoltaic module electrical performance: A review of efficiency/power correlations," *Solar Energy*, vol. 83, pp. 614–624, 2009.

[15] "Utility API data," UtilityAPI, 2018. [Online]. Available: https://utilityapi.com/

[16] G. James, D. Witten, T. Hastie, and R. Tibshirani, *Tree-Based Methods*, G. Casella, F. S., and I. Olkin, Eds. New York, NY: Springer Science and Business Media, 2013.

[17] H. Zhao and F. Magoules, "A review on the prediction of building energy consumption," *Renewable and Sustainable Energy Reviews*, vol. 16, pp. 3588–3592, 2012.

Appendix

Variable Definitions in Optimization Program

The variables and their corresponding definitions are enumerated in the table below.

Variable	Units	Definition
$G_I(i)$, $G_E(i)$	[kW]	Power imported or exported to/from the Grid
$G_{I,max}$, $G_{E,max}$	[kW]	Maximum grid import and export power
$S(i)$	[kW]	Power generated from solar PV
$L(i)$	[kW]	Load demand of the facility
$B_d(i)$, $B_c(i)$	[kW]	Power discharged from/charged to flywheel storage
$E(i)$	[kWh]	Energy level of flywheel storage
$C_{2,1}(i)$	[kW]	Charging power of Level-2 EV charging station
$Z_{2,1}(i)$	[kWh]	Cumulative energy delivered to Level-2 EV charging station
$Z_{2,1,min}(i)$	[kWh]	Minimum energy required for EVs at timestep i
η_2	[-]	Charging efficiency for Level-2 EV charging stations
$C_{2,1,max}$	[kW]	Maximum charging capacity of Level-2 EV charging stations
$c_I(i)$	[$USD/kW]	Time-of-use price of grid-imported power
$r_E(i)$	[$USD/kW]	Time-of-sale revenue of exported power to grid
Δt	[hr]	Time step
η_c, η_d	[-]	Charging and discharging efficiencies for flywheel storage
$E_{meas}(i)$	[kWh]	Measured initial flywheel storage energy level
E_{max}	[kWh]	Maximum energy stored in flywheel storage
B_{max}	[kW]	Maximum charging capacity of flywheel storage system

Building Load Demand Data Table

The following data table corresponds with 9 in section 1.3.3.

Hour	0	1	2	3	4	5	6	7	8	9	10	11
Mean (kWh)	0.237	0.200	0.179	0.167	0.161	0.169	0.211	0.277	0.286	0.267	0.259	0.255
StDev (kWh)	0.364	0.308	0.267	0.240	0.234	0.228	0.285	0.418	0.379	0.378	0.401	0.404

Hour	12	13	14	15	16	17	18	19	20	21	22	23
Mean (kWh)	0.249	0.245	0.239	0.238	0.250	0.286	0.336	0.375	0.411	0.425	0.380	0.294
StDev (kWh)	0.399	0.402	0.370	0.357	0.371	0.389	0.440	0.454	0.490	0.524	0.511	0.434

Figure 16: Table of Mean and Sample Standard Deviation of hourly energy demand.

Pacific Gas and Electric Residential Electricity Rates

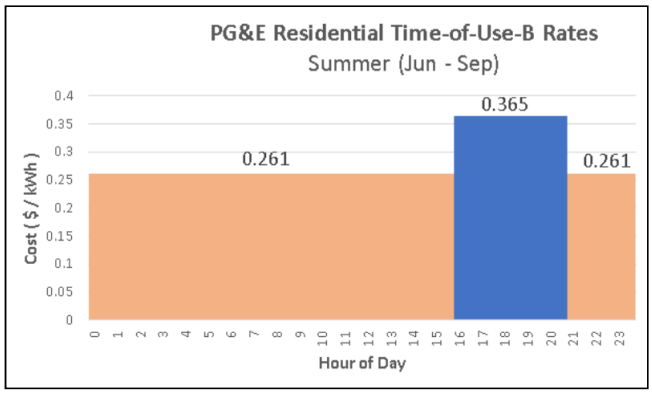


Figure 17: Data for Summer Time-of-Use Residential Electricity Rates from PG&E

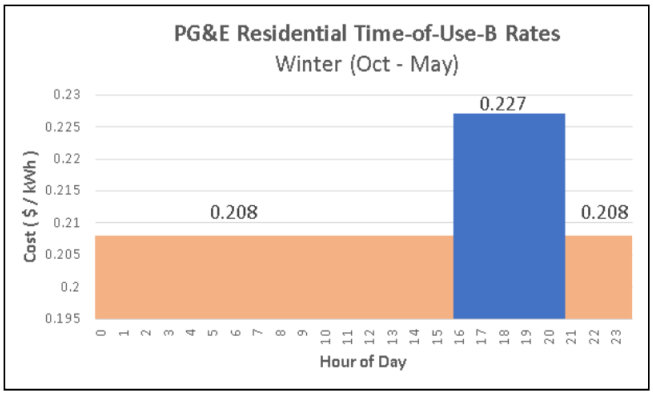


Figure 18: Data for Winter Time-of-Use Residential Electricity Rates from PG&E

Summary of the results obtained from the MCP

The results of the MCP are summarized as follows:

General Description	Value
Cost Without System*	$1,995.00
Cost with PV generation**	-$1,119.09
Maximum Possible Earnings***	-$1,463.30
Performance Values	**Value**
Lower bound of earnings	-$1,119.09
Higher Bound of earnings	-$1,463.30
Achieved Earnings	-$1,260.00
Percentage of possible earnings obtained	41%

**Represents the cost of energy the microgrid would pay under the existing conditions. That is no generation or storage.*
*** Represents the cost of energy if the microgrid had PV generation but no energy storage device.*
**** Represents the earnings possible with energy storage, PV generation and clairvoyance of all demand.*

Figure 19: Summary of the results obtained from the MCP

Further Analysis of Time Horizon Selection Implications

Since shorter time horizons mean that the program is unable to fully predict the future, the MCP will choose to discharge all of the flywheel's charge as soon as possible to make profit. This is seen in Figure 20 as compared to the flywheel state-of-charge response in Figure 21.

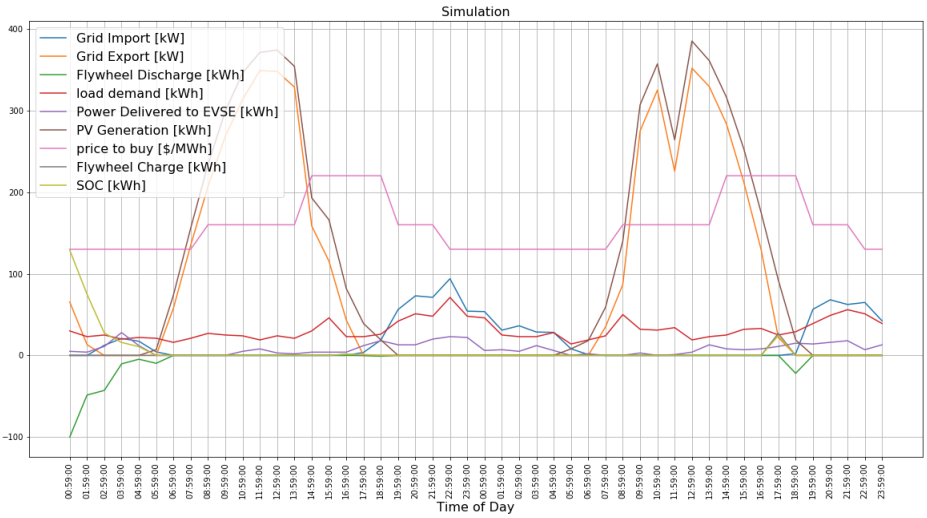


Figure 20: MCP operations simulation with only 3-hour time horizon

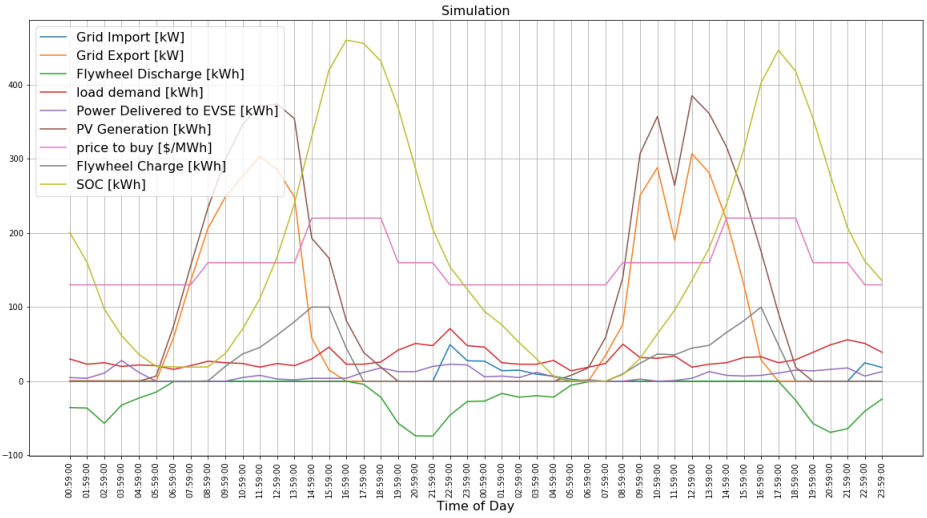


Figure 21: MCP operations simulation with full 12-hour time horizon

Acknowledgments

The team wishes to thank Professor Scott Moura for his guidance and insight throughout the course of this semester and project. The group also wishes to thank Bertrand Travacca for his ideas and discussions regarding alternative approaches for predictive modeling control. Additionally, the team acknowledges the entire CE 295 class for adding value to the overall curriculum of CE 295 through thoughtful and inquisitive questions and comments.

About the authors

Ramon Crespo M.S. Civil and Environmental Engineering: Energy, Civil Infrastructure and Climate.

Ioanna Kavvada M.S. Civil and Environmental Engineering: Energy, Civil Infrastructure and Climate.

Guo Jun Li M.S. Civil and Environmental Engineering: Energy, Civil Infrastructure and Climate.

Dieter Smiley M.S. Civil and Environmental Engineering: Energy, Civil Infrastructure and Climate.

Grid Optimization and Resiliency Study for the Puerto Rican Electrical Grid

Dylan Kato, Anoush Razavian, Shane Gallagher, Dane de Wet

Abstract

In order to meet carbon emission targets in the fight to stop climate change, many energy grids around the world are switching to clean energy sources. These sources help make modern society more sustainable, but come with their own challenges. Wind and solar power suffer from variability. Hydroelectric generators are geographically restricted, and nuclear power plants are inflexible to rapid load changes. This study uses Puerto Rico as a case study for designing a clean energy grid that balances these challenges to meet demand, without compromising on energy prices, reliable power generation, or grid resiliency. First, the minimal cost generating mix was found, while meeting the required physical constraints for a variety of wind energy grid penetrations. This was done by formulating a linear program to minimize total cost, subject to various demand and ramp rate constraints. From this, a generating mix was selected to conduct a grid resiliency study of a basic grid network.

Introduction

Motivation and Background

Climate change has become one of the greatest challenges facing the world today, resulting in an increased frequency of natural disasters and extreme weather events. These natural events are particularly devastating in areas with vulnerable infrastructure, like Puerto Rico. Recently, Hurricane Maria left much of Puerto Rico without power for over four months, crippling daily life and risking the health of millions of people. In order to prevent disasters like this in the future, it is important that power grids be designed for resilience to worsening natural disasters while eliminating the emissions that are causing climate change in the first place.

A large body of scientific work has helped show that greenhouse gas emissions are leading to cascading negative effects that will greatly affect the Earth and the future of people around the globe. Power generation is one of the primary causes of these emissions. Many grids around the world would like to switch to cleaner energy sources, but need to address the challenges associated with them. This project aims to help address that issue by designing a modernized grid for Puerto Rico that will greatly reduce the environmental impact of power generation while making it resilient to natural disasters.

Such resilience comes at a cost, since energy grid management can become cumbersome when trying to balance low cost power production with resiliency during extreme weather conditions. Control schemes need to be carefully thought out so that the grid can balance natural disaster resilience with needless inefficiencies created through over-fortification. Additional challenges arise when designing an energy grid that has a high penetration of renewable energy.

It is vital that we face the challenge of designing clean energy grids while maintaining resilient and economic energy production. This project aims to address the challenges associated with creating a grid that is resilient, economic, and clean.

Focus of this Study

For this study, we found the optimal power generation mix for a redesigned Puerto Rican power grid to reliably meet demand using clean generating sources while minimizing cost. The optimal generating mix was then used for a grid resiliency study to better understand how to better design a modernized grid to withstand natural disasters like Hurricane Maria.

Literature review

Ferris of E&E news reported that the aftermath of Hurricane Maria in a months-long blackout, despite the fact that the damage potential of the storm was similar to Hurricane Sandy, which in comparison, left residents without power for at most several weeks. These devastating impact on Puerto Rico caught the attention of officials, motivating them to find solutions superior to the traditional grid network infrastructure. Most are in agreement that microgrids can provide a reasonably compelling solution. Grid redesign does face some bureaucratic obstacles—even though many companies have lent helping hands to implement renewables, many have concerns about creating a microgrid system in Puerto Rico because of the potential it has to undermine the monopoly the Puerto Rico Electric Power Authority (PREPA) has over utilities. Robust and resilient redesign may require the aid of external factors to become a reality.[1]

De Jonghe et al. in their 2011 paper on grid optimization with high renewable penetration use many techniques that we wish to build upon for determining an optimal and resilient grid design. De Jonghe incorporates renewables as a "negative demand", shifting the variability of renewables from the supply side to the demand side, resulting in a "net demand" to be met by controllable energy sources. Ramp rates from traditional energy sources were then binned into Base, Mid, Peak, and High Peak ramp rates. A minimum amount of flexibility in ramp rate is necessary at any given time due to the variability of high renewable penetration. This constraint can be softened by renewable curtailment, but is done at an opportunity cost of not using renewable power. Additionally, energy storage was also considered for softening this constraint. [2]

Egbue's paper details how achieving resilience can be done through increasing system flexibility and robustness, namely through introducing microgrids that combat the fragility of the interconnected system. Installing microgrids is in essence the same as introducing defensive islanding. Microgrids allow for distributed generation, which allows for power to still be delivered via renewables on the microgrid in the case that power-demanding

infrastructure is cut off from the main source. The paper goes into discussion about the challenges in microgrid operation, control, cybersecurity, and other barriers. It also discusses their utility of microgrids in light of natural disasters, relevant to both the motivation of our project and solution we plan to attempt.[3]

Panteli's paper shows the need for resilient grid infrastructure to combat the damage caused by extreme weather events, namely the aftermath of Hurricane Sandy, motivated this paper. This is highly relevant to the state of Puerto Rico's grid redesign, as the destruction of Hurricane Maria was similar to, though greater in magitude than that of Hurricane Sandy The paper proposes a grid resiliency approach known as defensive islanding, where the objective is to maintain stability of resulting subsystems to reduce total losses across the board. The general method the paper suggests is to begin by modelling the impact of the weather event using a fragility curve, conducting risk assessment as suggested by the procedure outlined in the paper that can be applied to any specific weather event, use the defensive islanding algorithm which uses power flow data to split the system into islands and isolate the vulnerable components and apply the appropriate islanding solution, and finally, use the weather dependent failure probabilities obtained from fragility curves and a uniformly distributed random number to determine which components will trip due to the weather event. The paper applies this method by doing a case study for a simplified transmission network in Great Britain. They use this analysis to conclude that as the "Severity Risk Index" (SRI) they define increases, the more beneficial the application of the defensive islanding becomes.[4]

Key contributions

One aspect all these references had in common was the suggestion of distributed generation, which can be achieved through microgrid operation and control. In our study of resilience, we have thus far conducted testing to see how our simplistic macrogrid will function when branches are no longer functional. Some of these branch breaks effectively causing the system to behave like a microgrid. This is also similar to the defensive islanding strategy suggested by several of the references, as the isolated island operation will be key to our findings. Some of these studies in the literature were conducted prior to the extreme weather events in Puerto Rico, as microgrid operation seems to have been seen as a valuable solution to improve resilience for a while, but in the face of disaster, is finally being taken more seriously than ever before.

The De Jonghe paper contributed largely to the formulation of the optimization problem used to determine the optimum energy mix. We built upon this paper by adjusting it to nuclear, natural gas, and hydro power. It was adjusted to incorporate maximum hydroelectric power, and physical constraints related to the ramp rate of each specific technology. We ran the optimization for a spectrum of different wind penetrations, and calculated the total emissions for the time period analyzed for each. These were then plotted to gain an understanding of the influence wind has on optimal controllable energy mix on the grid.

Technical Description

Generation Mix Optimization

Part I: Problem Formulation

In order to find the optimal mix of generating technologies for the new Puerto Rican grid, we formulated an optimization problem based on the work by De Jonghe [2]. This formulation treats the variable generation by wind as a negative demand to create a net demand (1). The capacity of the controllable generators (natural gas, nuclear, and hydro) are then optimized to minimize the cost of the grid based on the fixed and variable costs of each generation technology (2), (3), and (4). The constraints included a requirement for generation to meet demand (5), as well as physical constraints related to the ramp rates (9 through 14) , must run capacity (7), and periodic maintenance factors (6). The capacity of hydro power is also constrained due to limited availability of hydro power expansion (8). The time based constraints were set for a large dataset of different demand and wind power generation data. In this optimization formulation, wind power was not curtailed, as the data likely already included curtailment by the Finish grid operator.

$$\forall j \in J : NET\_DEMAND_j = DEMAND_j - (WP_j * WCAP) \tag{1}$$

$$\forall i \in I : F_i = INV_i + FOM_i \tag{2}$$

$$\forall i \in I : V_i = FU_i + VOM_i \tag{3}$$

$$\text{minimize} : \sum_i F_i * cap_i + \sum_{ij} V_i * g_{ij} \tag{4}$$

$$\text{subject to} : \forall j \in J : \sum_i g_{ij} = NET\_DEMAND_j \tag{5}$$

$$\forall i \in I, j \in J : g_{ij} \leq cap_i * PM_i \tag{6}$$

$$\forall i \in I, j \in J : g_{ij} \geq MR_i * cap_i \tag{7}$$

$$cap_{HYDRO} \quad \leq MAX_{HYDRO} \tag{8}$$

$$\forall i \in I, j \in J : flex\_up_{ij} \leq RAMP\_C_i * g_{ij-1} + RAMP\_NC_i * (cap_i - g_{ij}) \tag{9}$$

$$\forall i \in I, j \in J : flex\_up_{ij} \leq cap_i - g_{ij-1} \tag{10}$$

$$\forall i \in I, j \in J : flex\_down_{ij} \leq RAMP\_C_i * g_{ij-1} + RAMP\_NC_i * (cap_i - g_{ij-1}) \tag{11}$$

$$\forall i \in I, j \in J : flex\_down_{ij} \leq g_{ij-1} \tag{12}$$

$$\forall i \in I, j \in J : g_{ij} \leq g_{ij-1} + flex\_up_{ij} \tag{13}$$

$$\forall i \in I, j \in J : g_{ij} \geq g_{ij-1} - flex\_down_{ij} \tag{14}$$

Part II: Calculating Parameters

Data for power generation in Puerto Rico was unavailable for this study, so we used data which came from Finland in the early part of 2017 [5]. We normalized the demand data to the maximum value for the whole dataset, and scaled it to the size of the Puerto Rican power grid. The wind power data was also normalized to its maximum value, and then scaled for the desired grid penetration. In order to get accurate numbers, data was obtained from a large number of sources [6].

Table 1 outlines the data found to determine parameter values used in the grid optimization. There are several things to note about the parameters used. For the proposed grid, it is desired to use advanced nuclear reactor designs as opposed to conventional reactor designs. The costs for these are not readily available, so several adjustments had to be made. The investment cost for nuclear power was set to the twice the value of the claimed price by General Electric, giving room for first-of-a-kind construction problems [7]. The ramp rates came from a integrated resource plan for CAISO, using the values they use for long term grid planning, as it was desired to minimize emissions while still meeting peak demand. Base load natural gas would likely have a lower variable operation and maintenance cost, although the investment, fixed O&M, and fuel costs are the same. The nuclear power was required to ramp at no more than twenty percent of maximum power per hour. It also has a must run capacity of eighty percent. This was chosen based on the values from CAISO, with some flexibility added to reflect more modern reactor design capabilities [8]. Hydroelectric power capacity was constrained to five times the current installed hydro capacity in Puerto Rico, and given a must run capacity of 13 percent to reflect requirements for downstream water supply. Solar power was not included, as the data from Finland reflected very strong seasonal solar variations, due to the longitude of Finland, that would not be reflective of Puerto Rico.

Table 1: Parameters values used in grid optimization

Type	**Invest**	**Fixed O&M**	**Var. O&M**	**Fuel**	**Must Run**	**Ramp**
Units	(k$/MW)	(k$/MW)	($/MWh)	($/MWh)	% P_{max}	$\frac{\%P_{max}}{hour}$
Nuclear	4000 [7]	84.5 [9]	0.6 [9]	8.4 [10]	80 % [8]	20% [8]
Nat.Gas	1200 [9]	14 [11]	6.8 [11]	11.825[11]	0 % [8]	330% [8]
Hydro	3000 [9]	29.23 [9]	3.16 [9]	0 [9]	13 % [8]	100% [8]

Part III: Grid Optimization Results

Using this optimization formulation previously described, we found the optimal mix of controllable generators, specifically nuclear, hydro, and natural gas for a range of different wind penetrations. We ran this for a wind penetration from zero to sixty percent over a data set of 10 days See Figures 1, 2, 3.

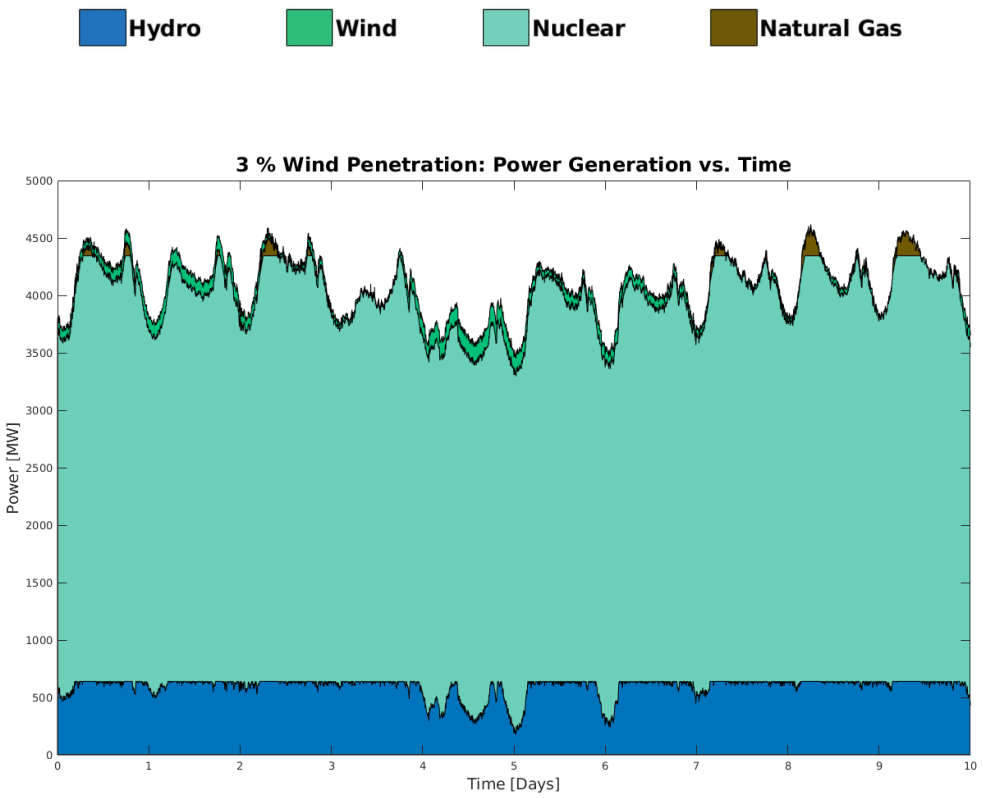


Figure 1: Effect of low penetration of Wind generation on the overall generation for 10 day period

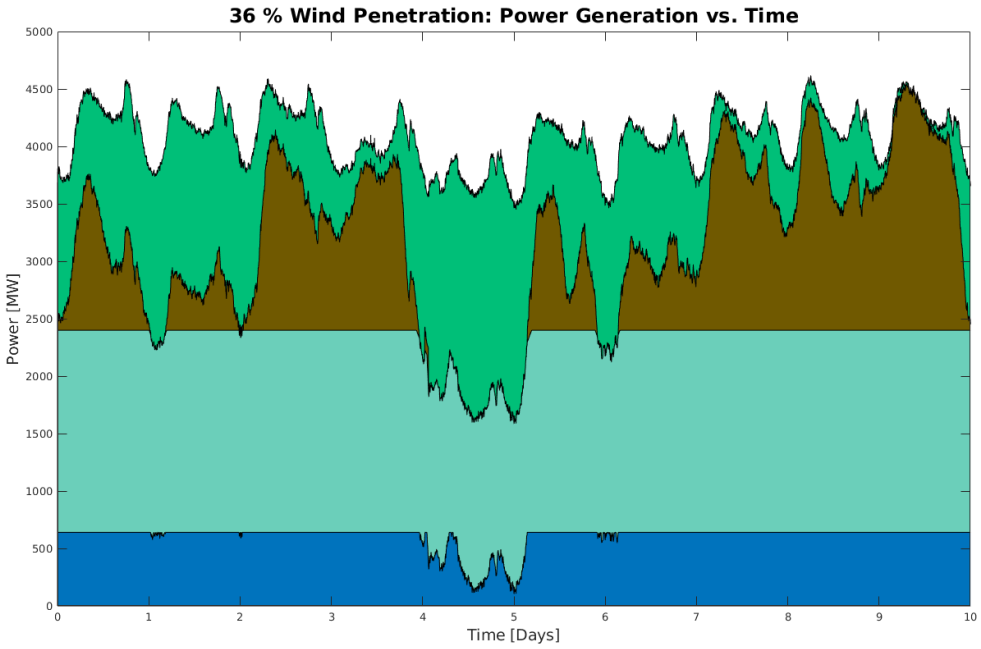


Figure 2: Effect of moderate penetration of Wind generation on the overall generation for 10 day period

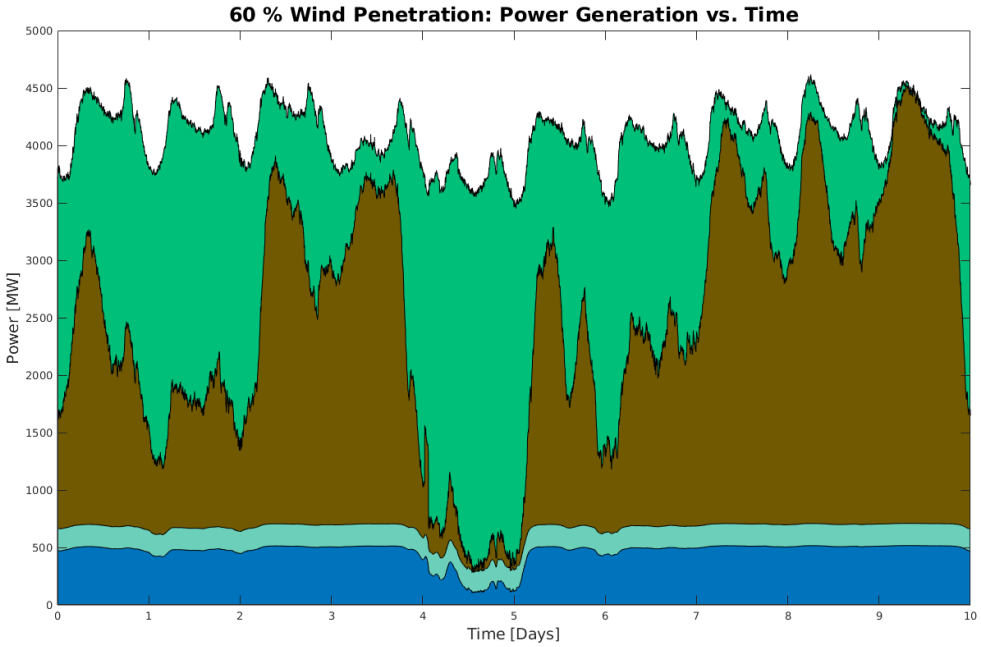


Figure 3: Effect of high penetration of Wind generation on the overall generation for 10 day period

There are several important conclusions from the running the optimization problem for a variety of data sets with a variety of parameters. The first is that the optimal result depends strongly on the length of the data set. Short time periods usually don't feature the bounding ramp rates that long time periods have. Very large ramp rate requirements for a short time period have a big impact on the final result. This shows why it may be beneficial to have distributed energy storage to reduce variability. Initially, the optimal result was a very large amount of hydro with little of anything else. This was due to the low costs with high flexibility to allow hydro to load follow. A maximum capacity was added, as there are large geographic restrictions on the amount of hydro that can be installed. There were also some interesting trends to note with varying wind penetration. As shown in Figure 4, as wind penetration increases, the required natural gas also increased to allow for more flexibility in the grid. With increased wind penetration, more flexible generation is required and therefore nuclear is not suitable for a grid with high penetrations of wind power due to its must run and ramp rate constraints. It is also evident that with increased wind generation, CO_2 emissions actually increase substantially due to the increased reliance on natural gas power plants. This was without wind curtailment, importing or exporting power, distributed storage, or a significant penetration of wind generation. Further work will likely address some of these challenges.

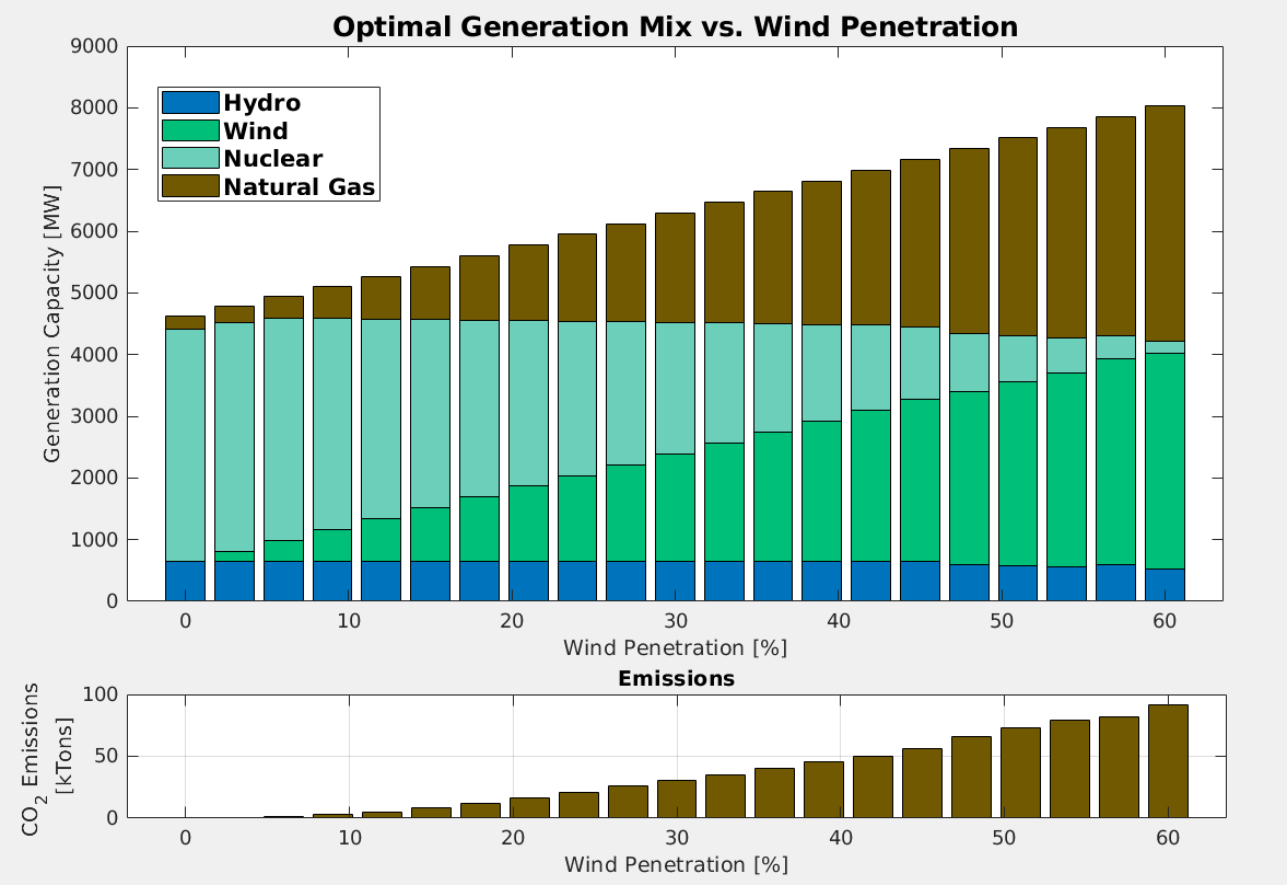


Figure 4: Optimal Generating Mix for Various Wind Penetrations, with Corresponding Emissions

These results were used to determine the energy mix to be used for the resilience study discussed later in this paper. A wind penetration of 30 percent was chosen for the grid resiliency study. This keeps emissions relatively low while incorporating a more diverse grid.

Resiliency Study

Part I: Obtaining Grid Network Model

Our base constraints of the report are borrowed from "HW3: Optimal Economic Dispatch in Distribution Feeders with Renewables." These constraints deal with the power in and power out of nodes, as well as voltage drops and minimum voltage. Our study focuses on adapting this framework to optimize for resilience in the face of natural disasters. This involves changing the objective function to account for total power delivered rather than cost.

Part II: Recasting Objective Function for Resilience

Our objective function measures our desire to meet load demand without incentivising meeting negative demand and is given by:

$$f(x) = \sum_i \max(l^P_{demanded} - l^P_{i,consumed}, 0) + \max(l^Q_{demanded} - l^Q_{i,consumed}, 0) \tag{15}$$

Using an epigraphic reformulation, in order to recast the objective function which contains a "max" into a familiar form, we introduce intermediary variables t^P_j and t^Q_j which represent the "Demand Gap" or the difference in power demanded and power delivered. The new objective function and constraints incorporating these variables are:

$$f(x) = \sum_{j=0}^{12} t^Q_j + t^P_j \tag{16}$$

$$t^Q_j \geq 0, t^P_j \geq 0 \quad \forall \quad j \in Nodes \tag{17}$$

$$t^Q_j \geq l^Q_{j,demanded} - l^Q_{j,consumed} \quad \forall \quad j \in Nodes \tag{18}$$

$$t^P_j \geq l^P_{j,demanded} - l^P_{j,consumed} \quad \forall \quad j \in Nodes \tag{19}$$

Next, we substitute $l^P_{j,consumed}$ and $l^Q_{j,consumed}$ for l^P_j and l^Q_j from the optimization in Homework 3 to get new power constraints:

$$P_{ij} = (l^P_{i,consumed} - p_i) + r_{ij}L_{ij} + \sum_{k \in Nodes} A_{ij}P_{ij} \quad \forall \quad j \in Nodes \quad i = \rho(j) \tag{20}$$

$$Q_{ij} = (l^Q_{i,consumed} - q_i) + r_{ij}L_{ij} + \sum_{k \in Nodes} A_{ij}Q_{ij} \quad \forall \quad j \in Nodes \quad i = \rho(j) \tag{21}$$

Part III: Adding Constraints for Node Disconnections

Here, we incorporated the necessary constraints to account for power line failure. We do so by detaching node j from its parent node i so that the detached node now has one unit voltage (allowing for separate networks), and no power or current flows between the node and its parent. The constraints are as follows:

$$P_{ij} = 0, L_{ij} = 0 \quad \forall \quad j = j_{detached} \quad and \quad i = \rho(j) \tag{22}$$

Part IV: Accounting for Renewable Uncertainty

In order to account for variability in solar and wind generation, we added constraints to make our optimization robust to the stochasticity inherent in each renewable generator:

$$\bar{a} * [s_i, \sigma_{i,1}, \sigma_{i,2}] + \|E * [s_i, \sigma_{i,1}, \sigma_{i,2}]\|_2 \leq 0 \tag{23}$$

$$\sigma_{i,1}, \sigma_{i,2} \in [0, 1] \tag{24}$$

Here, $\bar{a}$ and E contain information on the mean and covariance of the power generation. $\sigma_{i,1}$ and $\sigma_{i,2}$ are decision variables that are the portion of energy used from individual solar panels. Figure 5 shows the change in overall power delivery after including the robustness constraint and then increasing the variance.

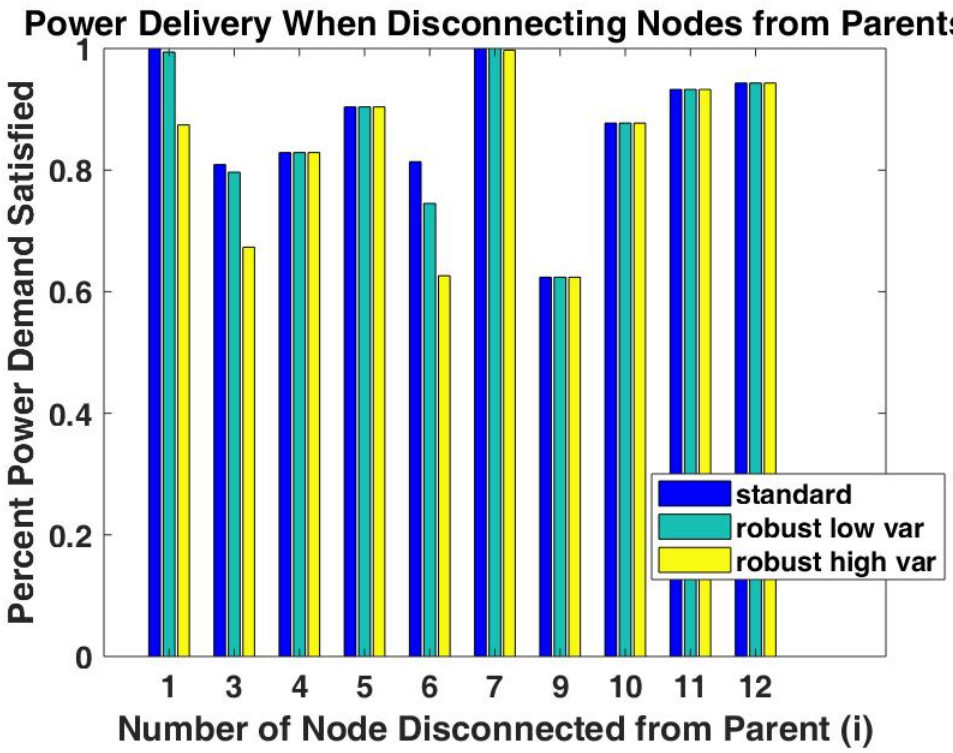


Figure 5: Comparing Power Delivered for Different Renewable Generation Variances

Part V: Prioritizing Nodal Power Delivery

We considered that it is more important to deliver power to certain nodes. For example, if a particular node were a grocery store, hospital, or other priority energy location, that node should take precedence over nodes that contain, for instance, recreational centers (sorry, Disneyland). We were able to account for this by adjusting our objective function to include the following "priority" term which weighs the Demand Gap by coefficients that are set by us to be larger if a node's power delivery is more important:

$$\text{maximize:} \quad \sum_{j=0}^{12} Priority_j * (t_j^Q + t_j^P) \tag{25}$$

Part VI: Final Integration and Results

Finally, after integrating all of our new constraints into the optimization formulation we obtained the following optimization problem:

$$\text{maximize:} \sum_{j=0}^{12} Priority_j * (t_j^P + t_j^Q) \tag{26}$$

$$\text{subject to:} \quad t_j^Q \geq 0, t_j^P \geq 0 \quad \forall \quad j \in Nodes \tag{27}$$

$$t_j^Q > l_{j,demanded}^Q - l_{j,consumed}^Q \quad \forall \quad j \in Nodes \tag{28}$$

$$t_j^P > l_{j,demanded}^P - l_{j,consumed}^P \quad \forall \quad j \in Nodes \tag{29}$$

$$P_{ij} = (l_{i,consumed}^P - p_i) + r_{ij} L_{ij} + \sum_{k \in Nodes} A_{ij} P_{ij} \quad \forall \quad j \in Nodes \quad i = \rho(j) \tag{30}$$

$$Q_{ij} = (l_{i,consumed}^Q - q_i) + r_{ij} L_{ij} + \sum_{k \in Nodes} A_{ij} Q_{ij} \quad \forall \quad j \in Nodes \quad i = \rho(j) \tag{31}$$

$$p_j \geq 0, q_j \geq 0, l_{i,demanded}^P \geq 0, l_{i,demanded}^Q \geq 0 \quad \forall \quad j \in Nodes \tag{32}$$

$$\|p_j, q_j\|_2 = s_j \quad \forall \quad j \in Nodes \tag{33}$$

$$s_j \leq s_{j,max} \tag{34}$$

$$v_{min}^2 \leq V_j \leq v_{max}^2 \tag{35}$$

$$L_{ij} \leq I_{ij,max}^2 \tag{36}$$

$$\bar{a} * [s_i, \sigma_{i,1}, \sigma_{i,2}] + \|E * [s_i, \sigma_{i,1}, \sigma_{i,2}]\|_2 \leq 0 \tag{37}$$

$$\sigma_{i,1}, \sigma_{i,2} \in [0, 1] \tag{38}$$

$$P_{ij} = 0, L_{ij} = 0 \quad \forall \quad j = j_{detached} \quad and \quad i = \rho(j) \tag{39}$$

After implementing in Matlab, the total percentage of power delivered was calculated as the geometric mean of the fraction of active and reactive power delivered. We calculated these percentages using the following equation:

$$FractionDelivered = 1 - \frac{\sum_{i=nodes} Desired - Delivered}{\sum_{i=nodes} Delivered} \tag{40}$$

The following figure shows visually how our new node network could look different from the network in Homework 3. We changed the energy sources to also include nuclear, wind, and hydro and their respective optimal generation determined from the Generation Mix Optimization. We also treated Node 12 like a hospital by giving it priority in the objective function.

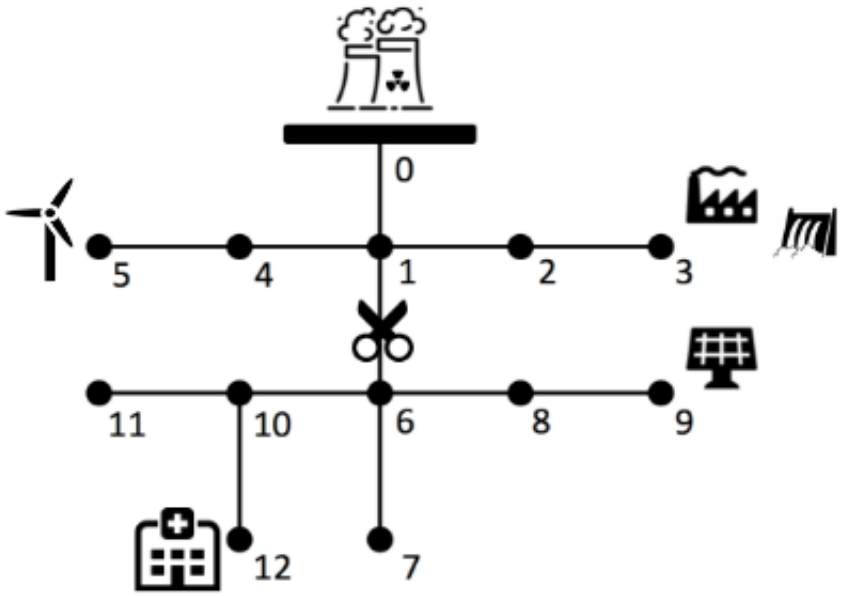


Figure 6: Homework 3 Network Adjusted

We wanted to see how much power would be able to be supplied to the entire network as a result of cutting off each node from its parent node. We used the *FractionDelivered* metric to be able to easily visualize this. As you can see, the worst case scenarios (scenarios where

the network is far from providing adequate power to the entire network) involve breaking off Node 6, Node 9, and Node 10 from their respective parent node.

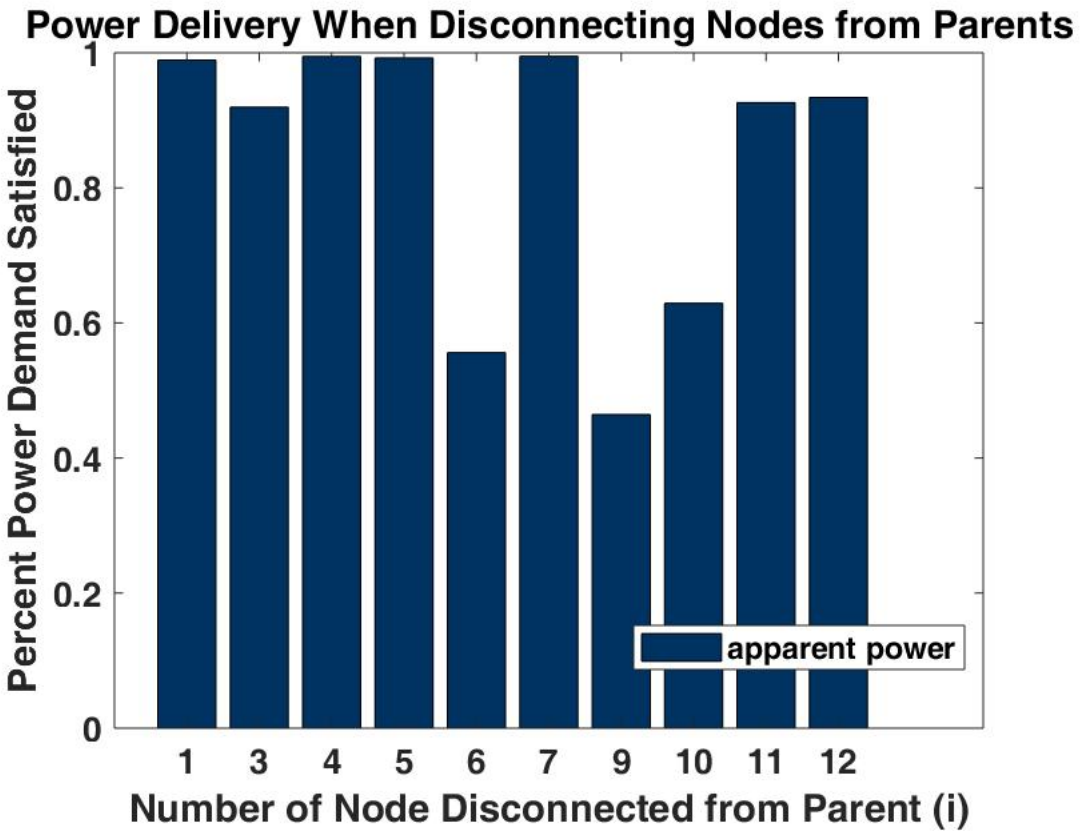


Figure 7: Comparing Power Delivered

We then considered what would happen in the disaster scenario where the line between Node 6 and its parent node became disconnected. The following figure depicts what the corresponding satisfied power demand would be at each node. As you can see, the entirety of Node 12's power demand is satisfied since it was prioritized in the objective function and the other connected nodes in the subnetwork would suffer. In order to prevent this disaster scenario, the redesign would involve adjusting generator placement in such a way that one renewable generator would not have to service 7 nodes, even in the worst case.

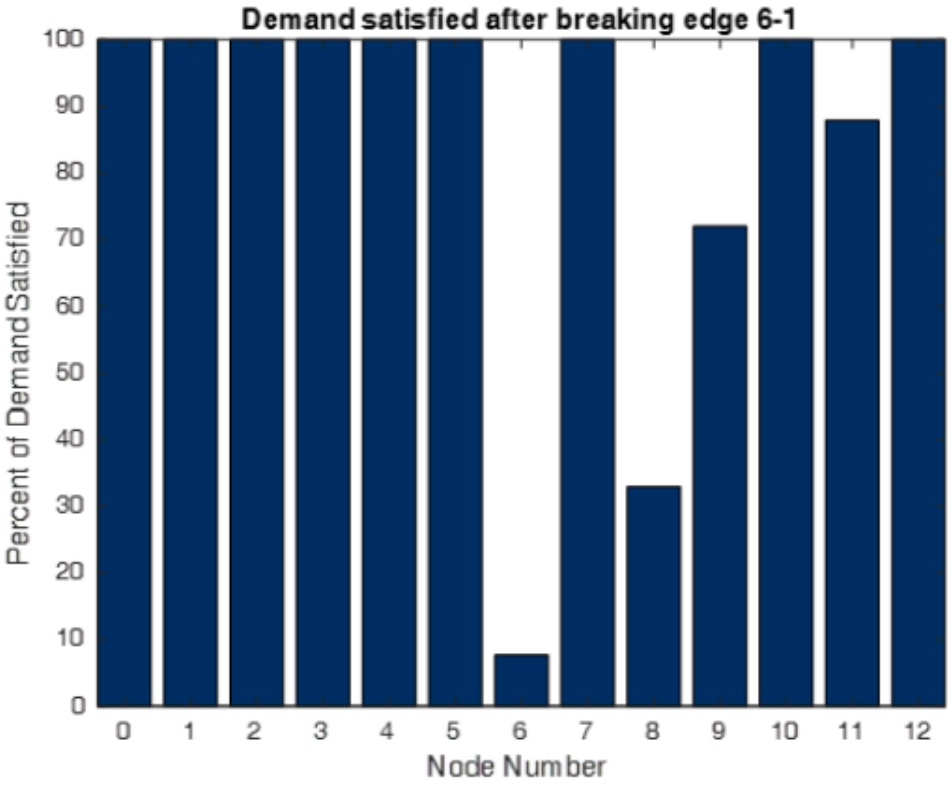


Figure 8: Comparing Power Delivered

Discussion

This kind of work is a step toward envisioning energy grids that are sustainable and also resilient. Through this work, we have developed a methodology that one can use to focus on the important factors of grid resiliency. Puerto Rico can use a model like ours to learn more about their grid and its shortcomings, utilizing this information to defend against insufficient power delivery. This information is crucial to grid redesign.

Power grids are complicated systems, and clean energy generation is inherently limited by variability and availability. One of the best ways to solve this problem is through systematically determining the best implementation of power distribution. With limited resources in system settings, it is essential to use optimization to make the best use of the resources we have.

Through our work, we created a framework for assessing resiliency of a network, but haven't answered several key questions. For one, we only used our framework for one network layout. This could be expanded by applying the resiliency study to several network layouts to find the optimal network layout for resiliency. This future work would also involve developing a method/metric for comparing networks in their performance under disasters. Future teams might also think about incorporating constraints for failure of grid components as a result of

increased power sent to some nodes. As it stands, we allow more power to be delivered than is demanded which could pose a problem if such power surges lead to component failure. Another future work could be incorporating the monetary value of resiliency and solving the original minimum-cost-optimization but with a multi objective function that accounts for generation cost and the cost of grid vulnerability. A final addition that may be interesting is looking at grid resiliency after incorporating energy storage. This would open a new problem of incorporating energy storage that would require the use of optimal control.

Summary

In this study, an optimal energy generation mix for Puerto Rico was determined by minimizing the cost of generation given a variety of physical constraints such as ramp rates and various levels of penetration of intermittent renewable energy sources. The results showed that increasing the amount of wind energy penetration increases the required amount of natural gas peaker power plants needed to meet demand, and therefore increases greenhouse gas emissions. The need for peaker plants is due to the slow ramp rates of base load nuclear power A generation mix was selected to conduct a grid resiliency study of a grid network. Natural disasters were simulated by severing transmission between grid network nodes. The results of resiliency study show that the implementation of smart grid control logic such as prioritizing certain nodes (such as hospitals) in an emergency scenario can be beneficial.

References

[1] D. Ferris and P. Behr, "Microgrids could save puerto rico. but first, a fight." 2017.

[2] C. De Jonghe, "Determining optimal electricity technology mix," *Applied Energy*, vol. 88, pp. 2231–2238, 2011.

[3] D. N. O. Egbue and P. Peterson, "2016 international conference on smart grid and clean energy technologies," 2016.

[4] P. M. M. Panteli, D.N. Trakas and N. Hatziargyriou, "Boosting the power grid resilience to extreme weather events using defensive islanding," *IEEE Transactions on Smart Grid*, vol. 7, no. 6, pp. 2913–2922, Nov. 2016.

[5] Fingrid. [Online]. Available: https://data.fingrid.fi/open-data-forms/search/en/?variable_id=75

[6] O. P. S. D. sources. Wind and solar power time series. [Online]. Available: https://open-power-system-data.org/data-sources#8_Wind_and_solar_power_time_series

[7] C. Surran, "Ge hitachi touts development of smaller, cheaper nuclear reactor," *Seeking Alpha*, Apr. 20, 2018.

[8] Caiso integrated resource plan 2017: Inputs and assumptions.

[9] Transparent cost database. [Online]. Available: https://openei.org/apps/TCDB/#blank

[10] Average power plant operating expenses for major u.s. investor-owned electric utilities, 2006 through 2016. [Online]. Available: https://www.eia.gov/electricity/annual/html/epa_08_04.html

[11] E. Commission, "Gcost estimates for thermal peaking plant," June 2008.

Acknowledgments

Thank you to Dr. Scott Moura for his guidance and mentorship throughout this project.

About the authors

Dane de Wet and Shane Gallagher: Two of the group members, Dane de Wet and Shane Gallagher, are Ph.D. students in the Department of Nuclear Engineering. Their work involves studying thermal hydraulics in advanced nuclear reactors, which address most of the concerns associated with old nuclear power generation. Advanced reactors are being designed for passive safety, adaptability, and economic power generation. Their research group has been designing a reactor called the fluoride salt-cooled high temperature reactor (FHR). This type of reactor operates at much higher temperatures, and therefore higher efficiencies, while being able to operate at near atmospheric pressures in the primary loop, providing enhanced safety margins. For them, the project will further the understanding of the role advanced nuclear power can play in providing a resilient, economic, and clean energy grid.

Dylan Kato and Anoush Razavian: The other two students are undergraduates in the CEE department and are primarily interested in making smarter energy systems. Dylan has taken 2 previous courses in optimization and another in controls and is excited to apply his previous knowledge as well as the knowledge gained in this course to tackle a real-world problem. Dylan plans to start his Ph.D. in the fall and hopes the knowledge gained from this project will strengthen his ability to model infrastructural systems.

Anoush is passionate about clean energy and smart city development. She hopes that by taking this course, she can learn valuable skills that she can apply to have an influential career in fields like clean-tech, microgrid management, and civil systems engineering. Anoush will be working in Kimley-Horn's Intelligent Transportation Systems group this fall, viewing it as a step toward being a front-runner in creating smarter and more sustainable infrastructure. She hopes to ultimately obtain a Masters in Civil Systems. Anoush has also taken one previous course in optimization and another in cyber-physical systems.

Together, Anoush and Dylan built a Cyber-Physical system for residential-scale energy monitoring and control that collects usage at the breaker level and actuates at the outlet level.

Robust Optimal Sizing and Operation of a Microgrid with Electric Vehicle Charging and Renewable Energy Generation

Armando A. Domingos, Marc R. Hutton, Aaron Jagtianey, and Soomin Woo

Abstract

Electric vehicles (EVs) are increasing in popularity and access to charging infrastructure will play an important role in their widespread adoption. Additional investment in existing commercial facilities will be required and pairing vehicle charging infrastructure with microgrids (MGs) may help to mitigate energy consumption requirements from EVs. This study developed a model for the economic optimization of a grid-connected commercial MG that includes renewable energy generation, on-site energy storage, and EV charging with Level 2 and 3 capabilities. A robust optimization method that considers the stochastic nature of energy demanded and supplied was developed while satisfying constraints imposed by the facility. Results from a Mixed-Integer Linear Program (MILP) and Mixed-Integer Second-Order Cone Program (MISOCP) were evaluated to determine primary drivers for infrastructure sizing and to evaluate the robustness of the optimization model for various uncertainty levels of input data. The MISOCP was able to identify infeasible scenarios and was able to plan the optimal scenario while being robust to all stochastic variables, making it the preferred choice for planning purposes over the MILP.

1 Introduction

1.1 Motivation and Background

Electric Vehicles and Microgrids

With new electric vehicle (EV) registrations reaching record numbers in recent years, the EV market is expected to reach forty to seventy million vehicles by 2025 [1]. To meet increasing demand, additional charging infrastructure will be required in most commercial facilities, where existing charging infrastructure is often limited to a small proportion of overall parking. Increasing the number of charging stations and accommodating a large number of EVs charging simultaneously could also multiply the energy load requirements of commercial facilities, which in turn could threaten the stability of local electrical infrastructure and require expensive upgrades to local and grid level equipment [2].

Additionally, due to long charging times for current charging technology, service reliability is crucial to alleviate EV owner range anxiety and to provide adequate charge for commuting needs. Methods to optimize service will be critical to satisfy EV owners, who will expect charging access at all times, even during periods of peak demand. While charging demand could be met by deploying additional charging stations, minimizing system cost will be an important consideration for facility owners when determining the number of chargers to install.

To meet the increased energy demand from EV charging, facility managers could face additional cost for upgrades to grid energy import infrastructure, or expensive tariffs with peak demand charges or time-of-use (TOU) rates. Installing distributed generation (DG) in the form of a microgrid (MG) that includes renewable energy generation (REG) and storage allows for facility managers to meet increased demand locally. So called "smart-grid" infrastructure has utilized local MGs to meet increased demand from a variety of sources, including EVs, and to reduce peak system demand [3]. In addition to reducing peak demand, MG components such as REG and energy storage systems (ESS) allow for additional revenue streams for facility managers when combined with net energy metering (NEM) schemes or community choice aggregation (CCA) arrangements.

Therefore, minimizing the cost of charging infrastructure upgrades and operational energy demand while meeting high level of service is critical to commercial facility managers. Combining traditional MG resources such as REG and ESS with EV charging provides synergistic opportunities that could mitigate operational expenses from energy demand and reduce costs for facility managers. Introducing revenue streams from REG export to the grid and EV charging can help offset initial investments and promote additional EV charging facilities.

Stochastic Elements

To determine optimal sizing, the stochastic nature of system inputs must be considered. Renewable energy is stochastic in monthly, diurnal, hourly (and shorter) timescales, due to seasonal changes and local weather, but reliable on long time scales when stochastic elements are aggregated into long-term trends. Similarly, building energy consumption is also reliable on long time scales while maintaining stochasticity on shorter timescales, primarily due to lighting and heating, ventilation and air-conditioning (HVAC) loads that can change due to weather and individual preferences [4].

EV charging demand enjoins energy consumption in these traits. Because of individual differences between driver schedule and travel distance, battery state-of-charge (SOC), and therefore overall charging times exhibit stochastic elements on shorter timescales while maintaining reliability in the long-term [5]. Unlike grid connected building energy loads, additional energy is not easily imported into the vehicle once the EV has left the charging point, increasing the importance of high-level reliability in charging service.

Project Goal

Our project aim is to develop a robust optimization model of a mixed-integer second-order cone program (MISOCP) from a white paper that integrates a MG and EV charging [6]. The model should incorporate stochastic REG, stochastic building energy demand, and stochas-

tic EV charging demand while minimizing capital and operational cost and determining the optimal number of chargers required to meet charging demand. Linear and non-linear methods of optimization should be explored, and both feasible and infeasible solutions should be determined for optimization. Robustness should be the defining model feature, and boundary cases and corner conditions should be explored to resolve any outstanding concerns in this area.

1.2 Literature Review

Recently, there has been significant interest in optimal planning for integrated EV charging systems. The most comprehensive systems include EV charging stations, non-EV site loads, vehicle-to-grid (V2G) connections, REG such as photovoltaic (PV) panels, and ESS. For example, a smart home equipped with an EV charger, smart appliances, DG, and ESS was modeled using a mixed-integer linear program (MILP) with the objective of minimizing operational cost under different demand response (DR) strategies [7]. The smart home system was reevaluated with respect to optimal PV and ESS sizing in [8], using a MILP with the objective function of minimizing both capital costs and operational costs. A smart home with bi-directional EV and ESS was considered in [9], using a MILP to minimize the cost of electricity.

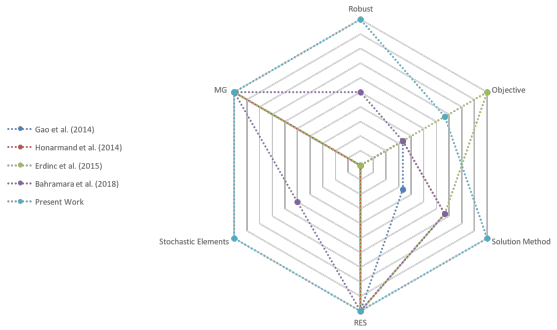


Figure 1: Spider plot of weighted values in several categories for a typical sample of the reviewed literature. The primary method that has been used to minimize cost for similar systems is a mixed integer linear program with a combination of renewable energy, storage, and vehicle charging, but none also included the minimization of capital and operational expenses. Furthermore, none considered robustness as a model feature

The most common method for addressing optimal planning for integrated EV charging systems is to utilize MILP in varying scenarios, from minimizing capital expenses or operational expenses. Optimal operation is explored in [10], using a robust two-stage stochastic problem which accounts for solution uncertainty and model uncertainty. The problem is formulated as a MILP with the objective of minimizing operational cost given stochastic power trading with the grid, stochastic EV charging demand, but does not consider stochastic REG.

While incorporating stochastic elements into optimization problems is well understood, few studies also incorporate robustness as a model feature, accounting for all stochastic variables present in a MG with EV charging. As seen in Figure 1, we see that robustness as a model feature is not well represented in existing literature, even as renewable energy and EV charging plays a prominent role. It is likely that this is due to the difficulty of solving SOCP optimization problems, which are a method for addressing robustness with stochastic variables.

1.3 Focus of this Study

This study aims to develop a robust optimization model from a MISOCP to minimize capital and operational expenses of a microgrid incorporating building load renewable energy, energy storage, and EV charging demand while accounting for the stochastic variables associated with these components. Robustness across all stochastic variables is considered the defining model feature.

2 Technical Description

2.1 Optimization Model

The model is formulated to optimize the sizing and energy management of a commercial facility with REG, ESS, and EV chargers. The model contains two components: the first contains the capital costs for REG, energy storage, and EV chargers. The second contains operating costs for how the facility allocates the energy supply throughout the MG network.

Constraints for the model include physical system parameters such as grid import and export limits, energy storage dynamics, EV charging dynamics, renewable energy scale limits, battery scale limits, and the number of EV chargers.

Mixed Integer Linear Program

To start, a deterministic optimization model was formulated to verify the objective function variables, parameters, and constraints. After formulating the SOCP model, changes were first tested with the deterministic model, allowing for increased flexibility due relatively fast computational times. The MILP was also used in comparison to the robust optimization model to determine how chance constraints influenced optimal sizing.

Objective Function

Our objective equation is to minimize capital cost, energy cost, demand charge, and export revenue as follows:

$$
\begin{aligned}
&\min c_b \cdot b + c_s \cdot s + mc_2 \cdot n_2 + mc_3 \cdot n_3 && \text{(capital cost)} \\
&+ \sum_{k=0}^{N} c_I(k) \cdot \Delta t \cdot [\bar{L}(k) + B_c(k) + C_2(k) + C_3(k) + G_E(k) && \text{(energy cost)} \\
&- s \cdot \bar{S}(k) + S_c(k) - B_d(k)] \\
&+ c_d G_d && \text{(demand charge)} \\
&- \sum_{k=0}^{N} \Delta t \cdot [r_E(k) \cdot G_E(k) + r_2(k) \cdot C_2(k) + r_3(k) \cdot C_3(k)] && \text{(export revenue)} \qquad (1)
\end{aligned}
$$

Constraints

Subject to these constraints:

$$0 \leq S_c(k) \quad \text{(PV curtailment)} \quad (2)$$

$$E(k+1) = E(k) + [\eta_c \cdot B_c(k) - \frac{1}{\eta_d} \cdot B_d(k)] \cdot \Delta t \quad \text{(ESS dynamics)} \quad (3)$$

$$E(0) = b \cdot E_{max} \cdot 0.10 \quad \text{(ESS initial condition)} \quad (4)$$

$$E(24) \geq b \cdot E_{max} \cdot 0.10 \quad \text{(ESS final condition)} \quad (5)$$

$$C_2(24) \leq F^{-1}_{zmin,2(24)} \cdot (\alpha_z^{'}) \quad \text{(level 2 charger limit)} \quad (6)$$

$$C_3(24) \leq F^{-1}_{zmin,3(24)} \cdot (\alpha_z^{'}) \quad \text{(level 3 charger limit)} \quad (7)$$

$$0 \leq E(k) \leq b \cdot E_{max} \quad \text{(ESS energy limits)} \quad (8)$$

$$0 \leq B_c(k) \leq b \cdot B_{max} \quad \text{(ESS power limits)} \quad (9)$$

$$0 \leq B_d(k) \leq b \cdot B_{max} \quad \text{(ESS power limits)} \quad (10)$$

$$Z_2(k+1) = Z_2(k) + [\eta \cdot C_2(k)]\Delta t \quad \text{(delivered EV energy dynamics)} \quad (11)$$

$$Z_3(k+1) = Z_3(k) + [\eta \cdot C_3(k)]\Delta t \quad \text{(delivered EV energy dynamics)} \quad (12)$$

$$Z_2(0) = 0 \quad \text{(initial EV energy)} \quad (13)$$

$$Z_3(0) = 0 \quad \text{(initial EV energy)} \quad (14)$$

$$Z_2(k) \geq F^{-1}_{zmin,2(k)} \cdot (\alpha_z) \quad \text{(EV energy lower bound} \quad (15)$$

$$Z_3(k) \geq F^{-1}_{zmin,3(k)} \cdot (\alpha_z) \quad \text{(EV mobility constraint)} \quad (16)$$

$$0 \leq C_2(k) \leq n_2 \cdot C_{2,max} \quad \text{(EVSE power limits)} \quad (17)$$

$$0 \leq C_3(k) \leq n_3 \cdot C_{3,max} \quad \text{(EVSE power limits)} \quad (18)$$

$$\sqrt{\sigma_s^2(k) \cdot s^2 + \sigma_L^2(k)} \leq \quad \text{(minimum import power limit)} \quad (19)$$

$$\frac{1}{\Phi^{-1}(\alpha_{G1})} \cdot [\bar{L}(k) - \ s \cdot \bar{S}(k) + B_c(k) + C_2(k) + C_3(k) + G_E(k) + S_c(k) - B_d(k)]$$

or equivalently

$$||s \cdot \sigma_s(k) \cdot \sigma_L(k)||_2 \leq RHS \quad \text{(minimum import power limit)}$$

$$||s \cdot \sigma_s(k) \cdot \sigma_L(k)||_2 \leq \quad \text{(maximum import power limit)} \quad (20)$$

$$\frac{1}{\Phi^{-1}(\alpha_{G2})} \cdot [\ s \cdot \bar{S}(k) - \bar{L}(k) + G_{I,max} - B_c(k) - C_2(k) - C_3(k) - G_E(k) - S_c(k) + B_d(k)]$$

$$0 \leq G_E(k) \leq G_{E,max} \quad \text{(grid export power limit)} \quad (21)$$

$$G_I(k) \leq G_D \quad \text{(demand charge)} \quad (22)$$

or equivalently

$$[\bar{L}(k) + B_c(k) + C_2(k) + C_3(k) + G_E(k) \quad \text{(demand charge)}$$

$$-s \cdot \bar{S}(k) + S_c(k) - B_d(k)] \leq G_D$$

$$s_{min} \leq s \leq s_{max} \quad \text{(solar scale limit)} \tag{23}$$

$$b_{min} \leq b \leq b_{max} \quad \text{battery scale limit)} \tag{24}$$

$$n_2 \in \mathbb{Z} \quad \text{(number of level 2 chargers)} \tag{25}$$

$$n_3 \in \mathbb{Z} \quad \text{(number of level 3 chargers)} \tag{26}$$

$$0 \leq n_2 \leq \eta_{2,max} \quad \text{(max number of level 2 chargers)} \tag{27}$$

$$0 \leq n_3 \leq \eta_{3,max} \quad \text{(max number of level 3 chargers)} \tag{28}$$

Variables

Primary optimization variables are b, s, n_2, n_3, $B_c(k)$, $C_2(k)$, C_3 (k), $G_E(k)$, $S_c(k)$, $B_d(k)$, G_d. Optimization variables in equations (1) through (28) are highlighted in blue. A full list of optimization variables are included in Appendix A.

Data

Solar resource data were obtained from NREL's PVWatts calculator, for Oakland International Airport and mean values and standard deviations were calculated from hourly data from 2016 [11]. Solar PV pricing was determined from national and regional averages from 2017 data [12]. Daily pricing was calculated from per kWh averages as shown in equation (32).

EV related data were obtained from a BMW Group project document that included the charging start and stop times with SOC levels for a limited fleet of BMW battery EVs and plug-in hybrid EVs [13]. Mean and standard deviations were calculated from this data set.

Building load data were also obtained from the BMW Group project document. Building load data included hourly demand in kWh for homes in the San Francisco Bay area [13]. Mean and standard deviations were calculated from this data set.

Capital costs and efficiency for EV charging infrastructure were determined from industry standards, and conversion to daily costs were calculated [14] [15]. Battery costs were also determined from industry standards, and daily price was calculated from per kWh averages as shown in equation (32) [15].

A full list of parameters and values are included in Appendix B.

2.2 Refinement of the Model

Mixed Integer Programming

The general form of a mixed-integer linear program is as shown in equation 29 as follows:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f^T x \\ \text{subject to:} \quad & Ax \leq b \\ & A_{eq} x = b_{eq} \\ & x \in \mathbb{Z} \end{aligned} \tag{29}$$

Integer programs are inherently non-convex, since they impose discrete values onto the decision variables of the system. Discrete problems are non-convex because a convex set cannot be formed. Solving a mixed-integer program as such is not possible with standard CVX software due to its non-convexity. In the case of this project, the discrete states are the number of Level 2 (L2) and Level 3 (L3) chargers installed in the commercial facility. To solve the mixed integer program without using mixed integer solvers, the LP and SOCP were solved iteratively for combinations of L2 and L3 chargers. The resulting total cost at optimal were compared from each iteration to determine the number of chargers that resulted in the least cost. Equation (30) illustrates the concept of converting the mixed-integer programs into programs solvable in CVX, with n_2 and n_3 being the number of L2 and L3 chargers, respectively. To limit computation times, the maximum number of L2 and L3 chargers were set to 4, with a range from 0 to 4.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f^T x \\ \text{subject to:} \quad & Ax \leq b \\ & A_{eq}x = b_{eq} \\ & x \in \mathbb{R} \\ & n_2, n_3 \in \{1, 2, 3, 4\} \end{aligned} \tag{30}$$

Calculating Daily Costs

$$\begin{aligned} c &= \frac{1}{365 \times t} P \left(1 + r\right)^t \\ \text{Where: } c &= \text{Daily cost} \\ t &= \text{Lifetime of equipment} \\ P &= \text{Principal} \\ r &= \text{Annual investment rate} \end{aligned} \tag{31}$$

The general formula for approximating daily costs taking into account annual interest rate is shown in equation (31).

$$c = \frac{1}{365 \times t} P \tag{32}$$

For simplicity, the interest rate was not considered in this project. This reduces equation (31) into equation (32), which was used to calculate the per-day costs of installed equipment. Refer to Appendix B for a list of assumed costs.

MILP

The Mixed Integer Linear Program was developed before the Mixed-Integer Second Order Cone Program to verify the validity of the dynamics and parameters to be used in the optimization program. The MILP uses the mean values of building load, solar generation,

and minimum energy delivered to the EVs ($L(k)$, $S(k)$, $Z_{\min}(k)$, respectively) as deterministic inputs from the data collected instead of treating them as random variables. Due to the lack of chance constraints, equations (6)-(7), (15)-(16), and (19)-(20) were not considered in the MILP.

SOCP

The Second Order Cone Program considered chance constraints on the building load, solar generation, and minimum energy delivered to the EVs ($L(k)$, $S(k)$, $Z_{\min}(k)$), which distinguishes it from the MILP. It is assumed that the hourly insolation and building load have independent Gaussian distributions, with an average ($\bar{L}(k)$, $\bar{S}(k)$) and standard deviation calculated from the obtained data sources ($\sigma_L(k), \sigma_S(k)$).

3 Results

Utilizing iterations of multiple SOCP problems, our optimal number of L2 and L3 chargers are 3 and 1, respectively, considering the uncertainty described by solar generation, building energy demand, and EV charging demand as shown in Figure 2. To determine robustness, we refined the model to reflect the realistic situations of operating a microgrid. These refinements include corner cases such as simultaneous battery charging and discharging, and boundary conditions including excessive discharging of battery at the final time step due to finite time horizon of optimization, and over-speculation of vehicle charging demand.

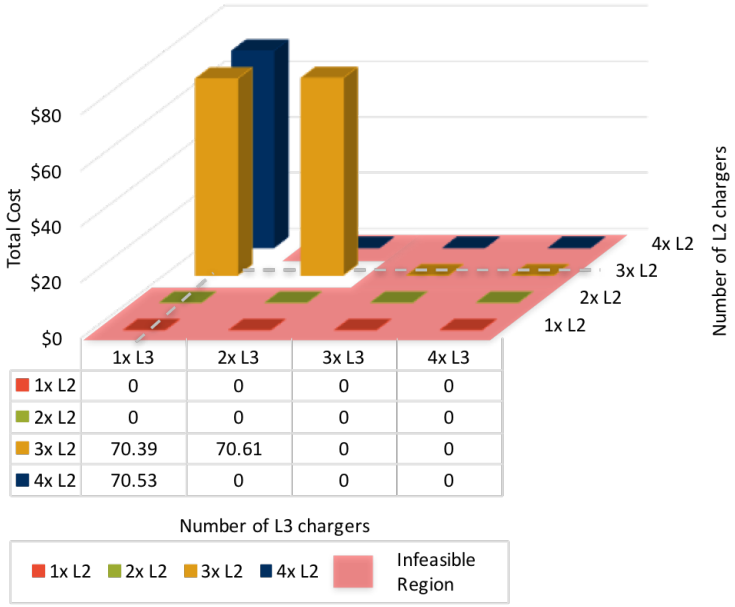


Figure 2: Optimal Size of Level 2 and Level 3 EV Chargers with Real Data. L2 and L3 reflect the number of Level 2 and Level 3 chargers, respectively. Total system cost is $70 per day.

3.1 Robust Optimal Sizing of the Microgrid

Figure 2 shows the result of SOCPs solved iteratively for combinations of L2 and L3 chargers. With the given data and assuming that L2 or L3 chargers will be used only for L2 and L3 charging demand, respectively, there are only three feasible combinations of sixteen possible combinations. Total cost of \$0 in Figure 2 represents infeasible combinations of the solution, where the objective value is unbounded. Among the feasible combinations, the optimal combination is the lowest total system cost per day, with the optimal combination three L2 chargers and one L3 charger. Total system cost includes capital cost such as solar panels, batteries, and EV chargers as well as operational costs such as building energy demand. Total system cost of \$70 per day results in \$2.33 per day per unit in the 30-unit commercial residential property, including charging costs for all EVs. At optimum, the solar scale and battery scale factor were 20 and 9.6 units, respectively. This results in a solar PV system size of 20 kW and battery size of 9.6 kW.

3.2 Robust Optimal Operation of the Microgrid

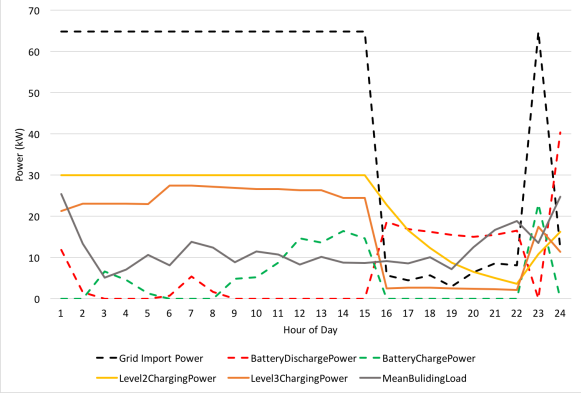


Figure 3: Optimal operation of a microgrid with EV charging using real data over 24 hours.

Figure 3 shows the result of the operation strategy at optimal size of the solar panels, battery scales, and EV chargers. Note that the TOU price of electricity import from the grid is artificially designed in this result. To simulate time-of-use tariffs, from 4PM to 10PM the grid imported electricity rate is increased threefold to observe its impact on the operation strategy. To minimize cost, the model results show that there is a dramatic decrease of grid import power during higher electricity rates, with demand met by battery discharge. Figure 3 also shows that the battery does not exhibit simultaneous charging and discharging. Refer to Section 3.5 for a detailed analysis of the behavior of battery discharge and the decrease of grid import at the final hour. Refer to Section 3.6 for a detailed analysis of EV charging behavior during peak electricity rates.

3.3 Robustness to Uncertainty in Energy Generation and Demand

The robustness of the proposed optimization problem to uncertainty of energy generation and demand is shown by comparison to a optimization problem that does not consider the uncertainty in these variables. Results show that the robust optimization model recommends installation of three L2 chargers, whereas the non-robust optimization problem recommends only one. Referring to Figure 4, it can be seen that if the charging demand of L2 occurs one standard deviation larger than the expected charging load, the non-robust optimization solution of only one charger cannot meet the charging load. However since the proposed optimization problem considers the uncertainty of the load prediction, the maximum capacity of the L2 charging with three L2 chargers from the robust model can facilitate the EV charging need.

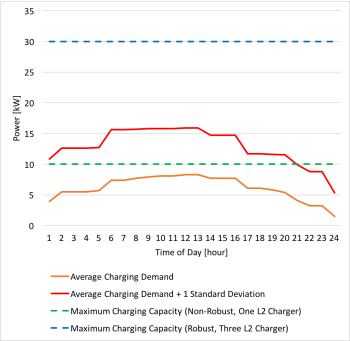


Figure 4: Robustness to Uncertainty in EV Charging Demand

3.4 Model Refinement

Refinement of the optimization model in operating conditions was necessary to maintain robustness and to account for realistic situations that may occur, such as corner cases and boundary conditions. The following scenarios as depicted in Figure 2 and Figure 3 were induced with hypothetical data.

Simultaneous Battery Charge and Discharge

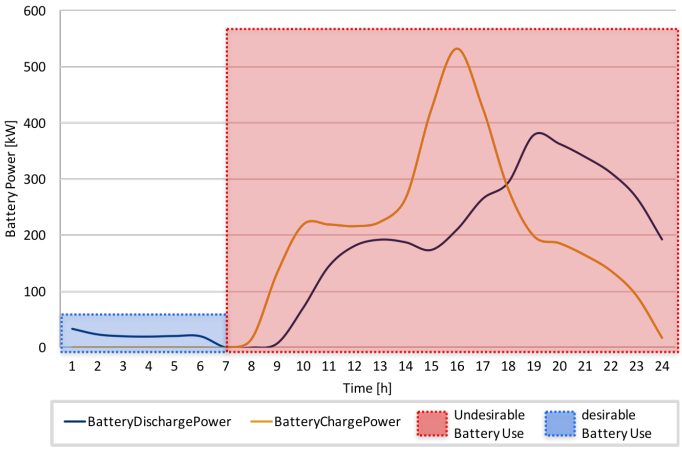


Figure 5: A Example Corner Case of Simultaneous Charging and Discharging of Battery. Blue regions indicate desired behavior while red shaded regions indicate undesirable battery behavior, i.e., simultaneous charging and discharging.

Referring to the constraints of the optimization problem as proposed, it was determined that a "battery wear" or "battery health" constraint was required. In real world scenarios it is undesirable to induce simultaneous charging and discharging as it degrades battery life. However, while implementing various scenarios of data input, i.e. different scales of solar energy generation, EV charging demand, and building energy demand, it was determined that it is mathematically possible to produce simultaneous charging and discharging of battery. This result is shown in Figure 5, where we observe both simultaneous charging and discharging after 8AM.

To induce this behavior, the total energy demand in the MG is significantly and unrealistically smaller compared to solar generation, while energy export to the grid is maximized from solar energy generation. As the original problem was not constrained to consider the battery's internal resistance, the optimization problem attempted to expend surplus energy generation. Since this is a corner case derived from unrealistic data, the solution proposed to this issue is to observe and analyze the input data to the optimization problem and adjust the optimization scenario to prevent such a result.

Excessive Battery Discharge on Final Time-step

As the optimization problem considers the operation of the MG in a finite time horizon, the model tries to maximize the profit without consideration to the future after the time horizon. Therefore, the optimal result may produce a scenario in which the battery discharges either to the grid or to EV chargers that increase the revenue at the final time step. In Figure 6, the battery discharges largely at the final time step, and the battery size at optimal is the maximum level. This indicates that the model tries to maximize profit by use of the discharge of battery unrealistically. As this result is short-sighted and unsustainable for the operation of the MG, an additional constraint on the final SOC of the battery is necessary to prohibit this behavior. This constraint is described in the equation (5), limiting battery SOC at the final state to greater than or equal to 10 percent.

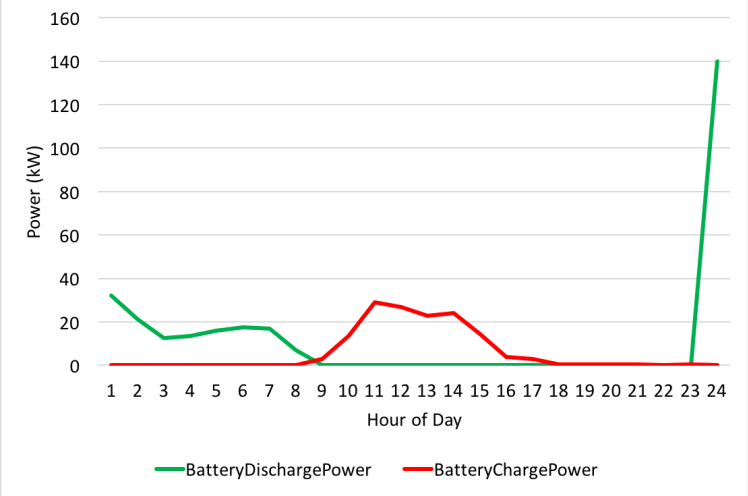


Figure 6: Comparison between Bounded and Unbounded Final SOC of the Battery

In Figure 3, results show that final hour discharge of the battery still occurs and grid import decreases even after implementing the final SOC level constraint. However, the scale of discharge is smaller than that of Figure 6, since the battery size at optimum is much more realistic in Figure 3.

Over-speculation of Vehicle Charging Demand

Due to conversion of the MISOCP to multiple SOCPs another boundary condition required refinement. Though the reduction in complexity of the problem is a large benefit to computation, iterating the problem over a given number of chargers creates over-speculation of vehicle charging demand. While the MISOCP considers the number of EV chargers as an optimization variable, the SOCP considers the number of chargers as a fixed given parameter. Since the objective function of

the SOCP is an affine function, the constant cost of chargers does not play a role in determining the planning and operation of the MG at optimum. Therefore, it is in the interest of each SOCP to maximize the profit by selling as much EV charging load as possible with a given number of chargers.

This strategy becomes unrealistic when the robust optimization scheme was adopted. Consider a non-robust optimization, where the expected charging load has no uncertainty. The optimal result would be to produce the exact amount of the EV charging load. However, a robust optimization problem considers the uncertainty in the prediction of the EV charging load and only requires the MG to be prepared for the EV charging load higher than the average load by a certain uncertainty level. This only presents the lower bound of the EV charging load as a constraint.

The absence of the upper bound for EV charging load conflicts with the motive of the SOCP within a given set of constant chargers to maximize revenue from the charging of vehicles. Therefore the result at optimum plans for an EV charging load that is much larger than the mean value. In other words, the EV charging load is bounded only by the physical power limits of the given EV chargers. This over-speculation of the vehicle charging demand only considers the minimum required EV charging load for EV mobility needs, but is not bound by a realistic EV demand scenario, i.e., it is unlikely that EV charging demand will be at such a high level at all time steps.

To address this boundary condition, additional constraints can be formulated to limit the upper bound of EV charging demand for each SOCP and therefore limit the over-speculation of the vehicle charging demand. This constraint is described in the equations (6) and (7), where we limit demand to a threshold value. The result of this constraint is given in Figure 7, where L3 charging power and charging demand on average are presented. The model then binds the EV charging power to be at maximum the 99th percentile EV charging demand throughout the time series.

Comparisons between the results depicted in Figure 3 and Figure 7 point to additional observations about model performance. In Figure 3, during peak electricity rates, the charging power of both L2 and L3 decrease while still matching the minimum delivered energy to the vehicles. During this time period, the electricity rate increases from \$0.26 to \$0.78 USD per kWh, while the revenue from the L2 and L3 charging are held constant as \$0.20 and \$0.30 USD per kWh, respectively. Therefore, during the time of low rates, the model maximizes revenue from the MG as much as possible. However, during the time of high electricity rates, the grid import price for charging the vehicles is

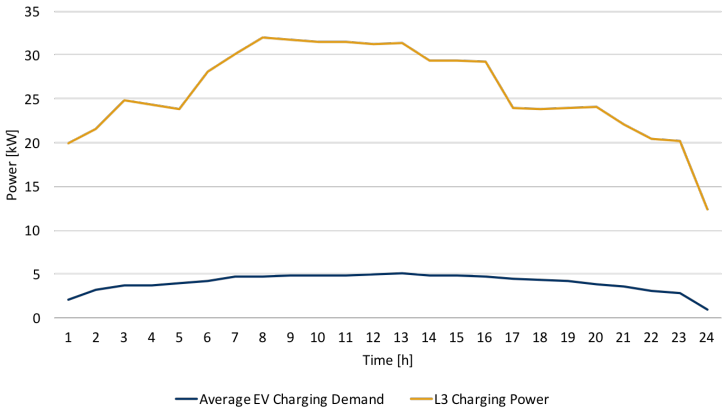


Figure 7: The Upper Bound of EV Charging Demand Uncertainty

not cost-effective, and the optimal strategy is to maintain charging at the minimum required level.

4 Discussion

Mixed-Integer Linear Program

Due to computation times both L2 and L3 chargers ranging between 1 to 4 is about 20 seconds on MATLAB, the MILP was an ideal test bed for verifying the fidelity of the code and the parameters used. However, each iteration could only be used for a single determinate scenario, which is insufficient for robust optimization. Various scenarios must be tested by iterating through cases such as days with low solar irradiation. Thus, even though each iteration is computationally light, the necessity for conducting various iterations for different scenarios makes the overall computational penalty of the MILP high. Considering the case of using this program for planning, the MILP fails to provide a single optimal case, as every iteration produces varying optimal values for equipment sizing.

Several cases also show that the MILP is not the most practical optimization method for planning a commercial MG. As long as the charging limits of L3 chargers are not active, the LP assumes that none of the EVs will be charging on L2, since the revenue per kWh for L3 charging is higher. This assumption is not realistic since popular PHEV vehicles such as the Chevrolet Volt cannot charge via L3 [16]. The MILP is also able to derive a feasible solution even when solar generation drops to nearly zero, as long as grid power can be imported. Additionally, the MILP fails to provide a feasible solution when the building load exceeds four times the mean load, or when the EV charging demand exceeds thirteen times the mean EV demand used. Both of these cases are better managed by the SOCP that uses chance constraints instead of a deterministic model.

Mixed-Integer Second-Order Cone Program

The MISOCP requires far greater computational power and often requires several hours for the calculations to be completed. However, it results in a global optimal planning scenario, as it is robust to stochasticity in energy generation and demand on the order of 1 hour. Therefore, the MISOCP is the preferred optimization program for the case of planning a MG. Using the MISOCP for planning MGs can advance adaptation of renewable energy resources while decreasing dependence on grid imports, as risk factors surrounding return on investment are not often predictable. However, since the MISOCP provides a benchmark of projected daily revenue or cost while taking into account stochastic uncertainties, it is able to present a more robust case for implementing a commercial MG in comparison to the MILP that is not able to consider uncertainties in energy demand and supply, and thus costs and revenues.

Future Work

Excessive battery discharge during the final time-step was revealed to related to the model selling power to the grid before the end of its 24 hour "lifetime" to maximize revenue for the day. Although a new constraint was implemented to avoid the battery from fully discharging during the final time-step, it did not address the cause of this phenomenon. Some cases in the MILP did occur where the battery was not being used other than discharging during the final time-step. With the current model, the initial energy stored in the battery does not have an incurred cost to it. A possible area of investigation would be to analyze the relationship between the revenue generated by the discharge and the TOU cost of energy imported from the grid.

The current models assume a time-step of 1 hour. The data collected to test the MILP and the MISOCP were in terms of hourly increments. In reality, a true MG system should adapt to rapid changes in energy generation and demand, such as when an energy-intensive appliance is turned on. These changes occur in smaller time-steps, between seconds to minutes. Shorter time-scales may require modified behavior for some system components. For example, the power capacity of the battery may have to increase for it to be able to regulate the system by injecting power as necessary in short time increments.

The results from both the MISOCP and MILP show that the optimal sizing for the number of solar panels installed was often the maximum number possible. Although this model was able to calculate the optimal sizing from an economic perspective, it does not take into account the stability of the system. For example, in a scenario where the number of solar panels is high and cloud cover suddenly clears, the increased generation from solar will result in a sharp spike in energy supply. Without a protective device that limits the power flow from solar generation, the system may attempt to shed excess energy in its battery as discussed above. Costs incurred by the addition of such protective devices should also be considered in the case of planning a commercial MG. Furthermore, the validity of the constraints must be confirmed through stability analysis instead of assuming an arbitrary number of maximum installed equipment.

Finally, V2G technologies have been gaining interest from both the public and private sector as a method to stabilize the electricity grid as renewable penetration increases [17]. This optimization program did not include V2G capabilities, such that EVs plugged into the chargers were not able to sell their electricity back to the grid. Enabling V2G capabilities in the optimization program may be beneficial for future commercial use as vehicle owners are able to choose whether they would like their vehicle to export power to the MG, and increase revenue potential therefore minimizing cost for the system.

5 Summary

In this project, we developed a MISOCP for robust optimal planning and operation of a commercial EV charging facility. The model was developed using real capital cost data, artificial TOU pricing scheme, real solar data, EV charging data, and building load data. The robust model results in an optimal design consisting of three L2 chargers and one L3 charger, with a total system daily cost of $70.39 per day. By comparison, the non-robust

model calculates an optimal design of one L2 charger and four L3 chargers. From this example, it is clear that the robust model is well-suited for a likely range of scenarios, while the non-robust model optimizes for the rare case of unusually high charging demand. Further model development could include constraints for battery health, finer time-steps, bi-directional EV charging, and accounting for capital costs of MG stabilization equipment.

References

[1] Frost & Sullivan. (2017) Global electric vehicle market outlook, 2017. [Online]. Available: https://cds.frost.com/p/72699/#!/ppt/c?id=MCC9-01-00-00-00

[2] Electric Power Research Institute (EPRI), "Understanding the grid impacts of plug-in electric vehicles (pev)," 2012.

[3] N. G. Paterakis, O. Erdinç, I. N. Pappi, A. G. Bakirtzis, and J. P. S. Catalão, "Coordinated operation of a neighborhood of smart households comprising electric vehicles, energy storage and distributed generation," *IEEE Transactions on Smart Grid*, vol. 7, no. 6, pp. 2736–2747, Nov 2016.

[4] J. Munkhammar, J. Widen, and J. Ryden, "On a probability distribution model combining household power consumption, electric vehicle home-charging and photovoltaic power production," *Applied Energy*, vol. 142, pp. 135 – 143, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261914012884

[5] S. Gao, K. T. Chau, C. Liu, D. Wu, and C. C. Chan, "Integrated energy management of plug-in electric vehicles in power grid with renewables," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 7, pp. 3019–3027, Sept 2014.

[6] S. J. Moura, "White paper: Robust optimal sizing energy management for solar + storage + ev chargers," Department of Civil and Environmental Engineering, University of California, Berkeley, Berkeley, CA, Tech. Rep., December 2017.

[7] O. Erdinc, "Economic impacts of small-scale own generating and storage units, and electric vehicles under different demand response strategies for smart households," *Applied Energy*, vol. 126, pp. 142 – 150, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261914003535

[8] O. Erdinc, N. G. Paterakis, I. N. Pappi, A. G. Bakirtzis, and J. P. CatalÃ£o, "A new perspective for sizing of distributed generation and energy storage for smart households under demand response," *Applied Energy*, vol. 143, pp. 26 – 37, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261915000318

[9] O. Erdinc, N. G. Paterakis, T. D. P. Mendes, A. G. Bakirtzis, and J. P. S. Catalão, "Smart household operation considering bi-directional ev and ess utilization by real-time pricing-based dr," *IEEE Transactions on Smart Grid*, vol. 6, no. 3, pp. 1281–1291, May 2015.

[10] S. Bahramara and H. Golpîra, "Robust optimization of micro-grids operation problem in the presence of electric vehicles," *Sustainable Cities and Society*, vol. 37, pp. 388 – 395, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210670717303505

[11] NREL. (2018) Pv watts calculator. [Online]. Available: https://pvwatts.nrel.gov/pvwatts.php

[12] R. Fu, D. Feldman, R. Margolis, M. Woodhouse, and K. Ardani, "U.s. solar photovoltaic system cost benchmark: Q1 2017," *NREL*, 2017.

[13] BMW Group, "White paper: Ev charging data," Tech. Rep.

[14] M. Gupta, "U.s. front-of-the-meter energy storage system prices 2018-2022," *GTM Research*, 2018.

[15] S. Knupfer, J. Noffsinger, and S. Sahdev, "How battery storage can help charge the electric-vehicle market," *McKinsey Center for Future Mobility*, 2018.

[16] EVTown. (2015) Levels of charging. [Online]. Available: http://www.evtown.org/about-ev-town/ev-charging/charging-levels.html

[17] K. M. Tan, V. K. Ramachandaramurthy, and J. Y. Yong, "Bidirectional battery charger for electric vehicle," in *2014 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, May 2014, pp. 406–411.

6 Appendix

Appendix A: List of Variables

List of Variables		
Variable	**Units**	**Description**
s, b	[-]	Scale factors for solar and ESS sizes
$G_{\mathrm{I}}(k), G_{\mathrm{E}}(k)$	[kW]	Power imported and exported from/to grid
G_{D}	[kW]	Maximum power for demand charge
$S(k)$	[kW]	(r.v.) Power generated from solar
$S_{\mathrm{c}}(k)$	[kW]	Curtailed solar power
$B_{\mathrm{d}}(k), B_{\mathrm{c}}(k)$	[kW]	Power discharged from / charged into battery
$C_2(k), C_3(k)$	[kW]	Charging load of L2, L3 EVSEs
$L(k)$	[kW]	(r.v.) Power load of facility
$E(k)$	[kW]	Energy level of ESS
$Z(k)$	[kWh]	Cumulative energy delivered to EVs
$Z_{\min}(k)$	[kWh]	(r.v.) Minimum delivered energy to EVs

Appendix B: List of parameters used in the optimization programs

List of Parameters Used			
Parameter	**Value**	**Units**	**Description**
c_b	0.0274	[USD/day]	Battery marginal cost for 20 year payment
c_s	0.0822	[USD/day]	Solar marginal cost for a 20 year payment
mc_2	0.137	[USD]	Marginal cost of level 2 chargers
mc_3	0.219	[USD]	Marginal cost of level 3 chargers
c_i	0.26	[USD/kWh]	Price of grid-imported power
r_e	0.03	[USD/kW]	Revenue of exported power to the grid
r_2	0.20	[USD/kWh]	Revenue for level 2 charging
r_3	0.30	[USD/kWh]	Revenue for level 2 charging
c_d	0.30	[USD/kWh]	Demand charge
dt	1	[hour]	Timestep
η_c	98	[%]	ESS charge efficiency
η_d	98	[%]	ESS discharge efficiency
E_0	5	[kWh]	Initial ESS energy level
E_{max}	14	[kWh]	Nominal ESS energy capacity
B_{max}	7	[kW]	Nominal ESS power capacity
η_2	98	[%]	L2 charge efficiency
η_3	98	[%]	L3 charge efficiency
$C_{2,max}$	10	[kW]	L2 EVSE power capacity
$C_{3,max}$	30	[kW]	L3 EVSE power capacity
$G_{I,max}$	100	[kW]	Maximum grid import power
$G_{E,max}$	100	[kW]	Maximum grid export power
s_{min}	1	[ ]	Solar scale limit

s_{max}	20	[ ]	Solar scale limit
b_{min}	1	[ ]	Battery scale limit
b_{max}	20	[ ]	Battery scale limit
α_z	0.95	[ ]	Probability of satisfying EV energy limits
$\alpha_{G,1}$	0.95	[ ]	Probability of satisfying grid import limits
$\alpha_{G,2}$	0.95	[ ]	Probability of satisfying grid import limits

Acknowledgments

Thank you mom and data, Professor Moura and Bertrand for their time and patience

About the authors

Armando A. Domingos is a M.S. candidate in the Civil and Environmental Engineering department at University of California, Berkeley. He likes cars, and thinks the best PHEV is the Porsche 918.

Marc Hutton is a M.S. candidate in the Energy, Civil Infrastructure, and Climate program in the Civil and Environmental Engineering department at University of California, Berkeley. He hopes this project will help him get a job at Tesla (or any job, really).

Aaron Jagtianey is a B.S. candidate in the Energy Engineering program at the University of California, Berkeley. He is a fan of ~~the high revving Honda Civic Type R~~ EVs.

Soomin Woo is a PhD candidate in the Civil and Environmental Engineering with focus on transportation engineering at the University of California, Berkeley.

Part II

Electric Vehicles and Transportation

The Application of Estimation Techniques to Autonomous Vehicles Related Object Tracking Problems

Hanxiao Deng, Haoran Su, Kun Qian, Yue Hu

1 Abstract

Tracking objects over time is a major challenge for understanding the present location and environment surrounding an autonomous vehicle. Kalman filters(with its extensions) and Particle filter are traditional but very powerful estimation techniques in estimation problems. In this project, we will implement a Kalman Filter and an Extended Kalman Filter in C++ environment to predict the location of other moving vehicles on the road with noisy LiDAR and Radar measurements around the autonomous car. We will also estimate the location of an autonomous vehicle with the measured location using Particle Filter. The main focus of this project is to apply estimation techniques to autonomous vehicle related estimation and tracking problems.

2 Introduction

2.1 Motivation and Background

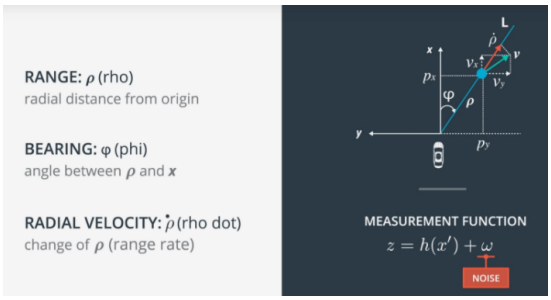
Autonomous Vehicle is a heated topic recently. A key technique used is state estimation-to understand the location of the vehicle itself and the environment around. Here, we want to solve two problems: Tracking around objects with LiDAR and Radar on an autonomous car; Tracking the pose of an autonomous vehicle with noisy location measurement data. We simplify the scenario that there are localization measurements by LiDAR and Radar of only one moving car, as shown in figure 1. We describe how data from LiDAR and Radar can be used for optimal estimation with Kalman filter. Sensor fusion of Lidar and radar combines the advantages of both sensor types to increase the precision of estimation. In the vehicle's pose estimation problem, we suppose that the sensor will present us with the location data of a vehicle in 2D sense.

The LIDAR will produce 3D measurement p_x, p_y, p_z. But for the case of driving on the road, we could simplify the pose of the tracked object as: p_x, p_y and one rotation. In other words, we could only use p_x, p_y to indicate the position of the object, and one rotation to indicate the orientation of the object. But in real world where you have very steep road, you have to consider z axis as well. Also in application like airplane and drone, you definitely want to consider p_z as well.

2.2 Relavent Literature

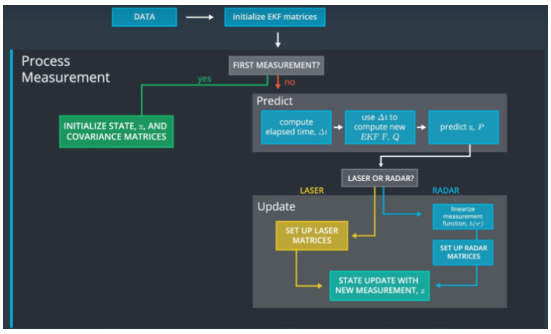
All Kalman filters have similar main steps: 1. Initialization, 2. Prior Prediction, 3. Measurement Update. A Standard Kalman Filter (KF) can only handle linear equations [1]. Both the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF) allow you to use non-linear equations. The work flow

(a) LiDAR Measurement

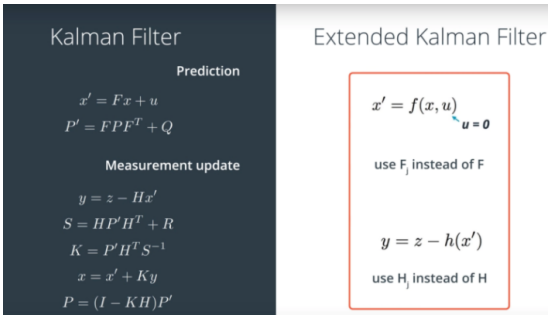


(b) Radar Measurement

Figure 1: Lidar and Radar Measurements



(a) Extended Kalman Filter Workflow



(b) Extended Kalman Filter v.s. Kalman Filter

Figure 2: Kalman Filter and Extended Kalman Filter

for Extended Kalman Filter and its comparison with Kalman Filter is shown in figure 2. The difference between EKF and UKF is how they handle non-linear equations: Extended Kalman Filter uses the Jacobian matrix to linearize non-linear functions[2]; Unscented Kalman Filter, on the other hand, does not need to linearize non-linear functions, instead, the unscented Kalman filter takes representative points from a distribution and then employ unscented transform[3]. These tools come in handy when trying to estimate the trajectory from LiDAR or RADAR data, where depending on the measurement we can both encounter linear and nonlinear equations.

Another problem to solve is that how to extract location information about vehicles from cloud point data collected by LiDAR. People made research on using machine learning techniques to extract vehicles' location information from cloud point data[6]. We follow the track to extract vehicle's location information from cloud point data that LiDAR collected.

For the tracking of autonomous vehicles itself, we use the simplified bicycle model[7] to describe the autonomous vehicle which is proved to be relatively precise and computationally efficient. We will set up the state space functions based on the bicycle model and apply particle filters directly.

2.3 Focus of this study

1. Implement feasible estimators to track other vehicles in the environment with LiDAR and Radar measurements;
2. Estimate the pose of vehicle based on the location measurement;
3. Improve algorithms to implement estimation to meet real time requirement;

4. Discuss and compare the precision and computation amount between different techniques;

3 Model Development

3.1 Model for Vehicles Tracking Problem

3.1.1 The goal of our state space model

We would like to depend on the state space to estimate the position of other moving cars. The data we have is the measurement data by Lidar and Radar. Lidar will give us the position data of a moving vehicles. We suppose that the Lidar data comes directly in the caritsean grid. Radar will provide us the position and velocity data of a moving object, however, in the polar grid.

$z_{Lidar} = \begin{bmatrix} x \\ y \end{bmatrix}$, $z_{radar} = \begin{bmatrix} \rho \\ \theta \\ \dot{\rho} \end{bmatrix}$

3.1.2 Physics function of our system

In fact, the physics equation that we depend on to do the tracking problem is the Motion function. Because here we have no knowledge about the acceleration information about the object that we are tracking, thus we consider the acceleration as a noise. It may cause some problems because usually Kalman Filter requires noise to be unbiased, however, here acceleration is sometimes biased.

$$\dot{x} = v$$

$$\dot{v} = \nu$$

We would like to write this equation in a discrete form.

$$x_{k+1} = x_k + \eta v_k + \nu_1$$

$$v_{k+1} = v_k + \nu_2$$

in which x_k is the location of the vehicle at time k, η is the time step size, v_k is the speed at time k, ν is the process noise of the model. Here the process noise mainly comes from our ignorance of acceleration information.

We consider the problem from $2D$ sense. So that the state of our system will be $\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}$ and the formulation of the state space model in continuous form will be

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v_x} \\ \dot{v_y} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix} + \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \\ \nu_4 \end{bmatrix}$$

If we write it in the discrete form, it would be(with the η be the time step)

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ v_x(k+1) \\ v_y(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \eta & 0 \\ 0 & 1 & 0 & \eta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \end{bmatrix} + \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \\ \nu_4 \end{bmatrix}$$

So that we will have the A matrix:

$$A = \begin{bmatrix} 1 & 0 & \eta & 0 \\ 0 & 1 & 0 & \eta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.1.3 The form of measurement equation

LiDAR data

As stated above, we suppose that Lidar will directly give us the location measurement at Cartisean grid. The measurement equation will be

$$\begin{bmatrix} z_x(k) \\ z_y(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \end{bmatrix} + \begin{bmatrix} w_x \\ w_y \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

in which z is the measurement data and w is the measurement noise.
To ensure that our estimation error will finally converge, here we want to see the observability or detectability of this system.
For the simplicity of our work, we use PBH test. Because the $\lambda_A = 1$, then

$$\begin{bmatrix} \lambda_A I - A \\ H \end{bmatrix} = \begin{bmatrix} 0 & 0 & \eta & 0 \\ 0 & 0 & 0 & \eta \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$Rank(\begin{bmatrix} \lambda_A I - A \\ H \end{bmatrix}) = 4$$

So this system is observable. In fact to ensure the convergence of Kalman filter error, we only need detectable. Anyway, here observability will do a lot of good to our problem solving and also proves that our formulation of this system is reasonable.

Radar data

The case for Radar measurement data is a little bit more complex. Because we need to do the transformation between polar and Cartisean. We suppose that the Radar will present us with the data $\begin{bmatrix} \rho \\ \dot{\rho} \\ \Phi \end{bmatrix}$ and the relationship with our state variables are

$$\rho = \sqrt{x^2 + y^2}$$

$$\Phi = \arctan(\frac{y}{x})$$

$$\dot{\rho} = \sqrt{v_x^2 + v_y^2} \cos{(\Phi - \arctan{\frac{v_y}{v_x}})}$$

We can see that the measurement output of Radar data is nonlinear. Here we can use both EKF or UKF. EKF depends on the linearization process to translate the problem to the Kalman Filter's range. UKF depends on the unscented transform to pass on the PDF of estimation object.
Here we will use EKF first, so we need to calculate the Jacobian of the nonlinear output function. The Jacobian is calculated as follows.

$$\frac{\partial \rho}{\partial x} = \frac{x}{\sqrt{x^2+y^2}}$$
$$\frac{\partial \rho}{\partial y} = \frac{y}{\sqrt{x^2+y^2}}$$
$$\frac{\partial \Phi}{\partial x} = -\frac{1}{\sqrt{x^2+y^2}}$$
$$\frac{\partial \Phi}{\partial y} = \frac{x_1}{\sqrt{x^2+y^2}}$$
$$\frac{\partial \dot{\rho}}{\partial x} = \sqrt{v_x^2+v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{1}{\sqrt{x^2+y^2}})$$
$$\frac{\partial \dot{\rho}}{\partial y} = \sqrt{v_x^2+v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(\frac{x}{\sqrt{x^2+y^2}})$$
$$\frac{\partial \dot{\rho}}{\partial v_x} = \sqrt{v_x^2+v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{v_x}{\sqrt{v_x^2+v_y^2}}) + \frac{v_x}{\sqrt{v_x^2+v_y^2}}\cos(\arctan(\frac{y}{x}) - \arctan\frac{v_y}{v_x})$$
$$\frac{\partial \dot{\rho}}{\partial v_y} = \sqrt{v_x^2+v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{v_x}{\sqrt{v_x^2+v_y^2}}) + \frac{v_y}{\sqrt{v_x^2+v_y^2}}\cos(\arctan(\frac{y}{x}) - \arctan\frac{v_y}{v_x})$$

3.2 Model for Vehicle Pose Estimation

In studying the dynamics of autonomous vehicles, as stated before, we usually use the simplified bicycle model to study the problem. The bicycle model consider the vehicle as a bicycle, and consider the location, heading angle as states of the bicycle.
For the inputs of the system, we will have knowledge of the steering angle and rotating speed of wheel. Comparing to the more complexed model, the simplified bicycle model doesn't consider side slip of wheels and thus can describe the system with less states.
The description of the bicycle model is as below. In which we will use x, y, θ as the state of the system. γ and ω are known inputs.

(The Picture is taken from E-Bicycle website, [9])

$$\dot{x}(t) = v(t)\cos(\theta(t))$$
$$\dot{y}(t) = v(t)\sin(\theta(t))$$
$$\dot{\theta}(t) = \frac{v(t)}{B}\tan(\gamma(t))$$

According to the physical relation, $v(t) = r\omega_B(t)$, $\omega_B(t) = 5\omega(t)$, t we will have $v(t) = 5r\omega(t)$. Then the equation will turn into the following form.

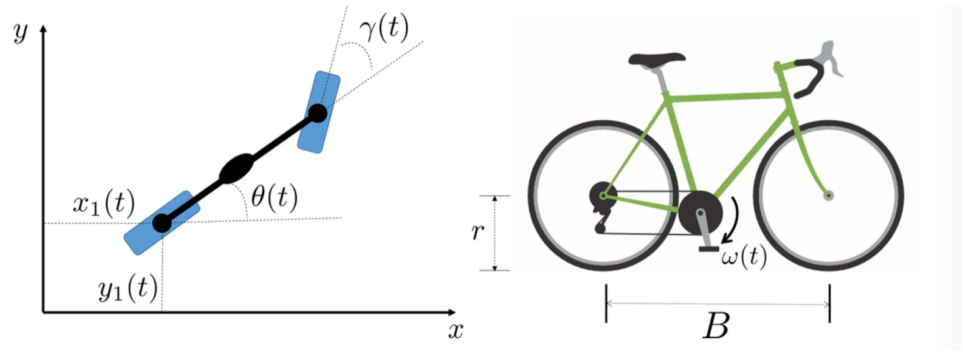


Figure 3: Bicycle Model

$$\dot{x}(t) = 5r\omega(t)\cos(\theta(t))$$
$$\dot{y}(t) = 5r\omega(t)\sin(\theta(t))$$
$$\dot{\theta}(t) = \frac{5r\omega(t)}{B}\tan(\gamma(t))$$

For the observation part, we are going to measure the location of the center of the bicycle. The observation functions are as follows.

$$x_c(t) = x(t) + \frac{1}{2}B\cos(\theta(t))$$
$$y_c(t) = y(t) + \frac{1}{2}B\sin(\theta(t))$$

In which $\gamma(t)$ and $\omega(t)$ are known inputs.
We will consider the noise terms in the following part.

3.2.1 Discrete Model

We use the first order Euler approximation to turn the continuous system into a discrete one. The discrete equations are as follows.

$$x(t+1) = x(t) + 5r\omega(t)\cos(\theta(t))\Delta t$$
$$y(t+1) = y(t) + 5r\omega(t)\sin(\theta(t))\Delta t$$
$$\theta(t+1) = \theta(t) + \frac{5r}{B}\omega(t)\tan(\gamma(t))\Delta t$$

The observation model will be the same as above.

3.2.2 Assumption on Noise

Let's then consider the noise terms.

Process Noise

According to the provided information, process noise can come from several sources.

1. The uncertainty in the size of bicycle B and r, the uncertainty of B is given between $\pm 10\%$, the uncertainty of r is given between $\pm 5\%$, however, for specific cars, the size is fixed.
2. The noise in the measurement of $\omega(t)$ and $\gamma(t)$
3. The time step may have uncertainty

We make the following assumptions on process noise.

1. $\omega(t)$ and $\gamma(t)$ also follow Gaussian distribution with $N(\omega(t), 0.01)$ and $N(\gamma(t), 0.01)$.
2. Comparing to other noise, the noise in time step size can be roughly ignored. Also, to solve the problem that measurement time gap is not fixed, we can just do the process update without measurement update if we don't have measurement at that step.

Measurement Noise

Measurement noise mainly comes from the noise in measuring the vehicle's location. We consider it as additive noise. The system with noise term will be as follows.

$$x_c(t) = x(t) + \frac{1}{2}B\cos(\theta(t)) + w_1$$
$$y_c(t) = y(t) + \frac{1}{2}B\sin(\theta(t)) + w_2$$

We make the assumption that the distribution of measurement noise will be the same all the time. So we will be able to calculate out the characteristics of measurement noise through callibration dataset.
Through the calculation, we can see that the distribution of measurement noise is as follows.
The variance of w_1 and w_2 are listed as below(we made the assumption that w_1 and w_2 are independent with each other):

$$\sigma_{w_1}^2 = 1, \sigma_{w_2}^2 = 1.7$$

Noise Model for Particle Filter

$$x(t+1) = x(t) + 5r\omega(t)(1 + v_{\omega(t)})\cos(\theta(t))\Delta t$$
$$y(t+1) = y(t) + 5r\omega(t)(1 + v_{\omega(t)})\sin(\theta(t))\Delta t$$
$$\theta(t+1) = \theta(t) + \frac{5r}{B}\omega(t)\tan(\gamma(t)(1 + v_{\gamma(t)}))\Delta t$$

$$x_c(t) = x(t) + \frac{1}{2}B\cos(\theta(t)) + w_1$$
$$y_c(t) = y(t) + \frac{1}{2}B\sin(\theta(t)) + w_2$$

in which $v_{\gamma(t)} \sim N(0, 0.1)$, $v_{\omega(t)} \sim N(0, 0.1)$, $w_1 \sim N(0, 1)$, $w_2 \sim N(0, 1.7)$.
We make the assumption on the distribution of initial state that $x(0) \sim N(0, 3)$, $y(0) \sim N(0, 3)$, $\theta(0)$ dim $N(\frac{\pi}{4}, 0.3)$.

4 The Iterative Form of Estimators

In numerical implementation, we need to do the estimation in discrete case. The estimation methods usually follow the Bayesian tracking thought in which three steps are initialization, prior update and measurement update. Different methods based on different assumptions on noise and different methods to describe the property of the distribution of the states. Detailed introductions are given below about different estimation techniques. We refer to relevant studying materials about these methods.[8]

4.1 Kalman Filter update

In the lecture we introduced the continuous form of Kalman filter. However, when doing computation, we should use the discrete form which is much easier to understand.
We use some auxiliary variables here.

$$x_m(0) = x(0)$$
$$x_p(k) = A(k-1)x_m(k-1) + \nu(k-1)$$
$$z_m(k) = H(k)x_p(k) + w(k)$$
$$f_{x_m(k)} = f_{x_p(k)|z_m(k)}$$

The philosophy behind this is two steps of update.
The first update is called prior update which is denoted by x_p. In this update step, we get our prior knowledge about the state with $x_m(k-1)$ and our process function(our state space model).
The x_m denote the measurement update of x which is defined above through its PDf. And from the iterative view, x_m is in fact the estimated x with the adjustment from measured data z.
The desired estimated state here is in fact to be extracted from the PDF of x_m.
Because of the finiteness of our inner storage space, we cannot store all the information about PDF. We will only pass the expected value and variance to the next iterative. The following is the iterative algorithm:

$$\begin{aligned}
\text{init: } \widehat{x}_m(0) &= x_0 \\
\widehat{x}_p(k) &= A(k-1)\widehat{x}_m(k-1) \\
P_p(k) &= A(k-1)P_m(k-1)A^\top(k-1) + Q(k-1) \\
K(k) &= P_p(k)H^\top(k)(H(k)P_p(k)H^\top(k) + R(k))^(- 1) \\
\widehat{x}_m(k) &= \widehat{x}_p(k) + K(k)(z(k) - H(k)\widehat{x}_p(k)) \\
P_m(k) &= (I - K(k)H(k))P_p(k-1)(I - K(k)H(k))^\top + K(k)R(k)K^\top(k)
\end{aligned}$$

To make the process more intuitive, I implemented it in Python as follows:

4.2 EKF difference

The only difference that EKF may have with KF is that EKF deals with nonlinear functions. However, EKF is not a magic but only an approximation with Taylor 1st order expansion. So here, in order to deal with the nonlinearity that Radar measurement brings to us, we are going to linearize the measurement data around every time period.

In fact, as I have listed the Jacobian of the nonlinear function, I just need to fill the linearized H matrix with the value that I have got.

$$H(k) = \begin{bmatrix} \frac{\partial \rho}{\partial x} & \frac{\partial \rho}{\partial y} & 0 & 0 \\ \frac{\partial \dot{\rho}}{\partial x} & \frac{\partial \dot{\rho}}{\partial y} & \frac{\partial \dot{\rho}}{\partial v_x} & \frac{\partial \dot{\rho}}{\partial v_y} \\ \frac{\partial \Phi}{\partial x} & \frac{\partial \Phi}{\partial y} & 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{x}{\sqrt{x^2+y^2}} & \frac{y}{\sqrt{x^2+y^2}} & 0 & 0 \\ \frac{\partial \dot{\rho}}{\partial x} & \frac{\partial \dot{\rho}}{\partial y} & \frac{\partial \dot{\rho}}{\partial v_x} & \frac{\partial \dot{\rho}}{\partial v_y} \\ -\frac{1}{\sqrt{x^2+y^2}} & \frac{x_1}{\sqrt{x^2+y^2}} & 0 & 0 \end{bmatrix}$$

in which

$$\frac{\partial \dot{\rho}}{\partial x} = \sqrt{v_x^2 + v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{1}{\sqrt{x^2+y^2}})$$

$$\frac{\partial \dot{\rho}}{\partial y} = \sqrt{v_x^2 + v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(\frac{x}{\sqrt{x^2+y^2}})$$

$$\frac{\partial \dot{\rho}}{\partial v_x} = \sqrt{v_x^2 + v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{v_x}{\sqrt{v_x^2+v_y^2}}) + \frac{v_x}{\sqrt{v_x^2+v_y^2}}\cos(\arctan(\frac{y}{x}) - \arctan\frac{v_y}{v_x})$$

$$\frac{\partial \dot{\rho}}{\partial v_y} = \sqrt{v_x^2 + v_y^2}(-\sin(\arctan(\frac{y}{x}) - \arctan(\frac{v_y}{v_x})))(-\frac{v_x}{\sqrt{v_x^2+v_y^2}}) + \frac{v_y}{\sqrt{v_x^2+v_y^2}}\cos(\arctan(\frac{y}{x}) - \arctan\frac{v_y}{v_x})$$

4.3 Particle Filter

Particle Filter is in fact similar to Kalman Filter inside. They are all Bayesian tracking problem based Bayes rule. However, Particle Filter employ Monte Carlo Method to describe the distribution. Also particle filter is not strict on the requirement of noise distribution. Usually the more particles we have, the better the estimation will be but also more expensive in computation. We pick up roughly 1000 particles to characterize the distribution of states of autonomous vehicle's location information.

Initialization

Draw N samples $x_m^n(0)$ based on $f(x(0))$. These are initial particles.

Prior update

$x_p^n = q_{k-1}(x_m^n(k-1), v^n(k-1))$, for $n = 1, 2, ..., N$
in which $v^n(k-1)$ will be N particles sampled from $f(v(k-1))$, the function q_k will be the process function listed above.

Measurement update

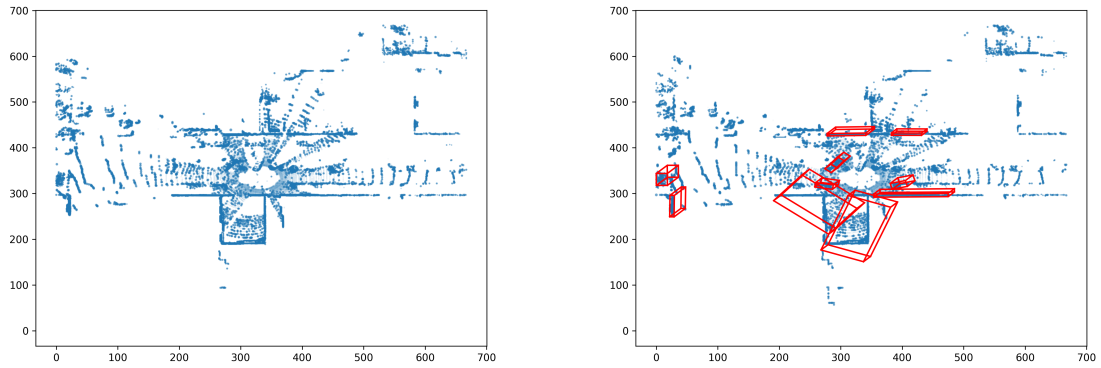
Scale each particle by measurement likelihood:

$$\beta_n = \alpha f_{z(k)|x(k)}(\widehat{z}(k)|x_p^n(k)), \text{ for } n = 1, 2, ..., N$$

where α is the normalization constant chosen such that $\sum_{n=1}^{N} \beta_n = 1$.
Re-sample N particles $x_m^n(k)$ from the scaled distribution.

5 Estimation Results

5.1 Car Tracking with LiDAR and Radar Result



(a) Single frame point cloud from Velodyne Lidar

(b) Detected objects in red bounding box

Figure 4: Vehicle Detection plots

From Figure.3, we present the raw point cloud data generated from Velodyne Lidar sensor in the real road. Then, we perform the standard connected component algorithm to detect objects, as showed in the red bounding boxes. After getting object point cloud from each bounding box, we need to classify these object categories based on 28 features extracted from point cloud, which include 4 global features and 24 local features. Each object labeled with categories can be associated with the objects in last frame, which is called data association, based on Kalman Filter prediction and Hungarian algorithm. After data association, we can use the measurement data - the center of the bounding box as object position to correct the prediction, which is called measurement update. Figure.4 shows a frame tracking of front car.

With the extracted measurement data from LiDAR and Radar, we track other vehicles with estimation technique and physics model. The estimation result is shown as below in Figure6. From the estimation result we can see that the red circles and blue circles are measurement points, green triangles are estimated positions. The estimated route of vehicle is smooth. However, cases we consider here are relatively linear. The extended Kalman Filter may have a poor performance in nonlinear cases.

5.2 Vehicle Pose Estimation Result

With Particle Filter, we present our estimation result below in Figure 7.

In the following figure, we plot the measurements points and the estimation results of the location of the autonomous vehicle in the same plot. Red crossings are measurements at some time steps. The blue line is the moving route drawn from our estimation.

Because we don't have knowledge of real states, we can only present the measurement points and compare it with our estimation to see if it's reasonable. We know that measurements come with noise so it's not very accurate. In some very noisy cases, we cannot tell the vehicle's moving track from only the measurement. With the help of physical model and the estimation technique, we present a relatively smooth and accurate moving track of the vehicle.

To decide the precision of the estimator, we will compare the estimation result at the last step with the true state and see the error. Because for particle filter, the estimation results will be different every time.

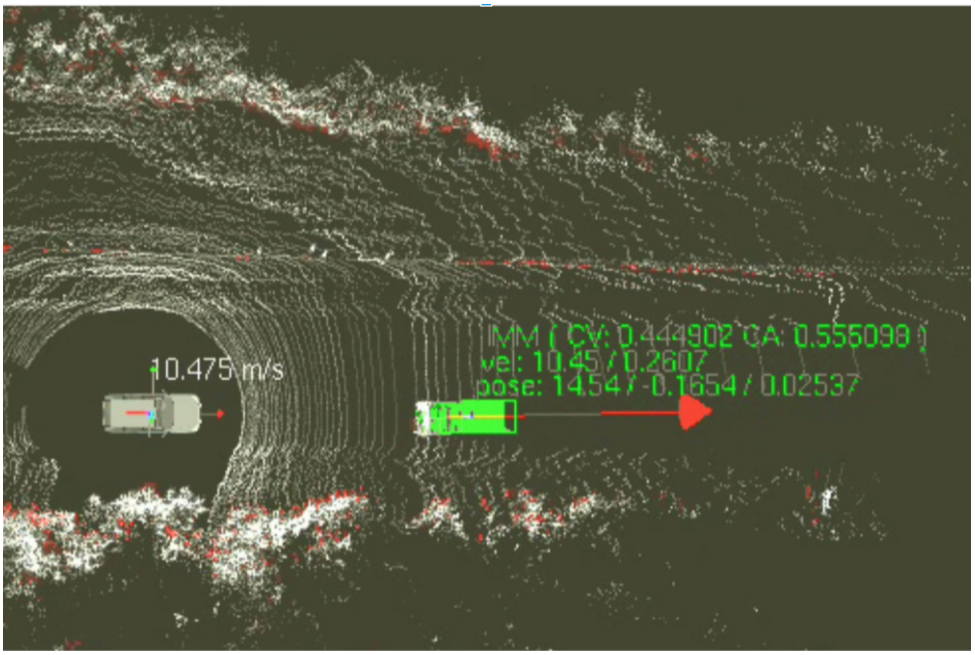


Figure 5: Tracked object (green) by Kalman filter

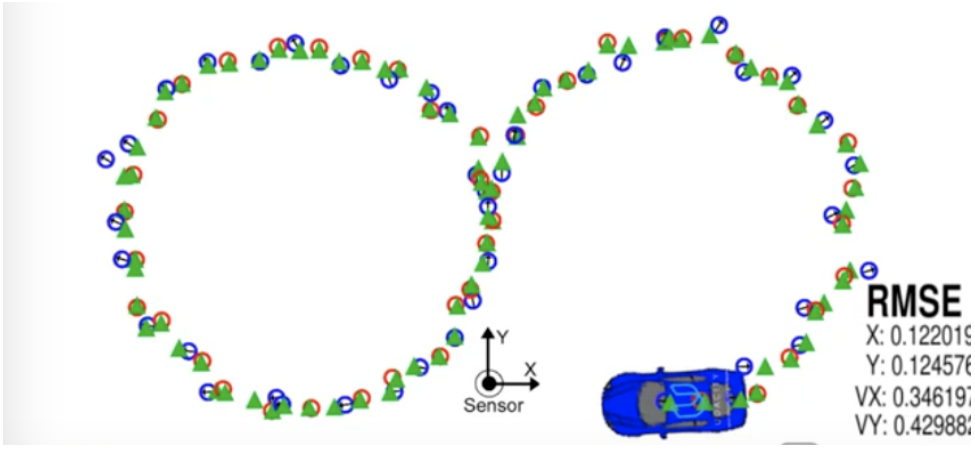
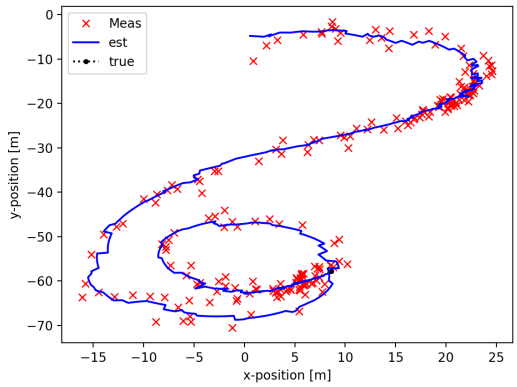


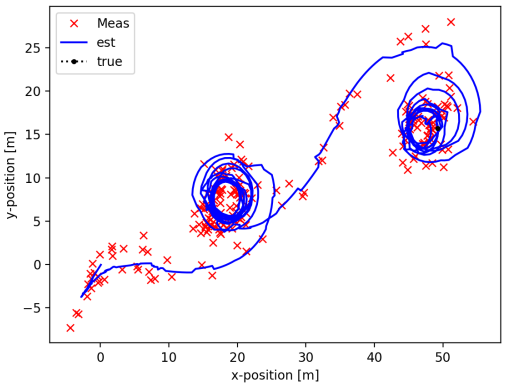
Figure 6: Estimation Results From LiDAR and Radar

So the error is not fixed after every run but will be similar.
A sample group of errors under several cases are listed below in the following table.

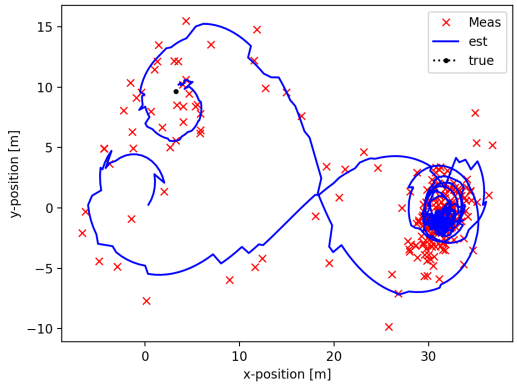
Case No.	Error X(m)	Error Y(m)	Error θ(rad)
1	-0.212	-0.311	0.022
2	0.296	0.801	0.385
3	0.322	0.640	0.215
4	-1.219	0.805	-0.292



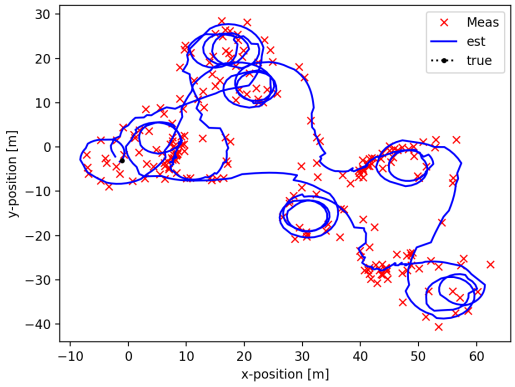
(a) Case 1



(b) Case 2



(c) Case 3



(d) Case 4

Figure 7: Vehicle Tracking plots

6 Summary and Future work

6.1 Summary

1. In the first part, we employed Kalman filter and Extended Kalman filter to track other vehicles with the measurement data from both LiDAR and Radar. Machine learning method is used to extract object's location information from cloud point data. And then Kalman filter is used to handle LiDAR measurement data and Extended Kalman Filter is used to handle Radar measurement because of the nonlinear measurement equation of Radar. Through our observation, we found that the estimation precision is greatly improved after combining measurements from two sources together. Also, because we made the assumption that the cases we focus on is close to linear cases, our estimation turns out to be good.

2. In the second part, particle filter is employed to estimate the pose of an autonomous vehicle itself. Particle Filter is another choice for estimation problem combining the thought of Monte Carlo Method and tracking method together. This usually gives better estimation result but at the cost of much higher computation amount. For a simplified autonomous vehicle model, it's tractable to employ particle filter to estimate because it doesn't have too many states. Thus we can estimate states under many very nonlinear cases with a reasonable number of particles. The key issues in implementing a particle filter include preventing the number of different particles from converging and giving a right assumption on the distribution of the noise.

3. When choosing appropriate estimation technique to use, we need to consider the assumption of noise of the system. It's always a trade off between estimation precision and computation amount. So we need to choose a suitable estimation method according to our estimation requirements. Usually Kalman filter(with its extensions) come with low computation cost but lower precision. Particle filter is with better precision and higher computation amount.

6.2 Future Work

1. In the first problem–tracking other vehicles around, we can try to extend the situation to relatively nonlinear cases and use some other methods to see the performance of different estimation techniques under these nonlinear cases.

2. In the second problem–estimation of vehicle's pose, there are some parameters in the system which should be a fixed number but we don't know for certain. We can list these parameters in the state to try to identify them. It may help to improve the accuracy of the estimation.

7 Acknowledgement

Thanks Professor Moura for excellent instruction for CE295, really benefit from this course a lot!
Thanks Bertrand for serving as GSI and holding helpful sections!
Thanks those who devoted to developing CE295 into a charming course!
Time in Berkeley passes by so quickly, no matter it's a happy or a not that satisfying experience to us, we cannot deny that we all learn a lot here. Goodbye Berkeley, will miss the campus and friends here.:)
Fiat Lux!

References

[1] Kalman Filtering: Kalman RE. A New Approach to Linear Filtering and Prediction Problems. ASME. J. Basic Eng. 1960;82(1):35-45. doi:10.1115/1.3662552.

[2] Extended Kalman Filtering: Frühwirth, R. Application of Kalman filtering to track and vertex fitting. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 262, 444–450 (1987).

[3] Unscented Kalman Filtering: Simon J. Julier, Jeffrey K. Uhlmann, "New extension of the Kalman filter to nonlinear systems", Proc. SPIE 3068, Signal Processing, Sensor Fusion, and Target Recognition VI, (28 July 1997); doi:

[4] Particle Filtering: Tutorial on Particle Filters for Online Nonlinear/NonGaussian Bayesian Tracking. Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking (2009). doi:10.1109/9780470544198.ch73

[5] Modelling: A Tutorial on Particle Filters for Online Nonlinear/NonGaussian Bayesian Tracking. Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking (2009). doi:10.1109/9780470544198.ch73

[6] LiDAR based object perception: M.Himmelsbach, A. Muller, T.Luttel and H.-J.Wunsche, LIDAR-based 3D Object Perception,Proceedings of 1st international workshop on cognition for technical systems, volume 1. 2008

[7] Bicycle Model: P. Falcone, F. Borrelli, J. Asgari, Predictive Active Steering Control for Autonomous Vehicle Systems, IEEE Transactions on Control Systems Technology, Volume. 15, Issue. 3, May 2007

[8] Mark Mueller, Advanced Control(2) Lecture Notes, 2018

[9] Bicycle picture: http://www.ebicycles.com/bicycle-tools/measure-frame

8 Appendix: Code Implementation in C++

8.1 Simulation of Moving Object

We simulate the moving object by dividing the program into initialization, prediction and update parts: The initialization part goes as follows:

```
void Tracker::ProcessMeasurement(geometry_msgs::PoseArray& msg)
{
  /*****************************************************************************
   *  Initialization
   ****************************************************************************/
  if (!is_initialized_)
  {
    // first measurement
    kf_.x_ = VectorXd(4);
    kf_.x_ << 1, 1, 1, 1;  //avoid no values in x_

    // we need to select the convoy leader vehicle here
    // we assume that the straight front and closed one is the leader vehicle
    geometry_msgs::Point point;
    point.x = numeric_limits<float>::max();
    point.y = numeric_limits<float>::max();
    geometry_msgs::Point origin;
    origin.x = 0.0;
    origin.y = 0.0;
```

```
20      float distance = euclidean_distance(point,origin);
21      int size = msg.poses.size();
22      for (int i = 0; i < size; ++i)
23      {
24        geometry_msgs::Point car_pose = msg.poses[i].position;
25        if ( (car_pose.x > 0.0) && (abs(car_pose.y) < 6.0) && (euclidean_distance(origin,
      car_pose) < distance))
26        {
27          point = car_pose;
28          distance = euclidean_distance(origin,car_pose);
29        }
30      }
31
32      // no leader vehicle associated
33      if ( distance > 1000 )
34      {
35        ROS_INFO("Convoy leader association failed at this frame\n");
36        return;
37      }
38      kf_.x_ << point.x, point.y, 0, 0; // x, y, vx, vy
39      previous_timestamp_ = msg.header.stamp.toSec(); // set current time stamp
40      // done initializing, no need to predict or update
41      is_initialized_ = true;
42      return;
43    }
```

The prediction parts go as follows:

```
1   // compute the time elapsed between the current and previous measurements
2     float dt = msg.header.stamp.toSec() - previous_timestamp_;  //  in seconds
3     previous_timestamp_ = msg.header.stamp.toSec();
4
5     float dt_2 = dt * dt;
6     float dt_3 = dt_2 * dt;
7     float dt_4 = dt_3 * dt;
8
9     // Modify the F matrix so that the time is integrated, from 1 to specific dt (see line 51)
10    kf_.F_(0, 2) = dt;
11    kf_.F_(1, 3) = dt;
12
13    // set model noises (here is the acceleration)
14    float noise_ax = 9;
15    float noise_ay = 9;
16
17    //set the process covariance matrix Q (Q = X * a * X_T)
18    kf_.Q_ = MatrixXd(4, 4);
19    kf_.Q_ << dt_4/4*noise_ax,   0,               dt_3/2*noise_ax,  0,
20              0,                 dt_4/4*noise_ay,  0,               dt_3/2*noise_ay,
21              dt_3/2*noise_ax,   0,               dt_2*noise_ax,    0,
22              0,                 dt_3/2*noise_ay,  0,               dt_2*noise_ay;
23
24    kf_.Predict();
```

The update part goes as follows:

```
1   // Lidar updates
2     kf_.H_ = H_lidar_;
3     kf_.R_ = R_lidar_;
4
5     // we need to select the convoy leader vehicle here
6     // by gated nearest neighbor data association but reasonable distance to avoid associate
        other vihicle
7     geometry_msgs::Point point;
8     point.x = numeric_limits<float>::max();
9     point.y = numeric_limits<float>::max();
```

```
10
11   geometry_msgs::Point car_pred;
12   car_pred.x = kf_.x_[0];
13   car_pred.y = kf_.x_[1];
14   float distance = euclidean_distance(point,car_pred);
15   int size = msg.poses.size();
16   for (int i = 0; i < size; ++i)
17   {
18     geometry_msgs::Point car_pose = msg.poses[i].position;
19     if ( (car_pose.x > 0.0) && (euclidean_distance(car_pred,car_pose) < distance))
20     {
21       point = car_pose;
22       distance = euclidean_distance(car_pred,car_pose);
23     }
24   }
```

8.2 Estimation Code

Then we would employ the extended Kalman Filter to estimate the therotical position of the object:

```
1  void KalmanFilter::Init(VectorXd &x_in, MatrixXd &P_in, MatrixXd &F_in, MatrixXd &H_in,
       MatrixXd &R_in, MatrixXd &Q_in)
2  {
3    x_ = x_in;
4    P_ = P_in;
5    F_ = F_in;
6    H_ = H_in;
7    R_ = R_in;
8    Q_ = Q_in;
9  }
10
11 void KalmanFilter::Predict()
12 {
13   /**
14    * predict the state x_k|k-1 and state covariance matrix P_k|k-1
15    */
16   x_ = F_ * x_;
17   MatrixXd F_t = F_.transpose();
18   P_ = F_ * P_ * F_t + Q_;
19 }
20
21 void KalmanFilter::Update(const VectorXd &z) {
22   /**
23    * measurement update to correct the prediction
24    * z is the true measurement
25    */
26   MatrixXd z_pred = H_ * x_;
27   MatrixXd S_ = H_ * P_ * H_.transpose()+R_;
28   MatrixXd K_ = P_ * H_.transpose() * S_.inverse();
29
30   //estimate
31   x_ = x_ + K_ * (z - z_pred);
32   P_ = P_ - K_ * (H_ * P_ * H_.transpose() + R_) * K_.transpose();
33 }
```

The following is the code for particle filter in tracking autonomous vehicle which is implemented with python.

```
1 import numpy as np
2 import scipy as sp
3 #NO OTHER IMPORTS ALLOWED (However, you're allowed to import e.g. scipy.linalg)
```

```
4
5  def estInitialize():
6      # Fill in whatever initialization you'd like here. This function generates
7      # the internal state of the estimator at time 0. You may do whatever you
8      # like here, but you must return something that is in the format as may be
9      # used by your run() function.
10     #
11
12     #we make the interal state a list, with the first three elements the position
13     # x, y; the angle theta; and our favourite color.
14
15     nparticles = 1000
16     x_init = np.random.normal(0, 3, (nparticles))
17     y_init = np.random.normal(0, 3, (nparticles))
18     theta_init = np.random.normal(np.pi / 4, 0.5, (nparticles))
19     particles = np.column_stack([x_init, y_init, theta_init])
20     w_mu = np.array([[0], [0]])
21     w_sig = np.array([[1, 0], [0, 1.7]])
22
23     color = 'green'
24     # note that there is *absolutely no prescribed format* for this internal state.
25     # You can put in it whatever you like. Probably, you'll want to keep the position
26     # and angle, and probably you'll remove the color.
27     internalState = [nparticles,
28                      particles,
29                      color,
30                      w_mu,
31                      w_sig
32                      ]
33     return internalState
34
35 import numpy as np
36 import scipy as sp
37 #NO OTHER IMPORTS ALLOWED (However, you're allowed to import e.g. scipy.linalg)
38
39 def estRun(time, dt, internalStateIn, steeringAngle, pedalSpeed, measurement):
40     # In this function you implement your estimator. The function arguments
41     # are:
42     #  time: current time in [s]
43     #  dt: current time step [s]
44     #  internalStateIn: the estimator internal state, definition up to you.
45     #  steeringAngle: the steering angle of the bike, gamma, [rad]
46     #  pedalSpeed: the rotational speed of the pedal, omega, [rad/s]
47     #  measurement: the position measurement valid at the current time step
48     #
49     # Note: the measurement is a 2D vector, of x-y position measurement.
50     #  The measurement sensor may fail to return data, in which case the
51     #  measurement is given as NaN (not a number).
52     #
53     # The function has four outputs:
54     #  est_x: your current best estimate for the bicycle's x-position
55     #  est_y: your current best estimate for the bicycle's y-position
56     #  est_theta: your current best estimate for the bicycle's rotation theta
57     #  internalState: the estimator's internal state, in a format that can be understood by
       the next call to this function
58
59     # Example code only, you'll want to heavily modify this.
60     # this internal state needs to correspond to your init function:
61     print time
62     nparticles = internalStateIn[0]
63     particles = internalStateIn[1]
64     color = internalStateIn[2]
65     w_mu = internalStateIn[3]
```

```
66      w_sig = internalStateIn[4]
67
68      (x_m,y_m) = measurement
69      if np.isnan(x_m):
70          #(here to be fixed about dt)
71          r = 0.425
72          B = 0.8
73          v_r = np.random.normal(0, 0.05, (nparticles, 1))
74          v_w = np.random.normal(0, 0, (nparticles, 1))
75          v_b = np.random.normal(0, 0.1, (nparticles, 1))
76          v_gamma = np.random.normal(0, 0, (nparticles, 1))
77          x_p = particles[:, 0][np.newaxis]
78          x_p = x_p.T
79          y_p = particles[:, 1][np.newaxis]
80          y_p = y_p.T
81          theta_p = particles[:, 2][np.newaxis]
82          theta_p = theta_p.T
83          x_new = x_p + 5.0 * r * pedalSpeed * dt * (1.0 + v_r) * (np.cos(theta_p)) * (1.0 + v_w
    )
84          y_new = y_p + 5.0 * r * pedalSpeed * dt * (1.0 + v_r) * (np.sin(theta_p)) * (1.0 + v_w
    )
85          theta_new = theta_p + 5.0 * r / B * pedalSpeed * dt * (1.0 + v_r) * (np.tan(
    steeringAngle * (1.0 + v_gamma))) * (1.0 + v_w) / (1.0 + v_b)
86          particles = np.column_stack([x_new,y_new,theta_new])
87      else:
88          # (here to be fixed about dt)
89          r = 0.425
90          B = 0.8
91          v_r = np.random.normal(0, 0.05, (nparticles, 1))
92          v_w = np.random.normal(0, 0, (nparticles, 1))
93          v_b = np.random.normal(0, 0.1, (nparticles, 1))
94          v_gamma = np.random.normal(0, 0, (nparticles, 1))
95          x_p = particles[:, 0][np.newaxis]
96          x_p = x_p.T
97          y_p = particles[:, 1][np.newaxis]
98          y_p = y_p.T
99          theta_p = particles[:, 2][np.newaxis]
100         theta_p = theta_p.T
101         x_new = x_p + 5.0 * r * pedalSpeed * dt * (1.0 + v_r) * (np.cos(theta_p)) * (1.0 + v_w
    )
102         y_new = y_p + 5.0 * r * pedalSpeed * dt * (1.0 + v_r) * (np.sin(theta_p)) * (1.0 + v_w
    )
103         theta_new = theta_p + 5.0 * r / B * pedalSpeed * dt * (1.0 + v_r) * (np.tan(
    steeringAngle * (1.0 + v_gamma))) * (
104         1.0 + v_w) / (1.0 + v_b)
105         particles = np.column_stack([x_new, y_new, theta_new])
106
107         f_z = np.zeros((len(particles[:, 0]), 1))
108
109         w_1 = x_m - x_new - 1.0 / 2.0 * B * np.cos(theta_new)
110         w_2 = y_m - y_new - 1.0 / 2.0 * B * np.sin(theta_new)
111         f_x1 = 1.0 / (2.0 * np.pi * 1.089 * 1.089) * np.exp(-(w_1 - 0.0) * (w_1 - 0.0) / 2.0 /
     (1.089 * 1.089))
112         f_x2 = 1.0 / (2.0 * np.pi * 2.99 * 2.99) * np.exp(-(w_2 - 0.0) * (w_2 - 0.0) / 2.0 /
    (2.99 * 2.99))
113         f_z = f_x1 * f_x2
114         # to be fixed here
115         alpha = 1.0 / np.sum(f_z[:])
116         f_z = f_z * alpha
117         cdf = np.cumsum(f_z)
118
119         # resample distribution samples
120         old_p = particles.copy()
```

```
121         particles = np.array([old_p[np.argwhere(cdf > np.random.uniform())[0, 0]] for i in
        range(len(f_z))])
122         # adding noise to re-sampled particles to avoid it from converging(keep the amount of
        different particles)
123         v_xy = np.random.normal(0, 0.3, (nparticles, 2))
124         v_phi = np.random.normal(0, 0.1, (nparticles, 1))
125         v_sample = np.concatenate((v_xy, v_phi), axis=1)
126         particles = particles + v_sample
127
128     x = np.mean(particles[:, 0])
129     y = np.mean(particles[:, 1])
130     theta = np.mean(particles[:, 2])
131     return x, y, theta, internalState
132
133 import numpy as np
134 import matplotlib.pyplot as plt
135 from estRun import estRun
136 from estInitialize import estInitialize
137
138 #provide the index of the experimental run you would like to use.
139 # Note that using "0" means that you will load the measurement calibration data.
140 experimentalRun = 04
141
142 print('Loading the data file #', experimentalRun)
143 experimentalData = np.genfromtxt ('data/run_{0:03d}.csv'.format(experimentalRun), delimiter
        =',')
144
145
146 print('Running the initialization')
147 internalState = estInitialize()
148
149 numDataPoints = experimentalData.shape[0]
150
151 #Here we will store the estimated position and orientation, for later plotting:
152 estimatedPosition_x = np.zeros([numDataPoints,])
153 estimatedPosition_y = np.zeros([numDataPoints,])
154 estimatedAngle = np.zeros([numDataPoints,])
155
156 print('Running the system')
157 dt = experimentalData[1,0] - experimentalData[0,0]
158 for k in range(numDataPoints):
159     t = experimentalData[k,0]
160     gamma = experimentalData[k,1]
161     omega = experimentalData[k,2]
162     measx = experimentalData[k,3]
163     measy = experimentalData[k,4]
164
165     #run the estimator:
166     x, y, theta, internalState = estRun(t, dt, internalState, gamma, omega, (measx, measy))
167
168     #keep track:
169     estimatedPosition_x[k] = x
170     estimatedPosition_y[k] = y
171     estimatedAngle[k] = theta
172
173
174 print('Done running')
175 #make sure the angle is in [-pi,pi]
176 estimatedAngle = np.mod(estimatedAngle+np.pi,2*np.pi)-np.pi
177
178 posErr_x = estimatedPosition_x - experimentalData[:,5]
179 posErr_y = estimatedPosition_y - experimentalData[:,6]
180 angErr   = np.mod(estimatedAngle - experimentalData[:,7]+np.pi,2*np.pi)-np.pi
```

```
181
182 print('Final error: ')
183 print('   pos x =',posErr_x[-1],'m')
184 print('   pos y =',posErr_y[-1],'m')
185 print('   angle =',angErr[-1],'rad')
186
187 ax = np.sum(np.abs(posErr_x))/numDataPoints
188 ay = np.sum(np.abs(posErr_y))/numDataPoints
189 ath = np.sum(np.abs(angErr))/numDataPoints
190 score = ax + ay + ath
191 if not np.isnan(score):
192     #this is for evaluation by the instructors
193     print('average error:')
194
195     print('   pos x =', ax, 'm')
196     print('   pos y =', ay, 'm')
197     print('   angle =', ath, 'rad')
198
199     #our scalar score.
200     print('average score:',score)
201
202
203 print('Generating plots')
204
205 figTopView, axTopView = plt.subplots(1, 1)
206 axTopView.plot(experimentalData[:,3], experimentalData[:,4], 'rx', label='Meas')
207 axTopView.plot(estimatedPosition_x, estimatedPosition_y, 'b-', label='est')
208 axTopView.plot(experimentalData[:,5], experimentalData[:,6], 'k:.', label='true')
209 axTopView.legend()
210 axTopView.set_xlabel('x-position [m]')
211 axTopView.set_ylabel('y-position [m]')
212
213 figHist, axHist = plt.subplots(5, 1, sharex=True)
214 axHist[0].plot(experimentalData[:,0], experimentalData[:,5], 'k:.', label='true')
215 axHist[0].plot(experimentalData[:,0], experimentalData[:,3], 'rx', label='Meas')
216 axHist[0].plot(experimentalData[:,0], estimatedPosition_x, 'b-', label='est')
217
218 axHist[1].plot(experimentalData[:,0], experimentalData[:,6], 'k:.', label='true')
219 axHist[1].plot(experimentalData[:,0], experimentalData[:,4], 'rx', label='Meas')
220 axHist[1].plot(experimentalData[:,0], estimatedPosition_y, 'b-', label='est')
221
222 axHist[2].plot(experimentalData[:,0], experimentalData[:,7], 'k:.', label='true')
223 axHist[2].plot(experimentalData[:,0], estimatedAngle, 'b-', label='est')
224
225 axHist[3].plot(experimentalData[:,0], experimentalData[:,1], 'g-', label='m')
226 axHist[4].plot(experimentalData[:,0], experimentalData[:,2], 'g-', label='m')
227
228 axHist[0].legend()
229
230 axHist[-1].set_xlabel('Time [s]')
231 axHist[0].set_ylabel('Position x [m]')
232 axHist[1].set_ylabel('Position y [m]')
233 axHist[2].set_ylabel('Angle theta [rad]')
234 axHist[3].set_ylabel('Steering angle gamma [rad]')
235 axHist[4].set_ylabel('Pedal speed omega [rad/s]')
236
237 print('Done')
238 plt.show()
```

Nanjing Metro traveler entrance modeling and minimization of system-wide wait time by optimizing train distribution

Moon Ki Jung, Carlin Liao, Henry Teng, Johnny Wu

Abstract

Subway systems face two competing demands: from the passenger's perspective minimizing travel time is paramount, while the operator is most concerned with reducing cost. In this paper, we will implement and refine our methods on the Nanjing Metro system. Using trip data harvested by smartcards, we create a viable mathematical model of system-wide passenger wait time for the metro's governing board to best optimize its limited resources (trains) to meet both passenger and operational needs. Using the distribution of passengers' station entrances between trains also developed in this paper, our optimization problem also finds the ideal allocation of trains along multiple subway lines at any time of day for real or simulated demand schedules in order to minimize total passenger wait times with the resources available.

Introduction

Motivation and Background

As demonstrated in the first lecture of CE 295, many cities are growing rapidly, and Nanjing is no different. The burden of effectively mobilizing that ever-growing population, the tenth largest in China despite being close to the size of New York City, falls upon transit systems like the Nanjing Metro, which served 977.4 million passengers in 2017. To optimize the timetable of a metro system, we can take either the operator's perspective or the user's perspective to find widely different consequences. From the operator's point of view, an ideal system would use the least amount of resources while still getting travelers to their destinations regardless of how long that takes, while for a passenger, ideal would mean getting from origin to destination in the least amount of time regardless of operations cost.

Therefore it is necessary to consider how to balance these two sets of priorities. Similar to an approach taken from literature [1], this may be accomplished by minimizing the total travel time while imposing operator constraints. In deciding which metro system to analyze, our group was able to obtain two months' worth of smart-card data from the Nanjing Metro system. This smartcard data contains entries of every swipe-in and swipe-out of passengers

at every station on every core line in the Metro extant during the months of March and April 2015. With such a high fidelity data set, our group decided to investigate how we could model the system-wide wait time of a metro system and then how that wait time could be minimized while satisfying operational constraints.

Considering the large size of both the Nanjing Metro and our data set, our group anticipated challenges with processing and extracting useful information from our data set that would best describe the passenger demands at stations in the system. In addition, transit systems are inherently difficult to model and analyze, due to the variability in behavior of their passengers. Many operational standards and protocols for the Metro, such as speed limits, line constraints, train car specs, and even a fixed timetable were not made available to our group, which required us to make several reasonable assumptions in developing a mathematical model for our problem.

Originally our group had intended to implement our analysis on a dataset sourced from the Shenzhen Metro by way of the Tsinghua-Berkeley Shenzhen Institute. Unfortunately, due to complications with access and confidentiality, we were not able to secure the dataset in time. Instead we were able to secure a near-identically featured dataset from the Nanjing Metro system courtesy of Dounan Tang in Prof. Sengupta's group which allowed us to complete our analysis as planned.

Focus of this Study

Our project will develop a mathematical model of the system-wide wait time across the Nanjing Metro, which we will use to optimize the distribution of trains across the system. In doing so, we perform exploratory data analysis to investigate various assumptions of our mathematical model, including the distribution of passenger entrances at origin stations.

Literature review

Thanks to the usage of smart card technology for public transit applications, there exists an abundance of data upon which a myriad of analyses can be conducted. Many studies, [2], focus on discerning passenger route choice and behavioral aspects from this data. However, our team was more interested on using reducing wait-time across a system—a topic outside the scope of these papers. Nonetheless, the methods of data processing and background on transportation planning topics presented in these studies were invaluable in our project. Select papers [1] [3] [4] investigate optimizing timetables for various subway systems, and it is from these papers that we understood the context of the problem we wished to investigate, namely developing improved timetables that lead to reduced passenger wait times. However, these papers failed to account for the effects of limited resources on the operator's side, namely the number of trains available for dispatch. This then became our team's ultimate trajectory for the project.

However, with this question posed, many additional issues arose, specifically in the assumptions we could make of our system, and how we could formulate our model. To this end, we reached out to papers for techniques used by other researchers. For instance, we adopted how a study [1] focused on only a subset of a system (select lines) in order to simplify analysis. Select formulations (e.g. subdividing total travel time into its constituent

time components) and simplifications (e.g. uniform passenger arrival) used in papers [4] [5] also gave inspiration to many of our assumptions and the chosen form of our model. The exact details of our assumptions, formulation, and implementation are discussed in subsequent sections of this report.

Technical Description

Due to the substantial scope of this project, the technical description consists of (1) a description of our data set, (2) key assumptions & exploratory data analysis of our data set, (3) the optimization problem formulation, (4) code implementation, and (5) optimization results.

Dataset

The source dataset is a record of all smartcard trips made in the Nanjing subway system from January through April 2015, which for this project we subsetted to the last week of April. Originally provided as .mdb files with simplified Chinese headers, we used Microsoft Access and Python to convert the tables from proprietary Microsoft databases to csv files for easier manipulation and translated the table headers by hand.

The data is a collection of all tap-ins (entrances) and tap-outs (exits) by smartcards in the subway system over the aforementioned four months in time. Note that this model will thus not cover travelers using paper fares and as such assumes that trips made with smartcards are an accurate representation of all trips in the system.

The headers of the dataset after translation are:

```
Card_Id    Card_Id_Long  Card_Type  Fare  Time  Entry_Or_Exit  Line_Id
StationId  Device_Id
```

(We made an effort to match the initial translations used by Tang in the CEE 290I homework, explaining the inconsistency in underscores versus camel case.)

Please refer to the section on **Code Implementation** for additional details on our data processing.

Note that data .mdb and .csv files have not been included with this report due to their large sizes. Contact us if you would like the original files to replicate our results.

Key Assumptions & Exploratory Data Analysis

For the purposes of our train distribution optimization problem, we have made three key assumptions:

1. All trains have enough capacity to meet demand
2. Headways are constant on all lines in both directions during the time interval of our analysis

3. Travelers enter stations to take the train in the same way regardless of how long or short the headway between trains is (that is to say, they will always arrive in a uniform distribution between trains, with the only variable affected by the headway being the total number of people who arrive in each period)

None of these assumptions are made spuriously. The first arises from trains being the limiting factor to better service, rather than train cars, so the analysis we make optimizes for train frequency and not train length. The second comes from knowledge of scheduled train frequency in the system, and the last from a simplifying assumption common in transportation planning where passengers are assumed to arrive in a uniform distribution [5]. Even so, given our dataset, we have the capability to examine the last two assumptions, which we will do so in our exploratory data analysis.

For the purposes of this analysis, we have chosen to only examine terminal stations. Due to the nature of the dataset it is not immediately clear what direction train a traveler intends to take except at terminal stations where there is only one option. At other stations, inferring travelers' route direction from their exit station would be necessary. This is done during the optimization portion of this report, but is not necessary for the examination of headway consistency and traveler entrance distribution.

Headway Verification

In order to verify the scheduled three-minute interval between trains on the Line 1's northern portion, we examine a histogram of travelers exiting the line's northern terminus over small time intervals. If trains were arriving every three minutes, the histogram should exhibit corresponding spikes in exit counts approximately every 3 minutes as well as most travelers exit the station as soon as the train arrives. In fact, this periodicity is exactly what we see in Figure 1, empirically verifying that the headway on this line is three minutes long.

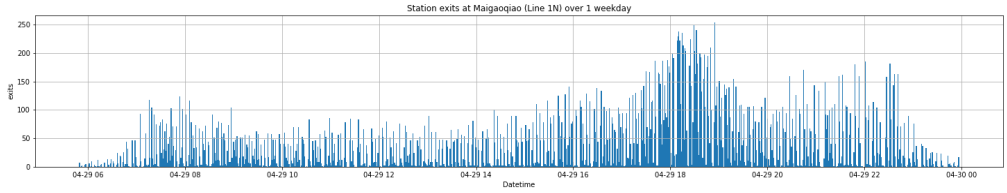


Figure 1: Histogram of travelers exiting the Line 1 north terminus over a weekday

Traveler Station Entrance Distribution

In line with the histogram of station exits, we also examined the same for station entrances. As shown in Figure 2, the periodicity is not as evident as with station exits, although there still do seem to be spikes in entrance count that could suggest some travelers are adjusting their arrival times to correspond with train arrivals, which would disprove the uniform assumption.

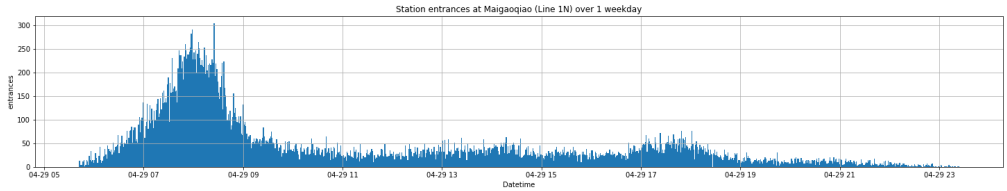


Figure 2: Histogram of travelers entering the Line 1 north terminus over a weekday

In order to evaluate this, we collapsed the entire day of arrivals into a single three-minute (180-second) period to correspond to their arrival times relative to the time until the next train, and normalized the resulting histogram of arrivals per relative second so all bars would sum to one (as in a probability distribution). Attempts to fit the resulting graph with known probability distributions (seen in Figure 3) showed that the uniform distribution gave the lowest sum squared error.

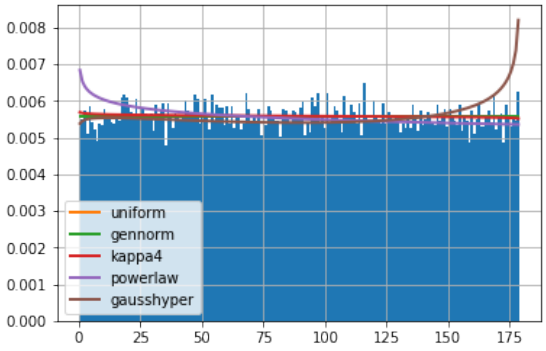


Figure 3: Normalized histogram of station entrances per second since last train and corresponding fits

Normalization of the data over each 3-minute interval yielded very similar results (as in Figure 4), so we conclude that the uniform distribution of station entrances is a realistic assumption for Nanjing travelers. This conclusion parallels those of Luethi, who further expands on passenger arrival modelling, [5].

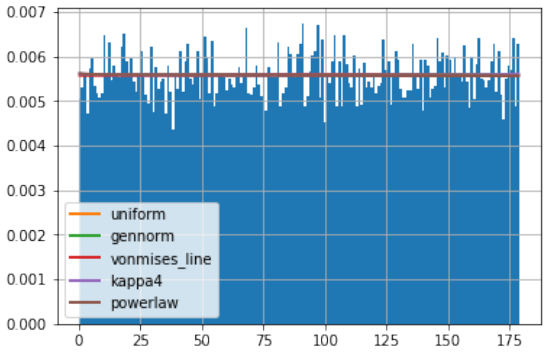


Figure 4: Normalized cumulative distribution of station entrances per each 3-minute period and corresponding fits

Train Distribution Optimization Problem Formulation

In developing our optimization problem, we will divide our formulation into two levels: station-scale and multi-line scale. In doing so, we address the following points:

- Arrival behavior of passengers at a station in-between trains
- Relationship between headway on a line and the number of trains on a line
- Feedback of changing headway on the arrival of passengers at the station

We also carry forth the same three key assumptions discussed in the previous section.

Station-Scale Formulation

In order to calculate the total wait time for one duration of time between train departure and arrival, we must account for all passengers on a platform, each of their arrival times, and each passenger's corresponding wait time. We first note that patrons are assumed to arrive at the station uniformly over time as discussed in the prior section. We define a passenger arrival function which will describe this rate of passenger arrivals. We then define a wait function, a linearly decreasing function, in which the first passenger on the platform waits the entire headway of the train, while the last passenger waits the least amount of time. We define headway to be the span of time between trains (which in turn becomes the time between train arrivals at a given station).

To obtain the total wait time on a single platform, we then multiply our passenger arrival function by the wait function, and scale by the total number of passengers on the platform during the time interval. This formulation is visually expressed in the following diagram.

Figure 5: Station-scale formulation

We define the passenger arrival function to be a_{HW}, and the wait function to be w_{HW}, with subscript HW denoting these two functions' dependency on headway.

It is interesting to note that our passenger arrival function is equivalent to probability density function of passenger arrivals, defined by a uniform distribution.

Our station-scale formulation is then defined as:

$$a_{HW} * w_{HW} * n \tag{1}$$

where n is defined to be the total number of people on that platform when the incoming train arrives. This value is extracted from our smart card data and is described ***Code Implementation***, an upcoming subsection of this report.

To illustrate how passenger arrivals change at a station over time, we provide the following diagram. Note that we discretize time into increments equal to the headway on that line (recall we assume that headways are constant along a line).

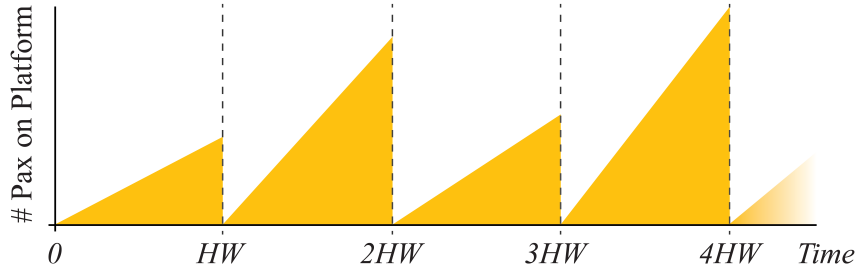


Figure 6: Program flow chart

Note how the assumption that train cars have infinite capacity allows us to reset our cumulative number of passengers on a platform whenever a train arrives at the platform.

We index each increment of time by $i \in 0, .., T$ where T is the final time duration we are interested in. To find the total wait time across all time increments i, we merely add T instances of Equation 1.

Multi-Line Formulation

Expanding Equation 1 beyond to multiple lines and stations, we note that a_{HW} and w_{HW} do not change along a line, as we assume people will arrive at all stations along a line in the

same fashion, and these two functions are merely based on headway, which is assumed to be constant along a line. In addition, we consider the bi-directionality of each line. in allocating trains, both directions of a line share the same trains. We remedy this by adding another index $d \in \{1, 2\}$, respectively denoting the forwards or backwards direction. Therefore, our total wait time for a given line l, direction d, across all stations j, for all time increments i, can be written as:

$$TWT = \sum_{l \in \{1,\ldots,L\}} \sum_{d \in \{1,2\}} \sum_{j \in \{1,\ldots,J_l\}} \sum_{i \in \{0,\ldots,T\}} a_{HW_l} * w_{HW_l} * n_{l,d,j,i} \tag{2}$$

As defined for a system:

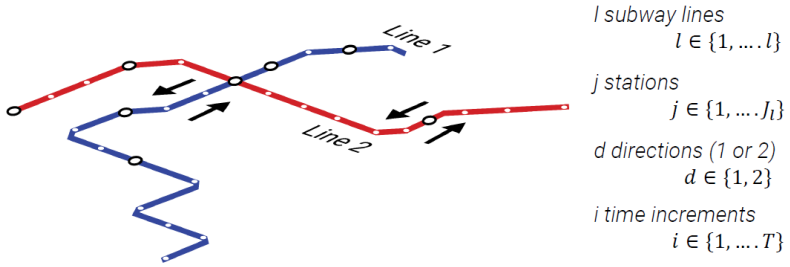


Figure 7: Schematic of multi-line system

See that a_{HW_l} and w_{HW_l} do not differ with direction of train, on one line. Special care must be taken to calculate the appropriate $n_{l,d,j,i}$ term in the expansion, as we must organize our data with respect to line, direction of journey, station, and time. With TWT fully defined, we seek to minimize total wait time across the entire system with reference to HW_l, which we understand influences every term in our cost function TWT. Again, we will discuss how our n term is affected by headway in ***Code Implementation***.

Our entire objective, however, is to optimally distribute a total number of trains. We therefore must formulate a relationship between HW_l and the number of trains on a line. For simplicity, we use the relation:

$$HW_l = \frac{s_l}{n_{trains_l} * v_{train_l}} \tag{3}$$

We divide the total length of a line by the speed of the trains on the line and the number of trains on that line. We have assumed that all trains travel at the same speed on a line. v_{train_l} is approximated from a timetable of first train arrivals at each station. Therefore, instead of optimizing across HW_l, we are actually optimizing across n_{trains_l}. With this relationship, we can complete our optimization problem:

$$\underset{n_{trains-l}}{\text{minimize}} \quad TWT = \sum_{l\in\{1,...,L\}} \sum_{d\in\{1,2\}} \sum_{j\in\{1,...,J_l\}} \sum_{i\in\{0,...,T\}} a_{HW_l} * w_{HW_l} * n_{l,d,j,i}$$

$$\text{subject to} \quad HW_l = \frac{s_l}{n_{trains_l} * v_{train_l}},\ l \in \{1,...,L\} \ \textbf{Headway- \# Trains Relation}$$

$$\sum_{l\in\{1,..,L\}} n_{train_l} \leq n_{train_{total}},\ l \in \{1,...,L\} \ \textbf{Total \# of trains}$$

$$n_{trains_l} > 0,\ l \in \{1,...,L\} \ \textbf{Non-negative, non-zero \# of trains on any line}$$

$$HW_l^{min} \leq HW_l \leq HW_l^{max},\ l \in \{1,...,L\} \ \textbf{Min. \& max. wait times on a line}$$

$$n_{trains_l} \in \mathbb{Z} \ \textbf{Integer \# of trains}$$

$n_{l,d,j,i}$ is from our smartcard data, s_l is found from a map of the system, v_{train_l} is calculated from a timetable, $n_{train_{total}}$ is the total number of trains we make available to the system, and HW_l^{min} & HW_l^{max} are limits on the headway that any line will experience, constraints imposed by us to effectively enforce speed limits and a minimum level of service, respectively.

We note that this optimization problem is an integer problem, as our decision variables, n_{trains_l} are restricted to whole numbers of trains. To solve this problem, we use a brute-force search method in which we will calculate TWT for all possible combinations of n_{trains_l} across l lines, and select the combination that arrives at the minimum TWT.

Remark on Feedback of headway on arrival of passengers

We must finally consider that if we change the value of n_{trains_l}, HW_l will also change (by way of Equation 4), which will then affect every term in TWT. In accounting for this dependency, we make our third key assumption (refer to the section, ***Key Assumptions & Exploratory Data Analysis***).

This is a crucial assumption as we have no means of modeling passenger behavior with merely our smartcard data (which does not have any information on the corresponding headways during those times).

With this assumption, we will merely discretize our data evenly with the new HW_l, and calculate a_{HW_l}, w_{HW_l} and $n_{l,d,j,i}$ based on the new increments of data. This idea is detailed further in ***Code Implementation***.

Code Implementation

To improve understanding of our code implementation, please refer to the Program Flowchart, Figure **??** in the **Appendix**.

Data Processing

The CSV data provided to us came in the form of rows of metro card swipe in/out in one day. The columns consisted of the a short passenger ID, a long Passenger ID (includes demographic information), card type, fare, time stamp of swipe in/out, entry (1) or exit (2), line, station, and device ID. In order to determine TWT, we had to write a program that

would calculate our $n_{l,d,j,i}$ on each station, on a specific line, in a specific direction, and at a specific time period. The inputs to the program would be the headways of the lines we were interested in (in this case, line 1 and line 2).

Extraction of Origin-Destination (OD) Pairs

The objective of the program is to calculate the total wait time of passengers on each line, given HW_l inputs to those lines (1 and 2 for example). As we were only focused on trains operating on lines 1 and 2, we exclude entries of passengers entering/exiting stations on lines 1 and 2 to either transfer into a different line or transferred into lines 1 and 2 from a different line. This specification was made because focusing on passengers travelling exclusively on line 1 or line 2 would allow us to readily determine the direction they would be commuting on that line by comparing an OD pair's station of entry and exit.

All data entries were ordered by time of card swipe entry/exit. Then, the data entries were filtered and parsed by line and entry or exit. We then created OD pairs by matching the occurrences of a unique passenger ID in the data. If a pair was successfully found, then the station ID at entry and exit were compared to determine which direction on that line the passenger was travelling in.

Computation of TWT Cost Function

Our program at this point generates a (2-by-N) cell array, matching the N number of stations on a specific line. The 2 rows indicate the direction of travel along that line. Each cell would contain all the entries of passengers (ordered by time) who entered that station and made a OD pair in direction 1 or 2. Given an input HW_l (let's say 6 mins on Line 1), the code iterates across each of the (2-by-N) cells, and calculates the number of entries that arrive in successive 6 minute increments in each cell until a time horizon T, thus finding each $n_{l,d,j,i}$. Assuming a uniform passenger arrival distribution, the code multiples each $n_{l,d,j,i}$ by $\frac{HW_l}{2}$, the average wait time at that station. Summing all $n * \frac{HW_l}{2}$ terms at each station, and then summing across all stations gives us the wait times on a line. This method is repeated for other lines, and then the wait times are summed across all lines, obtaining TWT

Exhaustive Search

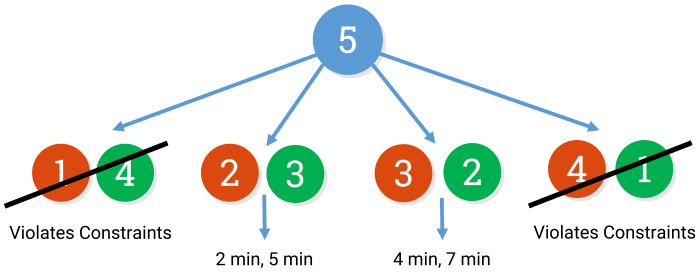


Figure 8: Exhaustive Search Method: Creating viable combinations of train distribution along two train lines

Recall the program calculating TWT does so for inputs HW_l on desired lines l. Implementing the exhaustive search method, we generate all combinations of N trains available for dispatch across k lines, which are directly related to HW by the headway to number of trains equality (see *Multi-Line Formulation*). Combinations that violate any constraints are ejected, and a TWT is generated for each viable HW_k. It is important to note that we enforce $\sum n_{trains_l,2} = n_{trains_{total}}$ instead of the inequality $\sum n_{trains_l} \leq n_{trains_{total}}$. This is because having the maximum number of trains in the system would always result in the minimal total time wait in the system, as more trains in the system always equated to lower wait times for passengers.

Optimization Results and Demand Simulation

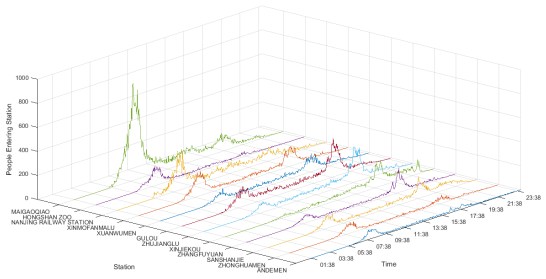
We can now execute our problem using the Nanjing smartcard data.

For all subsequent analyses, let us consider $n_{train_{total}} = 50$, to be distributed between lines 1 and 2. To visualize these two lines spatially, please refer to a map of the Nanjing Metro system, included in the Appendix.

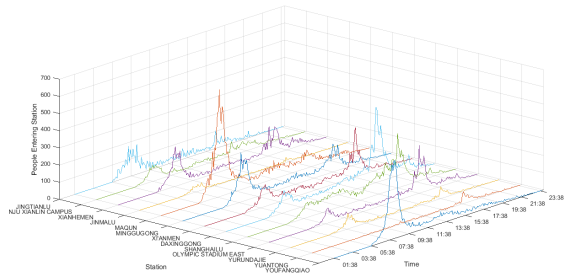
We recall that TWT can be calculated across any time horizon T. We choose to optimize train distribution across a daily horizon and a 3-hour horizon.

Daily Analysis

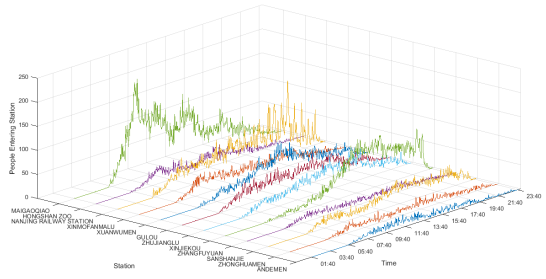
In this analysis, we compute TWT across an entire day. We visualize the passenger demand across a Sunday and Monday with the following plots for lines 1 and 2.



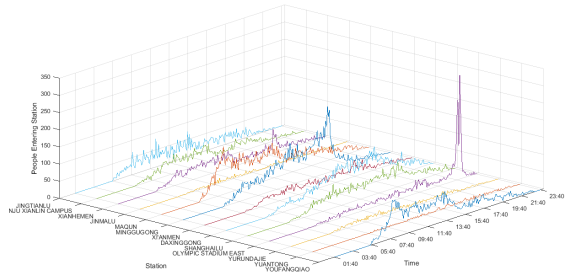
(a) Monday, Line 1



(b) Monday, Line 2



(c) Sunday, Line 1



(d) Sunday, Line 2

Figure 9: Passenger demand, Monday & Sunday, Lines 1 & 2, North to South

On both lines, the Monday passenger demands exhibits a clear workday pattern, with two peaks corresponding to morning and evening rush hours, while the Sunday passenger demands lack any clear pattern. With this in mind, we present our optimization results:

Table 1: Results of Daily Analysis, $n_{train_{total}} = 50$

Monday			**Sunday**		
$n_{train_{L1}}$	$n_{train_{L2}}$	TWT (min)	$n_{train_{L1}}$	$n_{train_{L2}}$	TWT (min)
25	25	796,269	25	25	481,183
26	24	791,539	26	24	479,626
27	**23**	**789,331**	**27**	**23**	**479,605**
28	22	789,654	28	22	481,140
29	21	792,577	29	21	484,4285
30	20	798,220	30	20	489,128

The bolded entries correspond to the optimal train distribution. We note they are equivalent, despite the passenger demands on both lines between Monday and Sunday being very different. We therefore move to calculate TWT across 3-hour chunks in order to capture more variation in the passenger demand through a day.

3-Hour Analysis

On the same passenger demands displayed in Figure 9, we present the results:

Table 2: Results of 3-Hour Analysis, $n_{train_{total}} = 50$

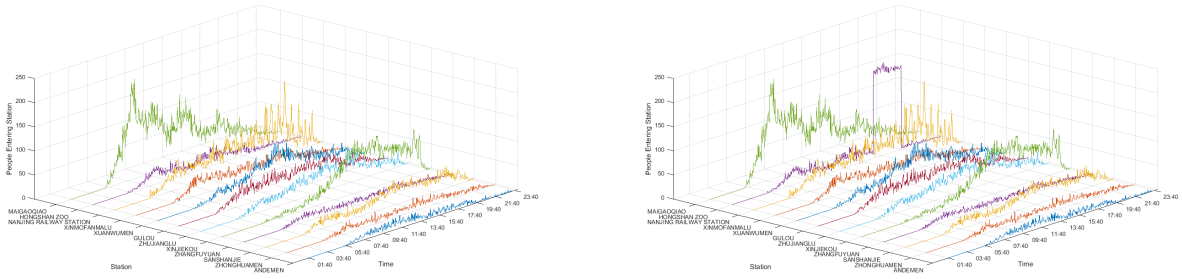
Monday			**Sunday**		
Time Block [HR]	$n_{trains_{L1}}$	$n_{trains_{L2}}$	Time Block [HR]	$n_{trains_{L1}}$	$n_{trains_{L2}}$
0-3	25	25	0-3	27	23
3-6	28	22	3-6	28	22
6-9	28	22	6-9	28	22
9-12	28	22	9-12	26	24
12-15	28	22	12-15	26	24
15-18	28	22	15-18	26	24
18-21	28	22	18-21	28	22
21-24	28	22	21-24	27	23

The train allocations are varied through the day, as we consider smaller chunks of time, allowing the optimization problem to capture fluctuations in passenger demand. This is particularly apparent on Sunday, in which there lacks a regular pattern in passenger demand through the day.

Artificial Demand Simulation

We now consider an artificial demand, in which we purposefully impose a large demand at a given station, and determine the best allocation of trains to meet that demand. In

this situation, we impose a load of 10,000 total passengers entering Station 15 on Sunday between 5 and 8 PM on Line 1. We visualize this load in the following comparison.



(a) Sunday, Line 1

(b) Sunday, Line 1, Artificial Demand

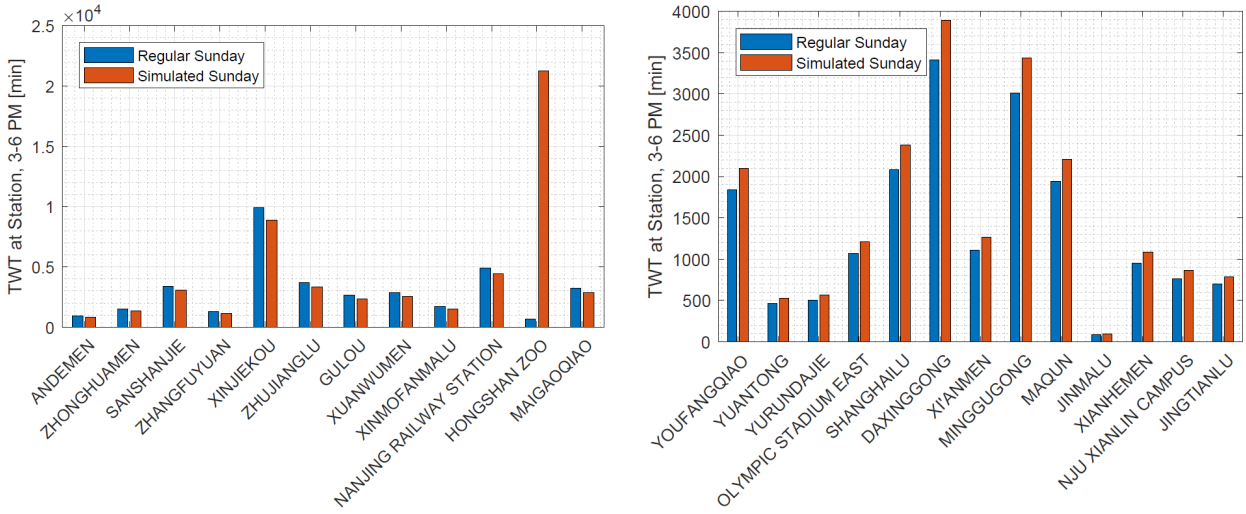
Figure 10: Passenger demand, Sunday, Lines 1 & 2, North to South

The 3-hour analysis on these two different passenger demand schemes results in the following train allocations.

Table 3: Results of 3-Hour Analysis, $n_{train_{total}} = 50$, Artificial Demand

Sunday			**Simulated Sunday**		
Time Block [0-24] HR	$n_{trains_{L1}}$	$n_{trains_{L2}}$	Time Block [0-24] HR	$n_{trains_{L1}}$	$n_{trains_{L2}}$
0-3	27	23	0-3	27	23
3-6	28	22	3-6	28	22
6-9	28	22	6-9	28	22
9-12	26	24	9-12	26	24
12-15	26	24	12-15	26	24
15-18	**26**	**24**	**15-18**	**29**	**21**
18-21	28	22	18-21	28	22
21-24	27	23	21-24	27	23

We see that the optimization problem shifted three trains from Line 2 to Line 1 during the 3 hours that the load was applied on Line 1. It is illuminating to see how this reallocation affects the wait times on stations along lines 1 and 2 during that 15-18 timeslot. We compute those total wait times per station, and present the results below:



(a) Stations on Line 1

(b) Stations on Line 2

Figure 11: Passenger Demand, Sunday, Lines 1 & 2, North to South

We note that the additional trains reallocated to Line 1 on the "simulated" Sunday, wait times across Line 1 are overall lower than a "regular" Sunday, except for Station 15 (where the 10,000 extra passengers were injected). On Line 2, which received fewer trains than a "regular" Sunday, wait times were slightly higher across the board on the "Simulated" Sunday. These results are to be expected due to the reallocation of trains.

Discussion

Our project results indicate that our mathematical model can respond to the fluctuating system demands that exist in a modern public transit system. Our formulation is flexible enough to analyze the problem at any time horizon desired, and is therefore powerful, as it allows one to allocate trains given that they know the anticipated passenger demand, acting within our model assumptions. Indeed, our project touches on an aspect of a growing issue in the transportation field; how to best allocate limited resources to meet demand while reducing costs.

There exist many ways to extend and improve this project. By happenstance, during the time period of data given, a new metro line opened. This inspired authors Max Jung, Carlin Liao, and others in Prof. Joan Walker's CE 264 to study the effects this opening has on passenger behavior. Such a behavioral aspect could also improve the accuracy of our model; we currently assume that passengers will arrive at stations in the same fashion regardless of headway, for instance. Following this thought, longer and variable headways may affect passengers' decision making in when they arrive at a station. An extension to the Traveler Station Entrance Distribution section of our analysis is explored by author Carlin Liao in Prof. Roy Dong's IEOR 290.

This project also illuminated the limitations of optimization and modelling techniques in engineering. The more accurate a optimization problem becomes, the more complex the model becomes, thereby limiting available solution methods. Our model ultimately required

an exhaustive search method. For a larger sized problem (perhaps if we considered more lines), such a method would become too computationally expensive. The computational aspect of our project is being investigated by authors Max Jung and Henry Teng in Prof. Raja Sengupta's CE 290I.

Another improvement would be to improve our model of passenger wait time. As learned through literature review, we must add terms accounting for passengers left on the platform due to limited train capacity. Also, as our methods are not specific to the Nanjing Metro system, further studies could also replicate our procedure on similar smartcard data from BART, or even the Shenzhen metro dataset we had originally anticipated.

Summary

Our project aimed at combining the operator's and passenger's perspective in allocating trains across lines of a metro system. In particular, we hoped to reduce operator costs (the number of trains deployed), while also reducing passenger wait time. This was accomplished by formulating a cost function that accounted for the wait time of passengers across multiple lines of a subway system, and related the number of trains on a line to the headway on that line. This formulation allowed us to optimize the cost function given additional operational and demand constraints to determine the best distribution of trains to reach a minimum system-wide passenger wait time. To ensure the validity of our problem, exploratory data analysis was done to study our problem assumptions, namely passenger arrival and headway assumptions. Our optimization problem was solved given various passenger demand schedules (from Nanjing Metro smartcard data), and we were able to successfully account for anticipated passenger demands with our model by allocating trains from lesser demand lines to higher demand lines during select hours of the day.

References

[1] P. Shang, R. Li, and L. Yang, "Optimization of urban single-line metro timetable for total passenger travel time under dynamic passenger demand," *Procedia Engineering*, vol. 137, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705816002587

[2] J. Zhao, F. Zhang, L. Tu, C. Xu, D. Shen, C. Tian, X.-Y. Li, and Z. Li, "Estimation of passenger route choice pattern using smart card data for complex metro systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 4, 2017. [Online]. Available: https://arxiv.org/pdf/1605.08390

[3] L. Sun, J. G. Jin, d.-H. Lee, K. W. Axhuasen, and A. Erath, "Demand-driven timetable design for metro services," *Transportation Research Part C: Emerging Technologies*, vol. 46, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X1400182X

[4] Y. Wang, B. De Schutter, T. van de Boom, B. Ning, and T. Tang, "Real-time scheduling for single lines in urban rail transit systems," *IEEE*

International Conference on Intelligent Rail Transportation, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705816002587

[5] M. Luethi, U. Weidmann, and A. Nash, "Passenger arrival rates at public transport stations," *Transportation Research Board 86th Annual Meeting*, 2007. [Online]. Available: https://trid.trb.org/view/801220

Acknowledgments

Thanks to Dounan Tang and by extension the Nanjing Metro Group Company for providing us the dataset for use in our analysis; Kai Zheng and Zhe Zhou of Tsinghua University for providing our initial inspiration with the Shenzhen system; and Hualiang Teng of the University of Nevada, Las Vegas for initial suggestions on our project.

And, of course, a very special thanks to Scott Moura, Bertrand Travacca, and the rest of CEE 295 for guiding and supporting our project, as well for awarding us the People's Choice Award for best overall project this year.

About the authors

Carlin Liao is graduating with his M.S. in Civil Engineering (Civil Systems) from UC Berkeley and will be pursuing his Ph.D. in Transportation at UT Austin.

Johnny Wu is graduating with his B.S. in Energy Engineering from UC Berkeley.

Henry Teng will be completing his M.S. in Civil Engineering (Civil Systems) at UC Berkeley with Prof. Khalid Mosalam.

Moon Ki "Max" Jung is graduating with his M.S. in Civil Engineering (Civil Systems) from UC Berkeley. He completed research with Prof. Stephen Glaser.

Appendix

MATLAB scripts for the train distribution optimization and Jupyter notebooks for the station entrance analysis have not been included with this report. If you would like the former, please contact the authors, while the latter can be found at

`https://github.com/CarlinLiao/smartcard-analysis`

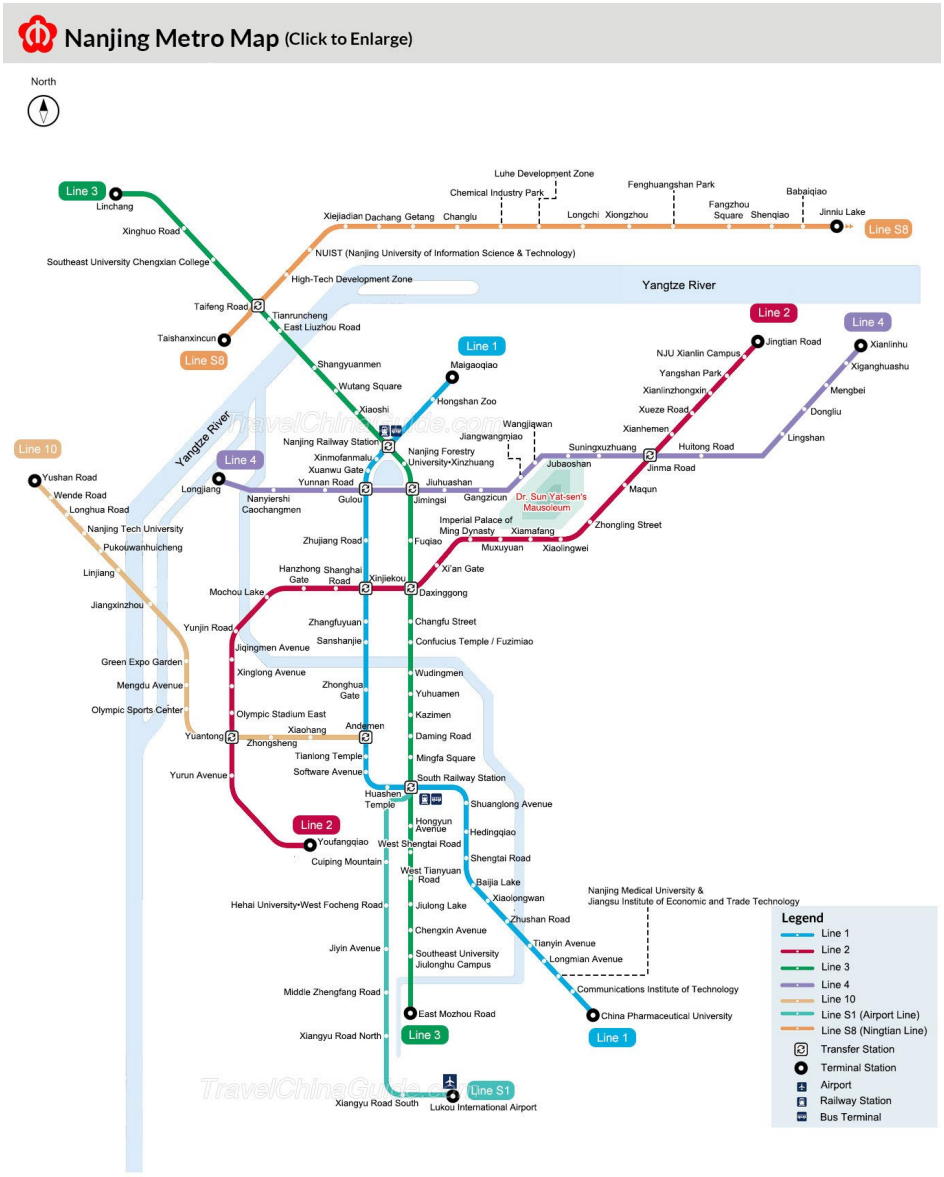


Figure 12: Map of the Nanjing Metro system, present day

Minimizing Ridesplitting Vehicle Miles Traveled

Caroline Neaves, Kaylee Homolka, Riley James, Shanay Kapadia

Abstract

To achieve emission reduction goals, it is necessary to understand the vehicle miles traveled (VMT) and emissions impacts of ridesourcing. Recent studies suggest that ridesourcing yields an increase in VMT and therefore an increase in emissions. However, ridesplitting may be a way to preserve the advantages of ridesourcing while negating the harmful environmental effects. This study aims to quantify the difference in VMT between ridesplitting and ridesourcing. To achieve this, we compared data from ridesplitting trips in Berkeley, CA, to an optimization program built to model rides in the same area. The results show that ridesplitting saves an average of 0.23 miles per vehicle miles travelled compared to single rides. This value could be increased by 35% if drivers followed an optimal route to minimize emissions.

Introduction

Motivation and Background

To keep global warming between 2 and 3 degrees Celsius, we need to stabilize the atmospheric concentration of carbon dioxide at 450 parts per million. To accomplish this goal, drastic cuts are needed in every sector of the economy. According to the United States Environmental Protection Agency, transportation accounts for 28% of US greenhouse gas (GHG) emissions, and 40% of GHG emissions in California. Currently, emissions projections for transportation are expected to grow. At the same time, the transportation sector is undergoing significant changes. Transportation services such as on-demand ridesourcing are becoming increasingly popular all over the world.

Since the introduction of Uber in 2009, ridesourcing services have increased significantly. Ridesourcing allows users many benefits. The convenience of door-to-door transportation is very enticing in many situations. A study done by A. Henao (2017) found that both car owners and non-drivers use ridesourcing for various reasons. People that own cars choose to use a ridesourcing service when they are out of town, when they are going to/from the airport, and for social trips.The primary reasons drivers may favor ridesourcing for social trips are difficulty of finding parking or to avoid drinking and driving (Henao, 2017). People that do not own cars regularly use ridesourcing to commute to work or school as a result of lack of satisfactory public transportation or time constraints (Henao, 2017).

On the other hand, the adoption of ride-hailing services may be harmful from an environmental perspective. The environmental effect of ridesourcing depends on the type of transportation passengers would have chosen if ridesourcing were not available. Studies show that this is very dependent on location. For example, Hampshire et al. (2016) completed a study in Austin, Texas where driving is very prevalent. If ridesourcing were not available, the study revealed that 45% would have driven. Conversely, Rayle et al. (2014) conducted a study in San Francisco, California where driving is far less common. The San Francisco study showed that only 7% of the trips would have been taken by driving had ridesourcing been unavailable. Instead, this study shows that 30% of ridesourcing trips are substituting for public transit and 9% walking or biking. In locations such as Austin, ridesourcing may not have significant environmental repercussion at this time. However, in locations such as San Francisco, where ridesourcing trips are replacing much lower emission forms of transportation, there must be an increase in emissions.

Another major concern related to the popularity of ridesourcing is induced demand. Induced demand is a phenomenon where more of a good is consumed after supply increases. In the case of ridesourcing, this means that passengers choose to take rides they otherwise would not have made because of the availability of these services. Rayle et al. (2014) reported that 8% of respondents said that, had ridesourcing not been available, they would not have made the trip at all. Similarly, Henao (2016), reported that in Denver and Boulder, Colorado, there was a 12% induced demand effect. While these values may seem small, they are not inconsequential. Any trip from induced demand results in additional emissions, which is very concerning from a climate change perspective.

An important metric of ridesourcing are out-of-service miles, also referred to as deadheading. Deadheading is the miles driven by ridesourcing drivers while driving to the passenger pickup point, waiting for a passenger request, or seeking passengers in a higher demand area (Shaheen et al., 2018) (Ngo, 2015). These additional miles driven are potentially increasing fuel consumption, emissions, and congestion. While there are not many studies on deadheading, the few results show very significant numbers. Henao (2016), found that, on average, ridesourcing drivers travel an additional 69.0 miles in deadheading for every 100 miles they are with passengers. A study completed in San Francisco found that 20 percent of ridesourcing VMT are deadheading miles (SFCTA, 2017). While there are not many studies examining deadheading, the results show very significant numbers that should not be overlooked.

There have been very few studies that attempt to quantify the overall Vehicles Miles Traveled (VMT) impacts of ridesourcing services. Presently, VMT change is difficult to measure because ridesourcing companies are reluctant to share data and there are potential modal shift implications (Shaheen et al., 2018). However, as a result of mode substitution, induced demand, and deadheading, it may be reasonable to assume that ridesourcing increases overall VMT in most locations. Schaller (2017) provided a preliminary calculation of 3.5 percent increase in citywide VMT. If left unmitigated, congestion and emissions are expected to grow as a result of this service (Clewlow et al.)

In 2014, some ridesourcing services began offering ridesplitting options. Ridesplitting involves matching passengers traveling in the same direction in order to share the ride and the cost. The primary motivation for users to choose this option over a private ride is the reduced cost. However, the social and environmental benefits may be reason enough for some

riders. Conversely, there are many factors that discourage the use of ridesplitting. Excess travel-time is considered the main deterrent. If a rider is in a hurry, they may be willing to pay the extra few dollars to get to their destination as quickly as possible. Additionally, personal preferences such as comfort travelling with strangers may prevent users from choosing the ride sharing option. Ridesplitting options are currently only offered in some locations.

In this study, we focus on the ridesplitting options offered by Lyft and Uber. Lyft Line and UberPOOL operate very similarly. A passenger requests a Line or Pool for up to two people for a discounted price. The ride may or may not be matched with other riders travelling in the same direction. Uber Express Pool is another ridesplitting options offered by Uber. For an Express pool, a passenger requests a shared ride for a discounted cost and is instructed to walk a short distance to a pickup point. At the end of the shared ride, passengers are dropped off a short distance away from their destination. Essentially, Express Pool is a ridesplitting service with more efficient pickups and dropoffs.

Ridesplitting systems pose a potential strategy for alleviating the harmful environmental effects while preserving the societal benefits. The intent of the rideshare system is to bring together travelers with similar itineraries and time schedules on short-notice (Agatz, Niels, et al.). Studies suggest ridesplitting could reduce the number of cars on the road, while simultaneously increasing the utilization of available seat capacity. This solution has the potential to significantly reduce congestion, vehicle emissions, fuel use, auto reliance and travel costs (Shaheen et.al, 2018). Ultimately, increasing occupancy rates can lead to improved air quality and reduced carbon dioxide emissions - a major contributor to global climate change. There are currently few studies of the impacts of new service models such as ridesplitting.

Dynamic ridesourcing is an automated system that facilitates drivers and riders to share trips on very short notice. Optimization of ridesplitting can focus on three specific objectives: 1. Minimizing system-wide vehicles-miles, reducing pollution and congestion 2. Minimizing the system-wide travel time, an important convenience consideration 3. Maximize the number of customers, maximizing revenues for private ridesourcing providers (Agatz, Niels, et al., 2012)

Relevant Literature

Given that ride-sharing companies have only risen to prominence over the last several years, research on the environmental impacts of ride-sharing is limited. In addition, many of these studies are small-scale and region specific. While there are not many aggregate studies on how ride-sharing can reduce emissions, several case studies of specific cities have been done. For example, a study of ride sharing use in Beijing found that using ride-sharing instead of a personal vehicle could lead to an overall reduction in CO_2 emissions from the transportation sector, along with reductions in emissions of other air pollutants like NOX and SOX (Yu et al., 2017). Furthermore, a study of vehicle use in the city of Changsha, China found that ride-sharing could reduce CO_2 emissions by up to 4 tons a day when compared emissions from using a personal vehicle (Jalali et al., 2017). These studies may not be applicable to other regions with different characteristics, and therefore more research is necessary, especially in US cities.

Additionally, several case studies have used dynamic ride-sharing models to investigate the carbon emissions impacts of ride-sharing systems. One such study from Sichuan Uni-

versity used an integer programming model to maximize greenhouse gas reductions from ride-sharing, and used this model to calculate both overall emissions savings and potential increases in savings from expanding ride sharing participation. The study found that ride-sharing could reduce emissions in two main ways: reducing the total number of vehicle miles traveled by sharing rides between passengers with overlapping routes, and decreasing the overall number of cars on the road, which decreased traffic and congestion (Ma et al., 2016). Similarly, another study using a mixed integer programming model to minimize total vehicle miles traveled in ride-sharing populations found that reductions in total VMT can be increased by increasing driver flexibility and the density of participants (Armant & Brown, 2014).

In contrast, some studies have found that the increased use of ride-sharing has not necessarily been effective at reducing the environmental impacts from driving. For example, a study investigating ride-sharing patterns in California found that Uber and Lyft vehicles are on average less fuel efficient than taxis, and therefore may not result in an overall reduction in carbon emissions (Wagner 2017). In addition, a case study of Paris found that while increasing vehicle occupancy through ride sharing had the potential to significantly reduce transportation CO2 emissions, various rebound effects could lead to decreases in these emissions savings in the long run (Yin et al., 2017).

Focus of this Study

This study aims to quantify the difference in VMT as well as associated emissions as a result of ridesplitting as opposed to individual ridesourcing.

Key Contributions

Due to the conflicting findings of existing studies, further research on the environmental impacts of ride-sharing is needed. This study provides a case study of the VMT and emissions associated with ridesourcing versus ridesplitting for Berkeley, CA.

Data Identification and Sourcing

Data is collected on Uber Pool, Uber Express Pool, and Lyft Line rides using the researchers own ride histories and the ride histories of other UC Berkeley Students. For each ride, the cost of the trip, the company and type of trip, the time and date, and the number of passengers in the riders party is recorded. The main riders pick-up and drop off points along with total trip distance and duration is measured. Additionally, the number of additional passengers and their pick up and drop off points is recorded, if applicable. This is used to calculate the overlap between the main riders route and the routes of additional passengers. Finally, in the case of Uber Express Pool, the distance the rider walked to get to the Express pick-up point is measureds. There are several limitations associated with this data. First, this study uses ridesharing data from only UC Berkeley students, meaning our results cannot necessarily be applied to rideshare customers as a whole. In addition, Uber and Lyft do not provide detailed data on the routes of additional passengers, so overlap between the main rider and additional passengers has to be estimated instead of directly measured. To collect

this data, it was necessary to note the approximate address of additional passengers pick up and drop off points as the ride was in progress. If respondents did not record these data, the ride data was unusable. Additionally, our current model is designed to analyze rides only in a section of Berkeley. As a result, many of our collected data points were unusable, leaving our sample size at 22 rides. Finally, we do not have any driver data, so we are not able to calculate differences of deadheading between individual ridesourcing rides and ridesplitting. For future research, we would aim to collect additional data points, and possibly collect more precise driver and rider data directly from Uber or Lyft.

Methodological Approach

To minimize VMT and emissions, an optimization program was implemented over a set of nodes located at cross streets in Berkeley. Each of these nodes has an associated longitude, latitude, and elevation. The grid boundaries were chosen to include common Berkeley destinations, including UC campus, grocery stores, and restaurants. The grid was simplified to include only intersections that were roughly one block (0.1 mi) apart or more. One ways streets and dead ends were accounted for when calculating optimal routes. An adjacency matrix was built to convert the grid into a logical array. For a given node, adjacent nodes were marked with a 1, and all other nodes left as 0. The grid of Berkeley was simplified to include 174 nodes, each at an intersection, as seen in Figure 1. Data is collected from group members and classmates based on their recent Uber trip history. Only rides beginning and ending within our defined grid were considered. The goal is to obtain the VMT saved by taking an Uber Pool vs. an individual ride. Respondents recorded their own pickup and dropoff points, as well as those of any riders on their trip.

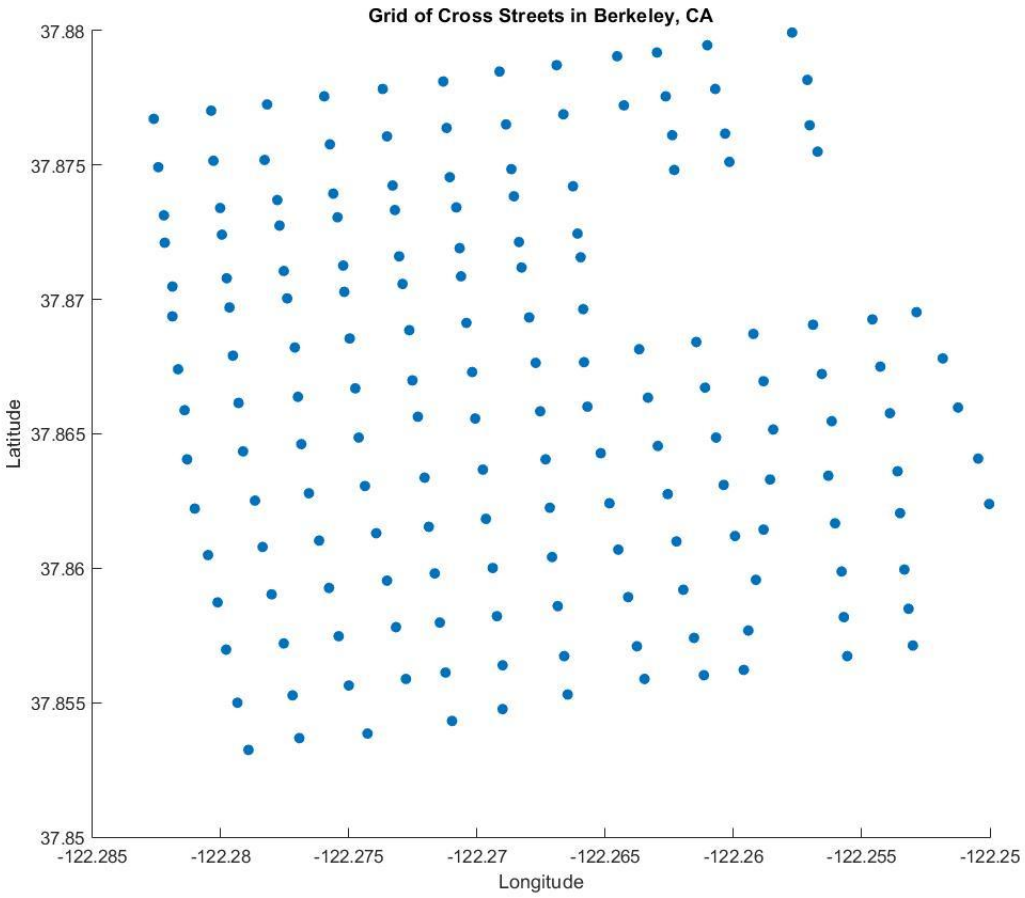


Figure 1: Grid of cross streets in Berkeley, CA

The distance between the nodes is found using the Haversine formula. Details on how the Haversine formula works can be found in Appendix A.

Using this grid and optimization techniques, the route that results in the least vehicle miles traveled was found. In addition, dynamic programming was used to find the shortest route between pickup and drop of points, and integer linear programming to find the optimal path between them. Dijkstra's algorithm, an implementation of dynamic programming, was used to find the distance between nodes. A brief explanation of how this algorithm works can be found in Appendix B.

The formulation for our integer linear program is as follows (Nicholas et al., 2016):

$$Minimize: \quad d = \sum_{ij} C_{ij} X_{ij} \tag{1}$$

$$Subject\ to: \quad B_i + t_{ij} - B_j \leq M(1 - X_{ij}) \tag{2}$$

$$B_j - B_i \leq M X_{ij} - t_{ij} \tag{3}$$

$$B_i \leq B_{i+2} \tag{4}$$

$$X_{ij} = \begin{cases} 1 \, if \, traveling \, from \, node \, i \, to \, node \, j \\ 0 \, otherwise \end{cases} \tag{5}$$

$$B_1 = 0 \tag{6}$$

$$\sum_j X_{ij} = 1 \; for \; i \in driver, origin \tag{7}$$

$$\sum_j X_{ij} = -1 for j \in origin, destination \tag{8}$$

where d is the shortest distance, C_{ij} is the cost, or distance, to go from node i to node j, X_{ij} is the decision variable which is equal to 1 if the optimal route passes from node i to j, B_i is the time of arrival at node i, t_{ij} is the time to travel from node i to node j, and M is a large number used in the "big M method," with B_i and X_{ij} being the optimization variables.

The objective function, in equation 1, minimizes the cost-to-go between two nodes over the full duration of an origin-destination trip, where X_{ij} is assigned a value of 1 if the link between nodes i and j is taken, and 0 if it is not, as shown in equation 5.

The function *intlinprog* was used to solve this in MATLAB. Integer linear programs require the inputs in the form (MATLAB 8.0, 2018):

$$min_x \, f^T x \, subject \, to \begin{cases} x \, are \, integers \\ Ax \leq b \\ A_{eq} x = b_{eq} \\ lb \leq x \leq ub \end{cases} \tag{9}$$

Using the formulation shown in equation 9, x_1 to x_{25} correspond to X_{11} to X_{55} and x_{26} to x_{30} correspond to B_1 to B_5. The A matrix and b vector are created to encode equations 2, 3, and 4, while the A_{eq} matrix and b_{eq} vector encode equations 6, 7, and 8. The integer requirement of the decision variable X is implemented by the integer linear program formulation.

To find the minimum VMT for the trip, first Dijkstras algorithm is used to find the shortest path between the driver node, the origin nodes, and the destination nodes. For the ridesplitting case, the above integer linear program is used to find the shortest route, while visiting the nodes in the optimal order. The path is then recorded and plotted, along with the associated VMT. For the single ride case, the path found by Dijkstras algorithm between the riders origin and destination node is the optimal path, and is plotted along with the optimal ridesplitting path.

Analysis

For this analysis, the primary variable of interest is vehicle miles traveled (VMT). Using Dijkstras algorithm and integer linear programming, the minimal ride distance is found for both ridesplitting and single ride cases. The results from the optimization program are used to calculate the difference in ridesplitting and single rides. This difference is calculated using

equation 10.

$$\Delta d_{single} = (d_{riderA} + d_{riderB}) - d \tag{10}$$

where Δd_{single} is the difference in distance for ridesplitting and single rides, and d_{riderA} and d_{riderB} are the optimal distances for two riders taking separate rides. Also calculated from the results of the optimization program is the difference between the optimal route and the actual route that was taken. The miles saved for single and split rides are calculated using the following formulas:

$$\Delta d_{actual} = d_{actual} - d \tag{11}$$

$$\Delta d_{actual,single} = (d_{riderA} + d_{riderB} - d_{actual}) \tag{12}$$

$$r_{actual} = \sum \Delta d_{actual} / \sum d_{actual} \tag{13}$$

$$r_{actual,single} = \sum \Delta d_{actual,single} / \sum (d_{riderA} + d_{riderB}) \tag{14}$$

$$r_{single} = \sum \Delta d_{single} / \sum (d_{riderA} + d_{riderB}) \tag{15}$$

where d_{actual} is the actual distance taken on the ridesplitting trip, d is the calculated optimal distance for the split ride, and Δd_{actual} is the difference between these values. That is, equation 11 calculates the miles that would be saved by taking the optimal route for a split ride. $\Delta d_{actual,single}$ is the difference between the optimal and actual distances for a single rider, calculated in equation 12. r_{actual} is the savings for ridesplitting passengers taking the optimal route compared to the actual route, calculated in equation 13. Equation 14 calculates the savings between the split and single rider trips ($r_{actual,single}$), and equation 15 calculates r_{single}, which is the savings for a single rider taking the optimal route. When calculating these savings for Uber Express Pool trips the walking distance was included in this difference.

In addition to the VMT differences between trips, the emissions saved by taking a shared trip is also calculated. The saved trip emissions, ϵ_{net}, is found using equation 16:

$$\epsilon_{net} = (d_{net})(E_{gas})/FE \tag{16}$$

where E_{gas} is the the CO2 emissions associated with a burning a gallon of gasoline. The two vehicles being analyzed in this case will be the Toyota Camry with a fuel efficiency (FE) of 29 miles per gallon and the Toyota Prius with an FE of 54 miles per gallon (Toyota 2018). These two vehicles were chosen for emissions analysis because they were the most common vehicles used in our data set. According to the Energy Information Administration, there are approximately 19.6 pounds, or 8.89 kilograms, of CO2 produced when one gallon of gasoline is burned (EIA 2017). Therefore, using equation 16, $E_{gas} = 8.89$kg/gallon.

Results and Discussion

To find the difference in distance for ridesplitting vs. single rides, the shortest distance for each single ride is first found. Then, the difference in distance is calculated between the sum of the distances of the two individual trips and the ridesplitting distance. Table 1 summarizes the optimal trip distance for a shared ride, along with the distance traveled if each trip had been made up of two single rides (A and B) instead of a split ride. This table also gives the combined distance of the two individual trips, and the actual distance of the ridesplitting trip.

Trip #	Optimal Distance (mi)	Single Distance A (mi)	Single Distance B (mi)	Combined Distance of Rides A+B	Actual Distance (mi)
1	0.7394	0.7394	0.6663	1.41	1.6
2	1.8256	1.5467	1.3306	2.88	1.93
3	2.4834	2.3168	1.8138	4.13	2.5
4	2.9218	2.3168	0.8616	3.18	3.04
5	2.1314	1.8562	1.5680	3.42	2.5
6	2.8102	2.3168	1.0437	3.36	2.9
7	2.6271	2.3168	1.0806	3.4	2.63
8	2.6789	1.3764	1.3070	2.68	2.05
9	1.0692	1.0557	0.5968	1.65	1.71
10	1.6829	1.6310	0.8462	2.48	2.3
11	2.7073	2.0955	1.9271	4.02	2.74
12	1.6892	1.6474	1.0784	2.73	2
13	1.7148	1.4118	0.6057	2.02	2.3
14	1.5837	1.5574	0.9739	2.53	2.2
15	2.3979	2.3168	1.4255	3.74	2.7
16	2.4152	2.0367	1.3181	3.35	1.8
17	0.8733	0.6142	0.7438	1.36	0.9
18	1.6010	1.6010	1.6010	3.2	1.8
19	1.5038	1.4867	0.7798	2.26	1.9
20	1.5499	1.2026	0.8434	2.05	2.2
21	1.1811	1.1811	1.1811	2.36	1.55
22	1.9428	1.9241	1.1096	3.03	2.2

Table 1: Results of path optimization for each of the collected trips

By comparing the Optimal Distance traveled when the two rides are combined into ridesplitting with the Total Distance traveled when each ride is conducted separately, it is clear that ridesplitting reduced the overall VMT. However, the actual distance of the route taken by the ridesplitting driver is often longer than the optimal route. In some cases, this even results in the actual ridesplitting distance being larger than the sum of the two individual distances for rides A and B. This occurs four times in our data set for Trips 1, 9, 13, and 20.

Figure 2 below shows a visual of the possible paths for Trip 15 traveling within the Berkeley grid. The individual routes for trips A and B are shown in blue and red, respectively. The optimal ridesplitting route is shown in black.

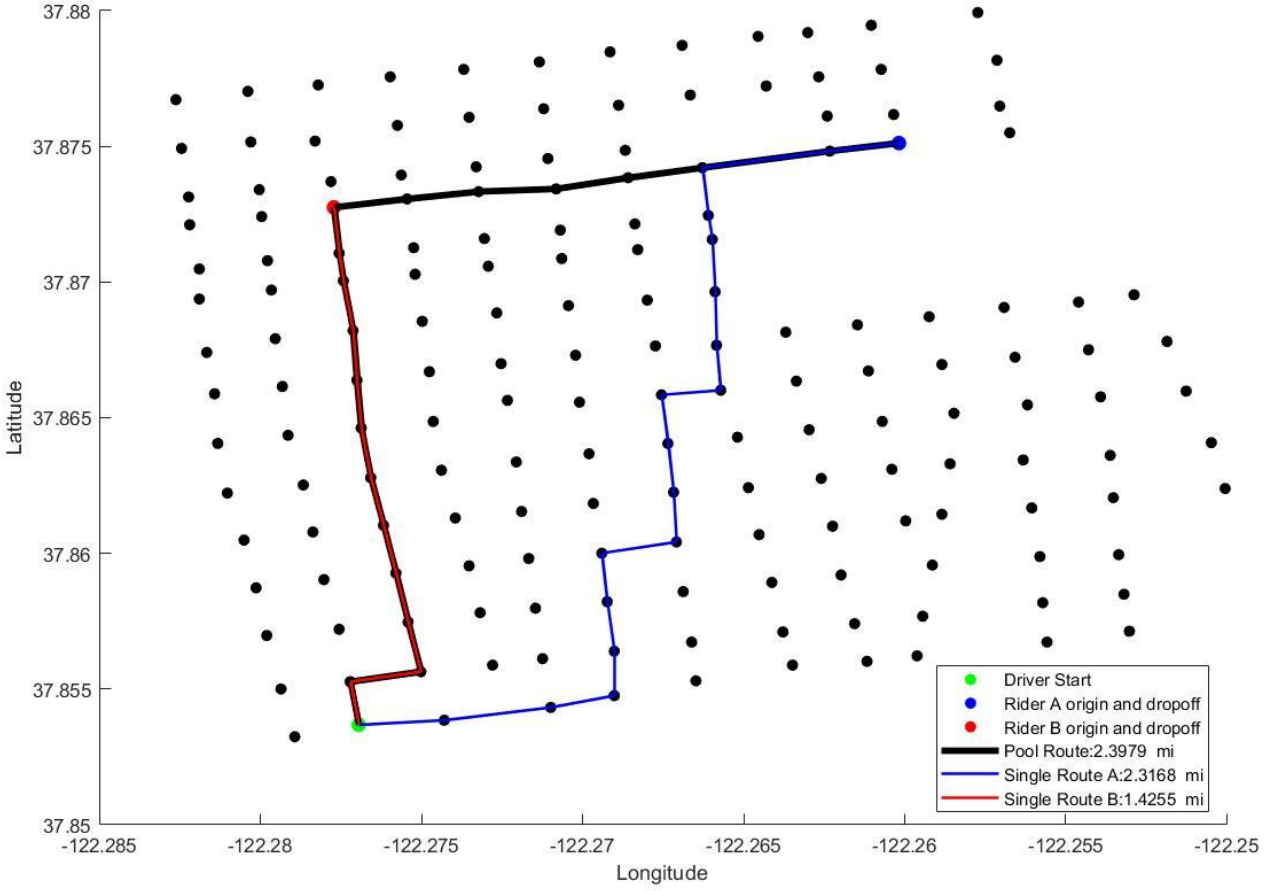


Figure 2: Plot of Trip 15. Ridespliting route plotted in black, with the single trips plotted in blue and red

As shown in Figure 2, ridesplitting reduces total VMT when compared to the total distance of each individual ride. The sum of the distances of individual rides A and B is 3.74 miles while the distance of the split ride is 2.40 miles, resulting in a total VMT savings of 1.34 miles.

Table 2 gives the average savings in VMT from using ridesplitting instead of single rides, for both the optimal ridesplitting difference and the actual ridesplitting difference. Thus savings are also normalized per mile traveled. Furthermore, the table gives the potential VMT savings from drivers following the optimal ridesplitting route instead of the actual route taken. Finally, the corresponding CO2 emissions savings per mile traveled are given for two different vehicles, the Toyota Camry and the Toyota Prius.

Interaction	Average Difference (mi)	Miles Saved per Mile Traveled	CO_2 Emissions Saved per Mile Traveled for Toyota Camry (kg)	CO_2 Emissions Saved per Mile Traveled for Toyota Prius (kg)
Optimal ridesplitting vs. Optimal single	0.869	0.312	0.096	0.051
Optimal ridesplitting vs. Actual ridesplitting	0.236	0.085	0.026	0.014
Actual ridesplitting vs. Optimal single	0.634	0.228	0.070	0.037

Table 2: Significant miles saved parameters

The results show that on average, 0.63 miles were saved from using ridesplitting instead of taking two single rides, calculated using the actual ridesplitting distance instead of the optimal distance. This corresponds to a per mile savings of 0.23 miles. Additionally, if ridesplitting drivers followed the optimal route, these average savings could be increased by over 35% to 0.31 miles saved per mile traveled. These optimal savings correspond to a CO2 emissions savings of 0.096 kg per mile traveled for the Toyota Camry, and a savings of 0.051 kg per mile traveled for the Toyota Prius.

Summary and Future Work

This paper supports the claim that ridesplitting reduces overall VMT and emissions from ridesourcing since the results show that ridesplitting saves an average of 0.23 miles per vehicle mile travelled compared to single rides. Additionally, ridesplitting fills empty seats in cars and potentially reducing congestion. In order to maximize these benefits, public policy should be implemented to incentivize ridesplitting. While ridesplitting already offers a significant discount for riders, policy could dictate cheaper split rides, or conversely, more expensive individual rides. This pricing technique to encourage ridesplitting could be especially impactful during peak hours. Similarly, ridesourcing could be priced to incentivize ridesplitting to public transportation connections, such as train stations, bus stations, or airports. The emissions savings of ridesplitting could be further improved by implementing ecorouting. Ecorouting is a navigation concept that finds a route requiring the least amount of fuel or producing the least amount of emissions (Boriboonsomsin et. al, 2012).

Additional insight could further enhance this study. We limited the study to Berkeley, which varies greatly from other parts of the country with respect to both geography and

demographics. This may have biased our data towards students and their behavior. It was also not possible to get data about other members in the split ride, so we only took into account the portion of the trip that overlapped with additional riders. Understanding the chain of ridership as riders get picked up and dropped off within the same vehicle and route could provide valuable data on behavior and miles saved. We also did not have the actual algorithm used by Uber or Lyft for routing, which takes into account additional factors such as congestion, speed limits, and u-turns. Finally, while the number of grid points was simplified for this study, the node density could be improved to include every intersection and crossing in Berkeley.

Additionally, further study should be done on the effects of elevation and deadheading on emissions. Especially in a city marked by topographical changes such as Berkeley, vehicles may have significant emission changes if rides are mostly uphill or downhill. Furthermore, emissions associated with both deadheading and induced demand are likely significant and should be accounted for in additional research.

Appendices

A: The Haversine Formula

This formula utilizes latitudes and longitudes to calculate the distance between two points on the surface of a sphere.

The Haversine function is given as:

$$hav(\theta) = sin^2(\theta/2)$$

Using this, the distance between two points on a sphere is calculated using the following formula:

$$d_{hav} = 2rarcsin(\sqrt{hav(\phi_2 - \phi_1) + cos(\phi_1)cos(\phi_2)hav(\lambda_2 - \lambda_1)})$$

where d_{hav} is the distance between the two selected points on a sphere, (ϕ_i, λ_i) are the latitude-longitude pairs of the points, and r is the radius of the earth.

B: Dijkstra's Algorithm

Given a source node, this algorithm calculates the shortest path to each other node in the grid. The algorithm works in three main steps: initialization, calculation, and evaluation. All nodes are initially set as unvisited, and a start node is chosen. The start node is given a distance value of 0, as it is the current node, and all other nodes are given distance values of infinity. In the calculation step, the cumulative distance between the current node and all of its children are evaluated. These calculated distance values are compared to the present values, and the smaller ones are assigned. The current node is then considered visited, and the node with the smallest distance value is then selected and the calculation step is redone.

This is done until all of the nodes have been visited. This allows for the optimal shortest path between any given origin and destination nodes to be calculated.

C: Division of Work

This project was divided among the authors as follows:

Caroline Neaves: Literature review, data collection, grid and adjacency matrix creation

Kaylee Homolka: Shortest path algorithm, mixed integer linear program algorithm assistance, LaTeX implementation

Riley James: Shortest path algorithm, mixed integer linear program algorithm, data collection

Shanay Kapadia: Data collection, grid and adjacency matrix creation

All members participated in the writing of the report and the discussion of the results.

Acknowledgments

We would like to thank our fellow classmates for providing survey data for us to conduct this study, as well as Scott Moura and Bertrand Travacca for their guidance and support throughout the process.

About the authors

Caroline Neaves Graduated from John Hopkins University with a degree in Environmental Engineering. Loves Papa Johns pizza, long walks on the beach, and gardening.

Kaylee Homolka Graduated from the University of Nebraska-Lincoln with a degree in mathematics. Fueled by coffee, dogs, and Radiohead.

Riley James Graduated from Occidental College with a major in physics. Passionate about calligraphy and Big Brother.

Shanay Kapadia Graduated from Rice with a degree in chemical engineering. Enjoys baseball, birdwatching, and battery storage.

References

Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2), 295-303.

Boriboonsomsin, K., Barth, M. J., Zhu, W., & Vu, A. (2012). Eco-routing navigation system based on multisource historical and real-time traffic information. *IEEE Transactions on Intelligent Transportation Systems*, 13(4), 1694-1704.

Clewlow, R. R., & Mishra, G. S. (2017). *Disruptive transportation: the adoption, utilization, and impacts of ride-hailing in the United States* (Vol. 7). Research Report-UCD-ITS-RR-17.

Hampshire, R. C., Simek, C., Fabusuyi, T., Di, X., & Chen, X. (2017). Measuring the impact of an unanticipated suspension of ride-sourcing in Austin, Texas.

Hawkins, A. J. (2018, April 02). BMW and Daimler are putting their differences aside to beat Uber. Retrieved from *https* : $//www.theverge.com/2018/4/2/17188374/bmw - daimler - merger - car2go - reachnow - mobility$

Henao, A. (2017). *Impacts of Ridesourcing-Lyft and Uber-on Transportation Including VMT, Mode Replacement, Parking, and Travel Behavior.* University of Colorado at Denver.

MATLAB 8.0 and Statistics Toolbox 8.1, The MathWorks, Inc., Natick, Massachusetts, United States.

Ngo, V. D. (2015). *Transportation network companies and the ridesourcing industry: a review of impacts and emerging regulatory frameworks for Uber* (Doctoral dissertation, University of British Columbia).

Nicolas, L., & Moura, S. J. (2016). Optimal Routing and Charging of Electric Ride-Pooling Vehicles in Urban Networks.

Rayle, L., Dai, D., Chan, N., Cervero, R., & Shaheen, S. (2016). Just a better taxi? A survey-based comparison of taxis, transit, and ridesourcing services in San Francisco. *Transport Policy*, 45, 168-178.

San Francisco County Transportation Authority (SFCTA). (2017). TNCs Today: A Profile of San Francisco Transportation Network Company Activity. Retrieved from *http* : $//www.sfcta.org$

Schaller, B. (2017). Unsustainable? The Growth of App-Based Ride Services and Traffic, Travel and the Future of New York City. Schaller Consulting.

Shaheen, S., & Cohen, A. (2018). Impacts of Shared Mobility: Pooling. *Shared Mobility Policy Briefs: Definitions, Impacts, and Recommendations*, 12.

Shaheen, S., Totte, H., & Stocker, A. (2018). Future of Mobility White Paper.

Toyota Motor Sales (2018). 2018. *Toyota Camry Full Specs and 2018 Prius Full Specs.*

Retrieved from $https://www.toyota.com/prius/features/mpg/1221/1223/1224$

US Energy Information Administration. (2017). *How Much Carbon Dioxide is Produced from Burning Gasoline and Diesel Fuel?* US Department of Energy. Retrieved from $https://www.eia.gov/tools/faqs/faq.php?id=307\&t=11$

Electric Vehicle Charging Station Controller

Salma Benslimane, Ilias Atigui, Zuwa Oriakhi, Mathilde Badoual

Abstract

In this project, we aim to minimize the Electric Vehicle (EV) owner's electricity cost from charging of his/her vehicle at a charging station, while ensuring that the EV state of charge is within acceptable bounds to meet their satisfaction. In developing this system, we consider varying electricity prices as well as the availability of the vehicle to be charged.

To tackle the problem of stochasticity, we develop a charging station controller that employs Stochastic Dynamic Programming and, more precisely, we model the vehicle availability as a **Markov Decision Process**. We contrast our results against those produced from using other established optimization/control methods.

Introduction

Motivation and Background

A growing public awareness about clean energy, combined with continued innovation in battery storage, suggests that the wide-scale adoption of Electric Vehicles (EVs) will be a likely scenario in the near future. Defining the role that EVs will play in society is a challenge that needs to be addressed on several fronts such as infrastructure compatibility, electricity generation and distribution constraints, effects on mobility, etc., as well as the opportunities that EVs present to the power system in the form of distributed storage.

Indeed, energy management strategies are the key to the successful integration of EVs both from a storage and load perspective. It is logical to assume that the adoption of EVs will increase the peak demand and will present added strain on the system. Effective demand response can decrease this peak by shifting the load demand from EVs to non-peak times which would significantly reduce consumer energy cost. Thus, our project will focus on control of a fleet of EVs.

This work is particularly important at a time where California's government wants 5 million zero-emission vehicles on the road by 2030 ('CEC: California EV chargers will add 1 GW of peak demand by 2025', *Utility Dive*, Gavin Bade). Nonetheless, using statistical data from the US Department of Energy, we can show that the current grid can only handle those electric cars so long as they actually represent less than 25% of the total number of cars on the road in a given area ('NREL Finds Future EV Charging Demand Will Require Coordination Between Utilities Car Owners', *CleanTechnica*, Steve Hanley).

The main issue of controlling EVs is to tackle EV presence and prices uncertainty. Then we will consider the stochastic approach using random variables for those values. This problem has been addressed in various papers cited in the Relevant Literature section.

Focus of this Study

Our goal is to develop an online controller to charge a fleet of EVs. The EVs should be in the charging state depending on the energy prices while guaranteeing the consumers satisfaction.

The first model will consider that the vehicles never leave and that the controller has to balance between minimizing costs owing to electricity prices and guaranteeing consumer satisfaction. Then, we will use a stochastic control to compute the optimal energy allocation taking into account the cars presence. Finally, we run a "stupid" model that charges when the car arrives. To make a relevant comparison, we will run a Monte Carlo simulation for the three models. We will assume that our models know the electricity prices during the day.

Here is a graph summarizing the system we are analyzing (Figure 1).

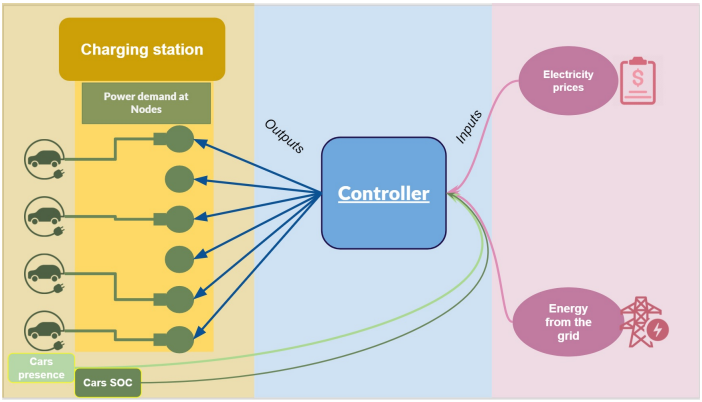


Figure 1: Description of the system's model

Literature Review

A number of modeling frameworks and optimization methods have been used to tackle the issues linked to the development of an online controller which would be used in order to charge a fleet of electric vehicles.

Keeping in mind that one of our main objectives is to minimize the electricity cost of the different charging stations: Caroline Le Floch et al. develop in [**?**] a model predictive control (MPC) formulation in order to minimize the electricity cost of a plug-in electric vehicles fleet under several distribution grid constraints; this has been crucial to get a better understanding of how a controller can actually be implemented in our system.

Motivated by the need to keep safe and efficient infrastructure, as well as the urgency to provide sustainable energy, Xiaohua Wu et al. worked in [**?**] on a smart home designed to

minimize the total cost depending on a specific tariff which varies with the time of the day. Furthermore, this paper also wants to satisfy the home power demand (which corresponds in a way to the satisfaction function we tried to establish, the comparison has been really helpful to get some insights of the problem).

In a context where the transition between fossil fuels and renewable energy is taking shape, Emil B. Iversen et al. address in [**?**] the burning issue of the involvement of different actors related to the energy sectors. And there is definitely a stochastic nature in these driving patterns. Therefore, it shows us tha tusing a Markov decision process can definitely be relevant in order to model the loading of an EV fleet.

Key Contributions

Thus, our team explored the computational complexity difference between convex optimization and DP as well as compared a 'simple model' to a more complex one, taking into account stochasticity of cars' presences.

Technical Description

We will first present the model of the EV charging station, the data we used for this study, followed by the different controllers we designed and finally the comparison between those controllers.

1 Description and Nomenclature

We define the variables and characteristics of our problem as follow:

- N : number of nodes (chargers) stations
- H : time horizon
- $c[k]$ electricity prices at time k, in $[\frac{\$}{kWh}]$
- $x_i[k]$ State of Charge (SOC) of car at node i at time k, in $[kWh]$
- $u_i[k]$ energy allocated to node i at time k, in $[kWh]$ - also **the control**
- $z_i[k] \in \{0, 1\}$ presence of car at node i at time k
- x_i^{init} initial state of charge of car at node i

Let's consider a charging station for electrical vehicles containing **N chargers or nodes** - each one representing a plug-in.

We consider a **time horizon of H** with $t \in [0, H-1]$ and a discrete time step $k \in \{0, ..., H\}$

Each vehicle present at a plug, i, at time, k, has an initial SOC and needs to be charged. At a certain time, the car will leave and we would need to keep track of the presence of cars, so we will define a vector containing the car presence at each plug i.

2 Data Used

Electicity Price

We gathered electricity prices data from the California Independent System Operator (CAISO). Our data is between 2013 and 2015, for each hour of the day and for each corresponding month.

The following graph is shows the electricity prices during 24h for different days in 2013. We can see that the prices are following a regular shape every day, thus we will consider the average price at each hour as the data known by our controller.

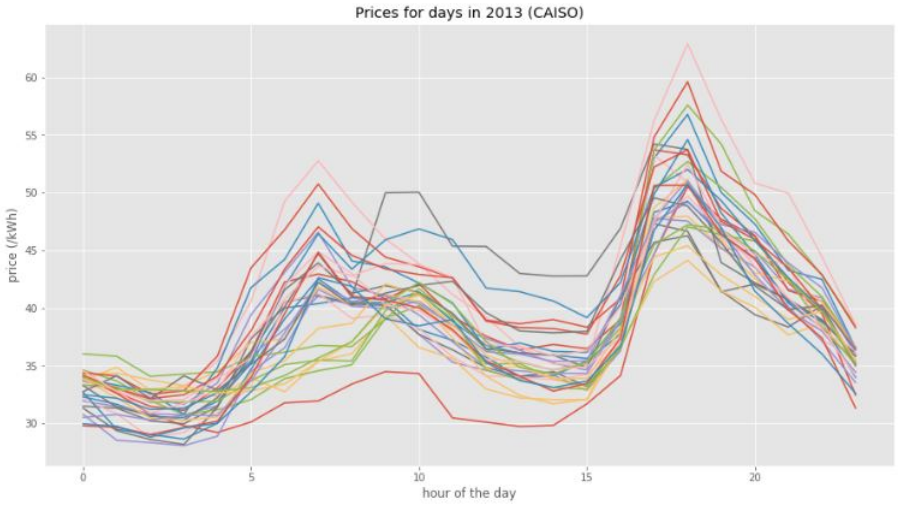


Figure 2: Prices data for one day from Caiso Prices

Electric Vehicles Energy Consumption and Presence

From PECAN Street data we collected the electricity consumption for five EVs at each hour of the day for each day of the month in 2014. This allowed us to deduce the presence of an EV during the day.

Our project is focused on residential charging station but PECAN Street data is collected from personal chargers, nevertheless, we can imagine the use case of a Smart Grid city such as Ecoblock Project in Oakland [reference] where residents are charging shared electric vehicles. Since it is a residential area, the charging profile might be similar to a personal charging station.

The figure below shows the vehicles charging patterns for Sunday as follows (Appendix A for the entire data) :

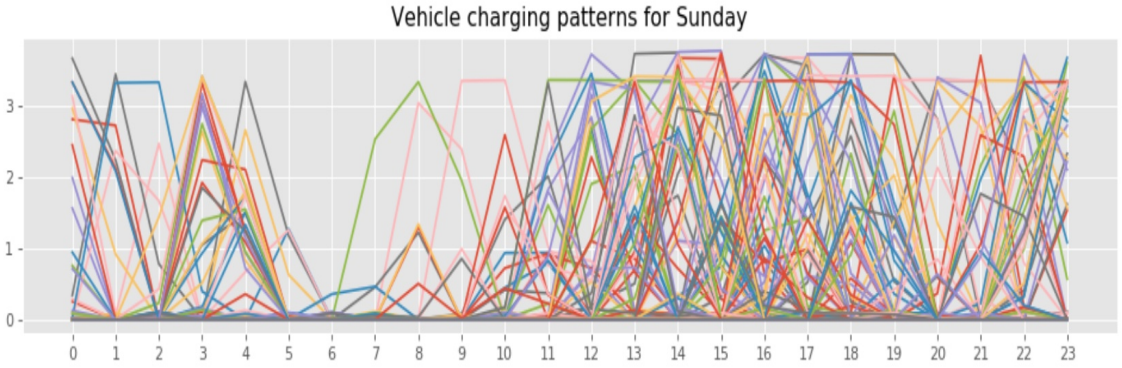


Figure 3: EV Data

By analyzing the shape of the car presence , we can deduce that it depends on the hour of the day but also on the previous state (charging or not). Then we can reasonably model this presence with a random variable following a Markov Chain model. If we denote $Z_i[k]$ the random variable associated to the presence of the car, we can write the Markov Chain model as:

$$\forall i, \quad \mathbb{P}\left[Z_i[k] = m | Z_i[k-1] = n\right] = p_{m,n}[k]$$

With m and n in $\{0, 1\}$. 0 is unplugged and 1 is plugged. The following plot represents the probability to switch modes.

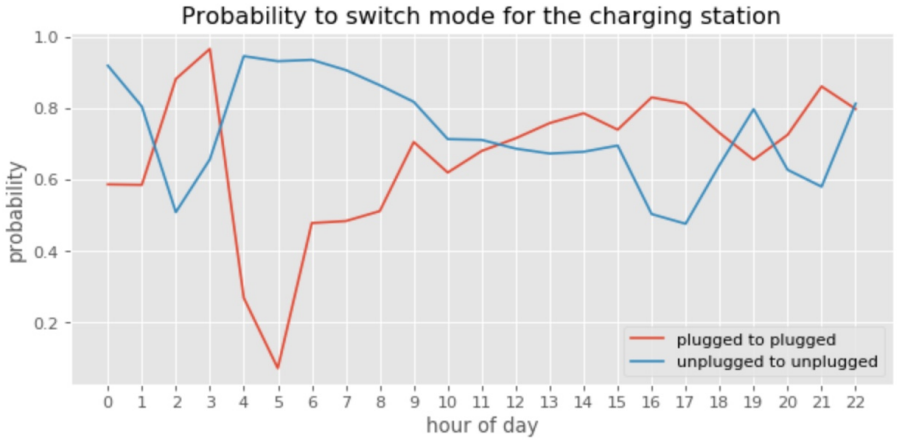


Figure 4: Probability of switching modes for a station

And here is an extract of the probability matrix for the first hour of the day and for the third one.

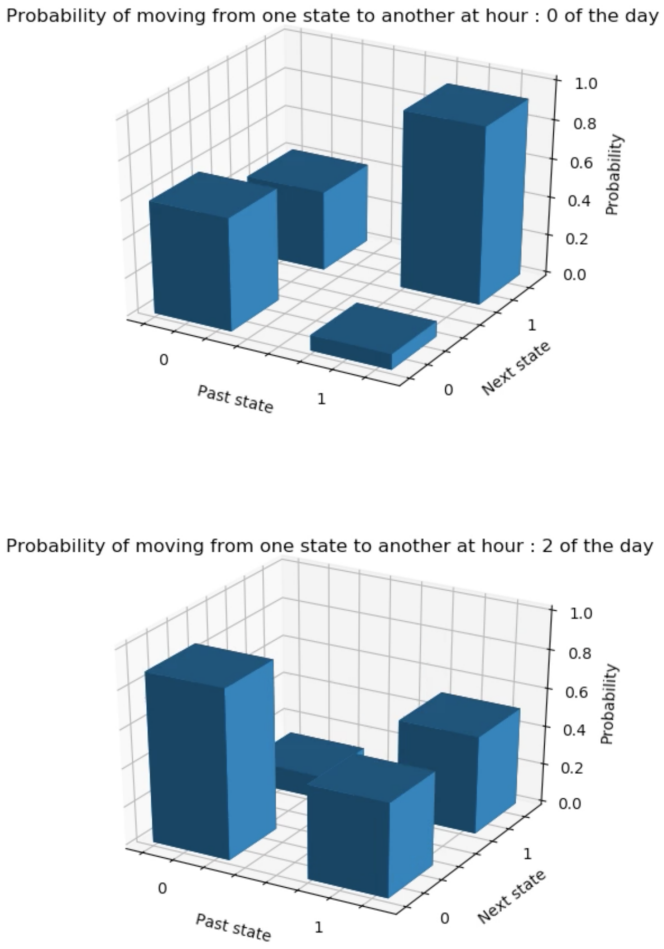


Figure 5: 3D representation of the probability of switching modes for a station

We can see that the plot above is consistent since the probability $\mathbb{P}$ of 'plugged to plugged' tends to 0 around 5-6AM. That's the time when most people go to work using their car.

Besides, we can see that during the night, between 2-3AM, Pr('plugged to plugged') is the highest (around 0.9) since the electricity price isn't too high compared to the price at the beginning of the night. Furthermore, we can highlight that although there are more attractive prices during the day, the cars still need to be charged during the night to allow people to commute in the morning.

We also computed the Markov Chain random variable for the prices in Appendix 1.

3 Satisfaction function

Each car plugged to the station is going to get charged up to a given SOC before leaving the station.

In order to control the level to which each vehicle is being charged, we constraint our problem by adding a satisfaction function to the cost.

We define the satisfaction function as follows - representing an exponential decay:

$$g(x) = \frac{1 - e^{\alpha \cdot (1-x)}}{1 - e^{\alpha}} \tag{1}$$

with α a constant.

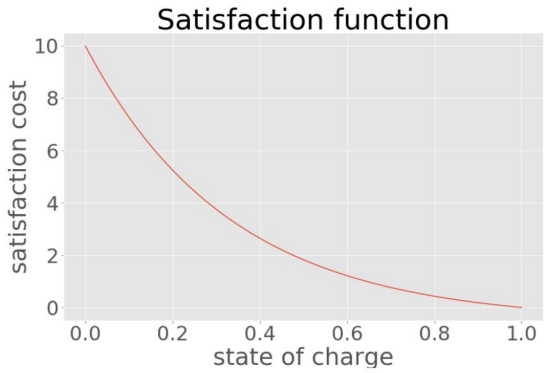


Figure 6: Plot of the satisfaction function vs the State of charge

4 Model 1: Simple Controller (Deterministic with constant car presence)

Optimization formulation

In this model, we assume that the price $c[k]$ is known. The controller is not taking into account the cars presence - meaning that the presence $z_i[k] = 1 \ \forall k, i$.

We would like to minimize the cost of charging the EVs with a penalty if the car is very empty, so our cost function would be:

$$\min_{u,x} \quad \sum_{k=0}^{H-1} \sum_{i=0}^{N-1} (c[k] \cdot u_i[k] + g(x_i[k])) \tag{2}$$

Under the constraints:

$$x_i[k+1] = x_i[k] + u_i[k] \tag{3a}$$
$$0 \leq u_i[k] \leq 1 - x_i[k] \tag{3b}$$
$$\sum_{i=0}^{N-1} u_i[k] \leq U_{max} \tag{3c}$$
$$x_i[0] = x_i^{init} \tag{3d}$$
$$x_i[H] = 1 \tag{3e}$$

The minimization of the cost is subject to system and vehicle-imposed constraints. The first (Equ. 3a) describes the equation of battery charging (x being the SOC of the cars plugged at each node). Then, the equations (Equ. 3b and 3c respectively) constrain the control input to ensure that the SOC doesn't exceed 1 and that the sum of the power drawn over a time step across all chargers does not exceed the maximum permissible power that can be drawn from the grid/transformer. Also, we assign the initial SOC and at last, we would like to fully charge the electrical vehicles at the last time step, constraint described by (Equ. 3e).

Results and Comparison

Optimization using CVX:

The problem defined in 3.1 is a linear problem that can be numerically solved and estimated using CVX (Appendix 1).

We run the simulation for 3 chargers in a station and plot the SOC of the cars, the cumulative costs as well as the energy provided at every node (Figure 8).

Optimization using Dynamic Programming:

Taking into account that the charging level for an electric vehicle is discrete, and that we would like to include more complexity in our model (stochasticity), a logical step to be taken after using linear programming to solve the problem is to use dynamic programming.

For that matter, we write the principle of optimality as follow:

$$V_k(x_k) = \min_{u[k]} \Big(c[k] \cdot u[k] + g(x[k]) + V_{k+1}(x_{k+1}) \Big) \tag{4}$$

with

$$V_N(x_N) = g(x_N) \tag{5}$$

We implement the following equation numerically for all the chargers of the station, and we obtain the plots in Figure 8.

Comparison of the two methods

The results of both methods are very close, the energy allocated to the cars in each node has the same shape, we can see that for the linear programming using cvx, the algorithm charges the station in order of the lowest SOC giving more energy to the station 1 - in our case. However, we can see more random charging in the algorithm that uses dynamic programming as it found a charging optimum that respects the constraints and the optimality principle.

We also compare the cost and computational time cost of the two methods and find that:

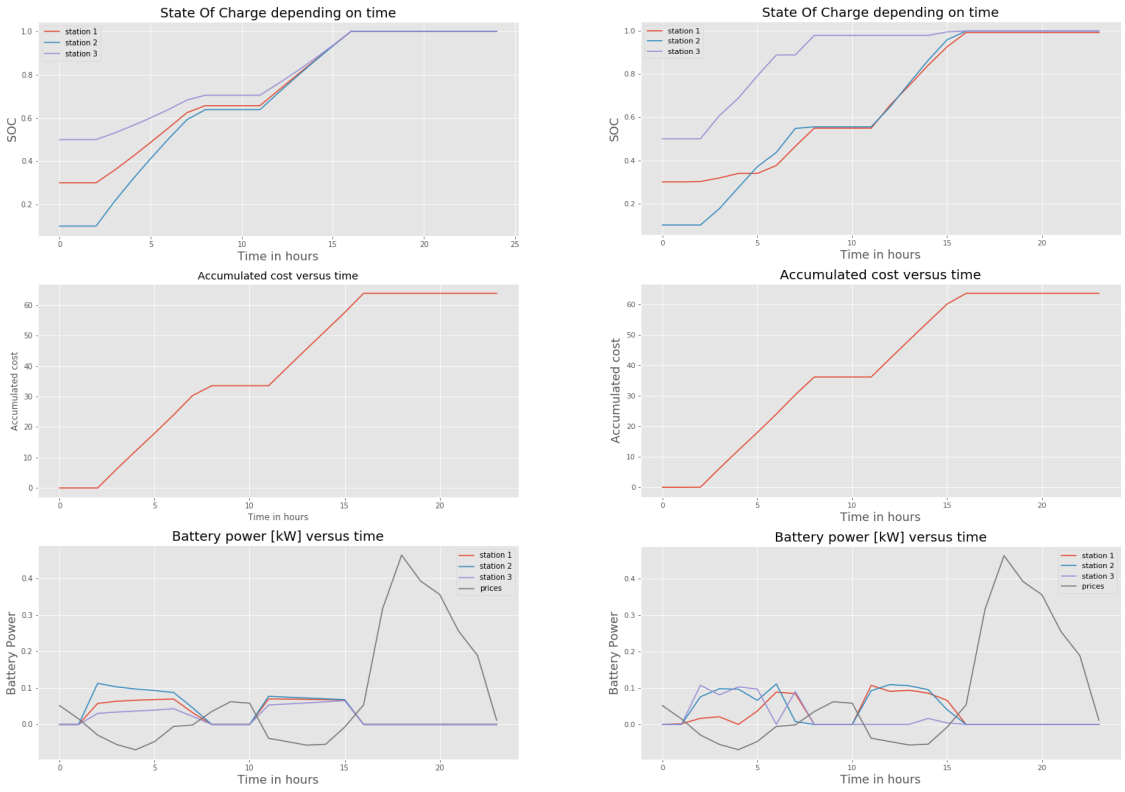


Figure 7: Plots of the simulation results for the simple model using linear programming (plot a on the left) and using dynamic programming (plot b on the right)

	Cost in USD	Computational Cost in seconds
Linear Optimization	63.9	0.27
Dynamic Programming	63.7	28

Table 1: Comparison between the two numerical methods in cost and cimputational cost

5 Model 2: Deterministic with changing car presence

Optimization formulation

As stated before, in reality the presence of cars in the station would vary throughout the time horizon. For this reason, we consider an intermediate model in which the presence of the cars is taken into account.

The only equation that would change from the previous model would be:

$$x_i[k+1] = z_i[k+1] \cdot \Big(x_i[k] + \cdot u_i[k] \Big) \tag{6}$$

With $z_i[k]$ the presence of a car at node i at time k. The optimization program can be written as follow:

$$\min_{u,x} \quad \sum_{k=0}^{H-1} \sum_{i=0}^{N-1} c[k] \cdot u_i[k] \tag{7}$$

Under the constraints:

$$x_i[k+1] = z_i[k+1] \cdot (x_i[k] + u_i[k]) \tag{8a}$$
$$0 \leq u_i[k] \leq 1 - x_i[k] \tag{8b}$$
$$\sum_{i=0}^{N-1} u_i[k] \leq U_{max} \tag{8c}$$
$$x_i[0] = x_i^{init} \tag{8d}$$
$$x_i[H] = 1 \tag{8e}$$

Results and Comparison

Using the same logic as previously, we solve this problem numerically using cvx and using the same DP algorithm as before and including the presence in the simulation.

We choose the following presence of our cars, in two nodes:

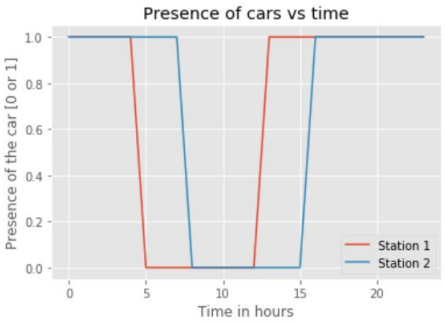


Figure 8: Presence of the cars at two different stations

We plot the results of the simulation for both methods for this second model:

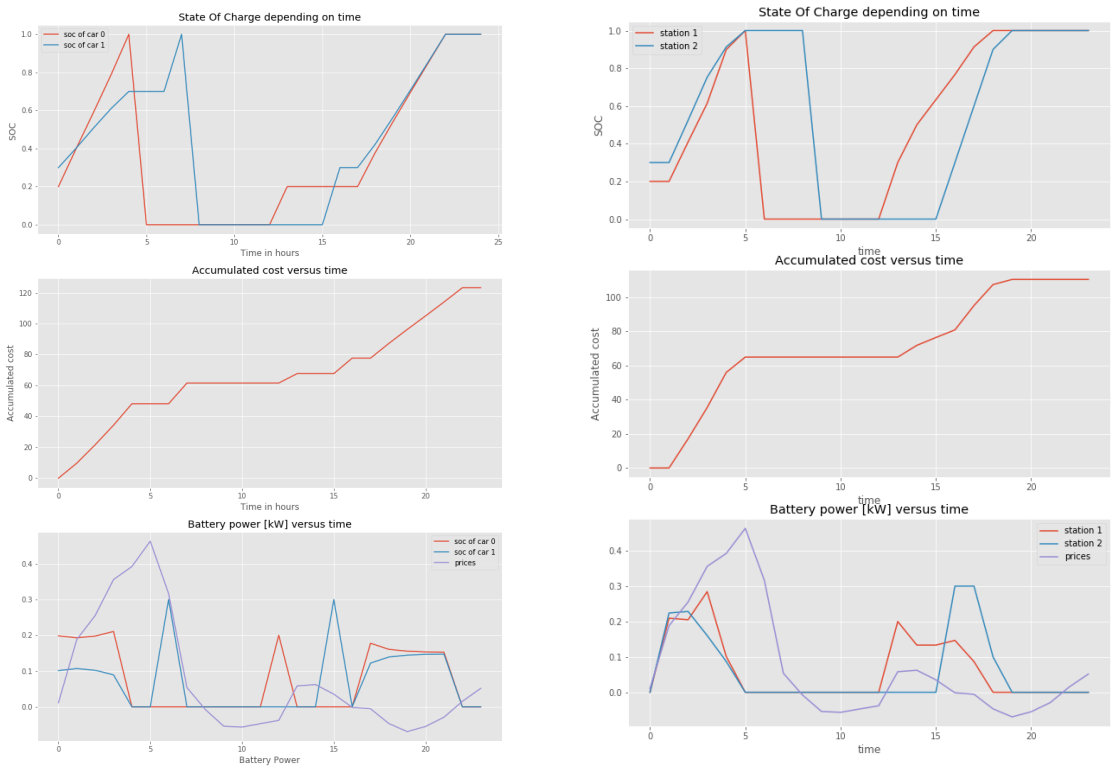


Figure 9: Plots of the simulation results for the second model (includes the presence) using CVX (plot a on the left) and then using dynamical programming (plot b on the right)

The advantage of dynamic programming in this case, is that it allows the use of a **satisfaction function** in the optimization function, instead of the hard constraints that the SOC needs to be full at the time where the car leaves.

We still witness a difference in computational cost between the use of cvx and dynamic programming, which increases when we increase the number of node or chargers in the station. (this is one of the reasons we choose to plot the simulation with 2 nodes with 2 cars each in the station).

Here is a table summarizing the difference of costs between the two methods (which does not imply the superiority of a method based on the numerical results only, because each has specific advantages - for example the possibility to add non linearity to the dynamic programming).

	Cost in USD	**Computational Cost in seconds**
Linear Optimization	123	0.3
Dynamic Programming	110	4.8

Table 2: Comparison between the two numerical methods in cost and computational cost

6 Model 3: Stochastic Model Predictive Control

Optimization formulation

The objective function can be expressed as:

$$\min_{u,x} \quad \mathbb{E}\Big(\sum_{k=0}^{H-1}\sum_{i=0}^{N-1} c[k]\cdot u_i(z_i[k]) + g(x_i[k], z_i[k])\Big) \qquad \forall i,k \tag{9}$$

Under the constraints:

$$z_i[k] \in 0,1 \tag{10a}$$
$$x_i[k] = 0 \quad \text{for} \quad z_i[k] = 0 \tag{10b}$$
$$x_i[k+1] = x_i[k] + u_i[k] \quad \text{for} \quad z_i[k] = 1, z_i[k+1] = 1 \tag{10c}$$
$$x_i[k+1] = x_i^{init}[k+1] \quad \text{for} \quad z_i[k] = 0, z_i[k+1] = 1 \tag{10d}$$
$$u_i[k] = 0 \quad \text{for} \quad z_i[k] = 0 \tag{10e}$$
$$g(x_i)[k] = 0 \quad \text{for} \quad z_i[k] = 0 \tag{10f}$$
$$0 \le u_i[k] \le 1 - x_i[k] \tag{10g}$$
$$\sum_{i=1}^{N-1} u_i[k] \le U_{max} \tag{10h}$$
$$x_i[0] = x_i^{init}[0] \tag{10i}$$
$$p_{m,n}[k] = Pr\big[z_i[k] = m \mid z_i[k-1] = n\big], \quad \forall m,n \in 0,1, \quad k = 0,\dots,N-1 \tag{10j}$$

From the above, the State of Charge $x_i[k]$ depends on the presence $z_i[k]$ and (10b) and (10c) outline this dependency. When the vehicle returns (presence transition from 0 to 1) it is given an arbitrary state of charge denoted by $x_i^{init}[k]$ as constraint (10d) shows. There can be no control input when the vehicle is not present which (10e) accounts for. All other constraints assume the same physical meanings as outlined in previous models.

Value Function

Let $V_k(x(k))$ denote the minimum cost from time step k to terminal time step N, given the present State of Charge $x(k)$ which depends on the vehicle presence $z(k)$.

Bellman's Principle of Optimality

In developing our controller, we consider the expected value function at every time step and by solving backward in the time horizon, we determine the optimal control inputs, u. In this case, the state variable x, the control input u, and the satisfaction function g are all dependent on the presence, $z(k)$ which is modeled as a Markov process. Thus, the Expected value function at time [k + 1] can be determined for the different possible values of z, x and weighted by the probability of transitioning to these states.

$$V_k(x(z[k])) = \min_{u[k]}\big\{c[k]\cdot u_i(z[k]) + g(x[k], z[k]) + \mathbb{E}[V_{k+1}(x(z[k+1]))]\big\}$$

$$V_k(x(z[k])) = \min_{u}\big\{c[k]\cdot u_i(z[k]) + g(x[k], z[k] = Z^m) + \sum_{m,n\in 0,1} p_{m,n}[k+1]V_{k+1}(x(z[k+1]))\big\}$$

The boundary condition is the same as in the previous model:

$$V_N(x_N) = g(x_N)$$

Results and Comparison

We first plot the results for one day simulation of the model 1 using DP and model 3 using SDP with the same presence scenario:

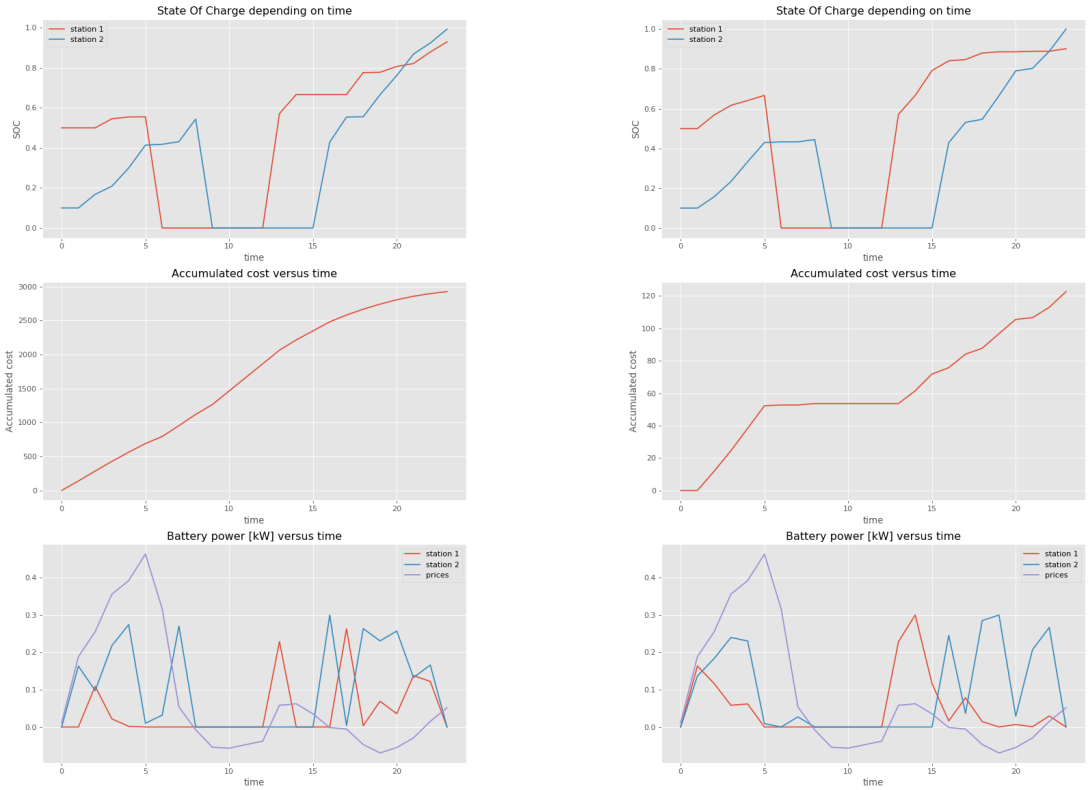


Figure 10: Plots of the simulation results for the first model using DP (left) and the third model using SDP (right)

We can see that the two models are performing well, but to be able to really compare the two controllers we have to run a Monte Carlo simulation. We did it using 2000 scenarios from Pecan Street cars presence profiles.

In addition to the comparison between the two controllers, we computed a "naive" controller which charges directly when the car arrives. The results are presented in the following figure:

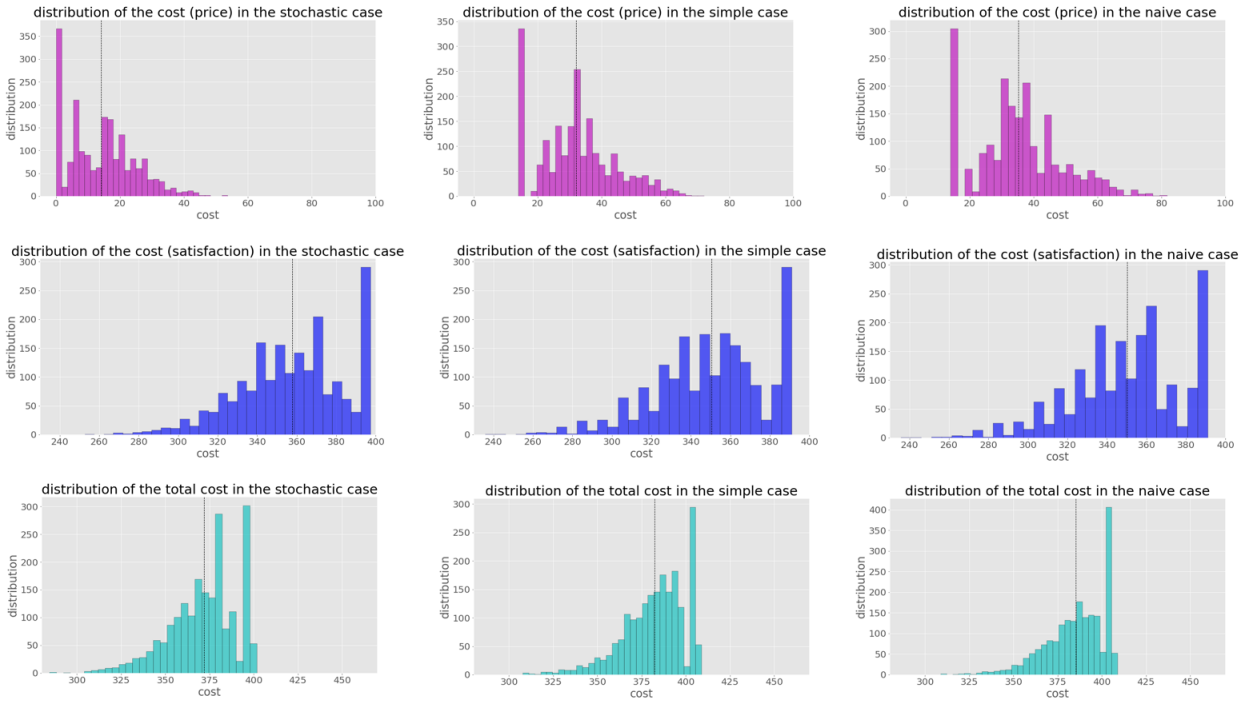


Figure 11: Monte Carlo simulation

We can see that our stochastic model is much better that the naive model but only a little better that the simple model. Indeed, the simple model is already taking into account the satisfaction of the customer as well as the electricity prices. And both controllers are better that the naive one which is not taking the prices into account.

Also, our stochastic controller adds a constraint (the presence of the car) which is decreasing a lot the price but increases the satisfaction cost. Indeed, the stochastic controller can make the car waiting more before being charged since it knows if the car will probably leave or not.

Executive Summary

This project aimed to develop a robust controller for a residential EV charging station. The controller minimizes the EV owner's charging costs while ensuring a level of satisfaction which is dependent on the vehicle's State of Charge at any time.

Our results show that Linear Programming is a reasonable controller and has a short computation time. But it has two main problems: it cannot handle discrete charging levels and cannot take the presence of cars into account. Dynamic Programming on the other hand is a better controller because it can give discrete charging levels but takes a lot of computation time.

Because prices are correlated to the demand/offer ratio, controlling the use of energy depending on prices has a direct effect to reduce the "duck curve" . It is the difference

between the low demand and high generation during the middle of the day and with the high demand low generation at the end of the day.

As we saw using stochastic dynamic programming might be a good solution but the curse of dimensionality and the changing environment might lead to the use of Markov Decision Process. It uses Machine Learning approach with a closed loop system to improve the controller depending on a measurement of its result.

[L]*APPENDIX*

Appendix

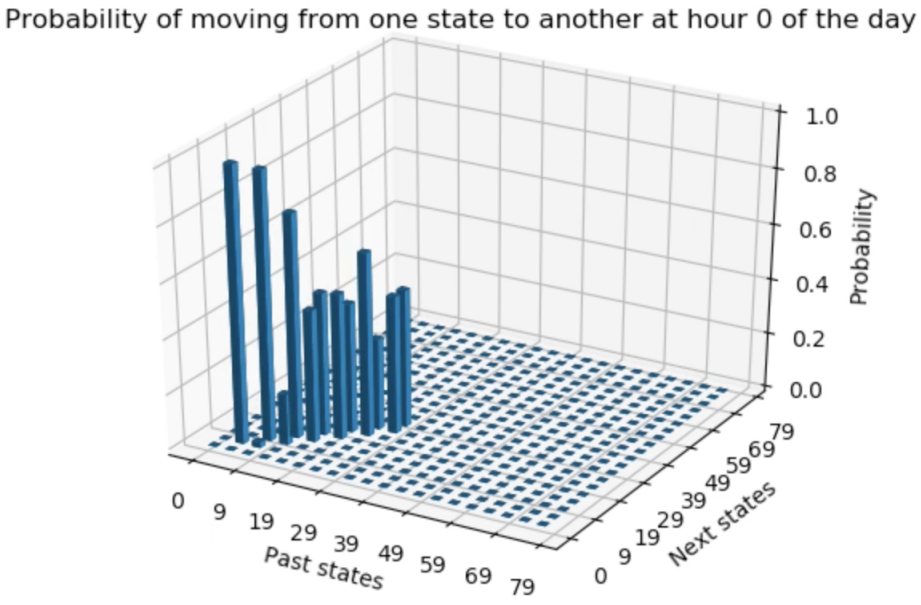


Figure 12: Markov Chain Model for electricity prices

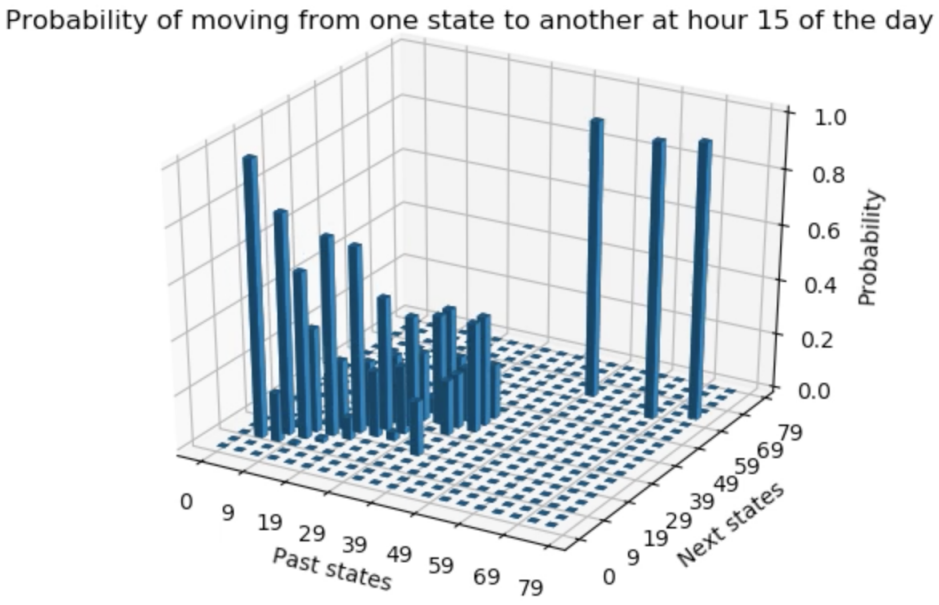


Figure 13: Markov Chain Model for electricity prices

Acknowledgments

We would like to thank our Professor **Scott Moura** of eCal at UC Berkeley for his guidance throughout the project and in the Energy Systems and Control course in general. Specifically, we appreciate his suggestions on best methods with which to approach our problem which allowed us to maximize the skills we gained from the course as well as develop a deeper understanding of systems design techniques.

Also, we would like to thank our Graduate Student Instructor, **Bertrand Travacca** for the time he took to meet with us regularly and discuss our progress in the project. He has been a great guide and was especially helpful in helping us formulate the problem which ensured the relevance of our focus of study.

[L]

About the Authors

Ilias Atigui is a Master of Engineering student in the Civil & Environmental Engineering Department at UC Berkeley completing his degree in May 2018. During his undergrad in France he experienced an overdose of theoretical Mathematics & Physics which convinced him to apply at UC Berkeley in order to get some different kind of Engineering experiences.

Mathilde Badoual is a Master of Science student in the Civil & Environmental Engineering Department at UC Berkeley. She has a bachelor in Electrical Engineering and Computer Science. She has a focus on control of energy systems and integration of renewables. Starting a PhD program at eCAL on this topic.

Salma Benslimane is a Master of Engineering student in the Civil & Environmental Engineering Department at UC Berkeley completing her degree in May 2018. After studying in the Preparatory Classes she decided to dive deeper into Aeronautical and Aerospace domain in which she received her bachelor, before applying at Cal to focus on Systems Engineering.

Zuwa Oriakhi is a Master of Engineering student in the Civil & Environmental Engineering Department at UC Berkeley completing his degree in May 2018. He is specializing in Systems Engineering with a focus on Energy systems and has a Bachelor's Degree in Electrical Power Systems Engineering. He is passionate about building sustainable energy infrastructure and facilitating the grid-level integration of distributed resources.

Part III

Building Energy Management

Predicting Building Electric Loads under Climate Change

Megan Dawe, Samuel Fernandes, Jin Pan, Julia Szinai

Abstract

Accurately predicting buildings' energy usage serves a variety of purposes, including utility load forecasting, building energy management, and energy efficiency savings estimation. In addition, because climate change is predicted to cause temperature increases, understanding the building energy response to warming will aid planning authorities and utilities in the development of climate adaptation strategies. There is a large body of literature about methods to predict commercial building energy use. Most of these methods use outside air temperature and building characteristics such as square footage and occupancy for their predictions. However, because data on building characteristics is often not publicly available, it is useful to have relatively accurate models of energy usage that rely only on outside air temperature measurements.

High resolution hourly smart meter data, and recent advances in data science and machine learning have produced new statistical forecasting methods that can be applied to building energy prediction. Using a sample portfolio of commercial buildings, the objective of this study is twofold:

1. To test and choose the best predictive model of hourly energy usage for each building with weather data inputs among a variety of publicly available machine learning methods;

2. To use the best predictive model for each building to forecast future energy usage under a variety of climate warming scenarios.

1 Introduction

1.1 Motivation and Background

Commercial and residential buildings currently account for approximately 40 percent of the total energy consumption in the U.S. and emit approximately one third of greenhouse gas emissions [1]. Decreasing energy consumption, specifically electricity, through improved efficiency in buildings can avoid construction of new power plants, reduce grid infrastructure costs, and lower carbon emissions — in addition to saving customers money on energy bills [2]. In parallel to these climate change mitigation efforts, adaptation to expected changes in

climate and weather patterns provides another motivation to accurately predict electricity loads. Accurate predictive modeling tools under climate warming will be necessary to target buildings for energy efficiency improvements and assist in grid-level resource planning and decision-making. Particularly for grid resource planning, if predictions of energy usage in general are not correct and underestimate loads, utilities may have challenges with under-procured generation resources and reliability.

There is a large body of literature about predicting building energy usage, and historically most studies have only used temperature data and traditional regression analysis. These studies have typically predicted monthly or annual energy usage. However, with an increasing amount of available hourly smart meter data from buildings and advanced computational and analytical capabilities, there is potential for improved prediction accuracy, even with just publicly available temperature data. Improved data-driven building energy usage modeling, especially at the hourly time resolution, can provide more dynamic capabilities for building energy management and building-to-grid interactions such as load-shifting for renewable integration, among other applications. Hourly energy usage predictions can also improve the forecast of peak load to help grid planners anticipate peak capacity needs.

1.2 Focus of this Study

The focus of the study is to identify robust models from currently available open-source statistical methods to predict the electric load of a set of commercial buildings, using outdoor air temperature data as an input parameter. Then, using the most accurate prediction method for each building and temperature data downscaled from global circulation models, this analysis forecasts future electric loads under feasible climate warming scenarios, which can enable better utility planning both for building energy management and building-to-grid interactions.

1.3 Literature Review

Numerous efforts have been made to develop, test, and compare methods that predict building energy use, including engineering (e.g. simulation), statistical, and machine learning models. The engineering methods use physical principles to calculate thermal dynamics and energy behavior on the whole building level or for sub-level components [3]. Several elaborate simulation tools have been developed based on the engineering methods including DOE-2, EnergyPlus, BLAST and ESP-r. They generate accurate results effectively, but require details of building and environmental parameters as inputs, which are usually difficult to obtain. Although some simplified methods have been put forward, their application is limited due to such simplification [3].

Statistical regression methods correlate energy consumption with influencing variables [4, 5, 6]. These empirical methods are developed based on the historical energy data and are widely used to predict the future energy usage over simplified variables, including climatic variables [3]. Bauer and Scartezzini proposed a simplified method to calculate heating and cooling load simultaneously [5]; Dhar et al. take outdoor dry-bulb temperature as the only weather variable to calculate the heating and cooling load in commercial buildings [7]; Lei and Hu further showed that a single variable linear model is sufficient and practical to

model the energy use in hot and cold weather conditions [8]. Autoregressive Integrated Moving Average (ARIMA) model was first derived on the previous data and was then used to predict the next-day load profile [9]. Overall, statistical regression methods are one of the most commonly used method in the building-level energy prediction and forecasting.

The most common machine learning methods cited in the literature include Artificial Neural Networks (ANN), Support Vector Regression (SVR), autoregressive models, random forest, decision trees, and k-Nearest Neighbors (k-NN) [10]. There are also several proprietary models developed by companies that provide energy prediction and energy management services to building owners. These include those developed by Buildings Alive Pty. Ltd., Gridiu, Inc., Lucid Design Group, and Performance Systems Development of New York, LLC. In addition, this study evaluates a publicly available model relying on random forests, MAVE, which was developed at the University of California, Berkeley [11].

Artificial Neural Networks (ANN) are very widely used in the application of building energy usage prediction, especially for hourly prediction. They are effective at solving non-linear problems and thus can be applied to some very complex conditions with a variety of variables. Moreover, ANN are demonstrated to have higher performance than engineering methods in estimating appliance, space cooling, and lighting energy consumption [12]. However, ANN usually requires data pre-processing techniques to reduce noise or useless variables [3].

Support vector machines (SVM) are highly effective in solving nonlinear problems with only small quantities of training data required. The performance of SVMs can be better than the traditional neural networks, especially for hourly prediction [13]. However, the training process of SVM becomes extremely slow when the training data size is large [14].

Another publicly available statistical model, Mave, which was developed by the Center for the Build Environment, is a tool based on the machine learning methods for automated Measurement and Verification (M&V) [11]. M&V is a process used to quantify energy savings from an energy efficiency measure, typically by estimating the counterfactual energy usage of the building, but-for the implementation of the energy efficiency measure, compared to the actual post-implementation usage [15]. M&V 2.0 techniques use machine learning to improve the processing speed and accuracy of the counterfactual estimation. Mave uses a random forest model to estimate the counterfactual energy usage.

Based on the findings from Burger and Moura, even among the sophisticated machine learning methods described above, no one model is the "best" for all buildings or scenarios [16], and some buildings are inherently difficult to predict because of changing business processes, occupancy, thermostat setpoints, and so on [17]. Model selection depends on the scale of the study or project goal. For instance, studies that focus on individual building energy load predictions with substantial building details may determine certain models are more appropriate than those with the goal of portfolio-level predictions and limited building-level data, such as experienced by utility forecasting efforts. At the building-level scale, the motivations for energy prediction are for fault detection diagnosis (FDD) [10] and to provide baseline estimates to assess energy efficiency intervention savings [18]. At the grid or utility level, the motivation is to assist in load forecasting and utility-scale planning, such as energy efficiency and demand-side management programs [16]. A major difference between the investigations between these two groups is the availability of data and the resources (e.g. time and funding) available for computation. For large-scale energy demand forecasting,

efficient and low-cost methods are preferred.

Studies use various methods to assess and compare goodness of fit for models. Common metrics include root mean squared error (RMSE) [3] (and the normalized version called the coefficient of variation RMSE or CV(RMSE)) and mean average percentage error (MAPE) [16]. The choice of parameters used to fit the model and predict energy use vary between studies and are related to data availability and project goals, as discussed above. Most commonly, models are fit using measurements of electricity demand at some time interval, date, time, and at least one weather variable [10]. Training and testing periods also differ by project based on prediction goals. Most studies have approximately two years' of electricity demand data and use between 12 and 18 months as training, and the rest used for testing [10]. Granderson et al. [18] found that "when the training period was shortened from twelve months to nine, and then to six, there was a gradual degradation in predictive accuracy." Electricity consumption data can be available at 15-minute, hourly, or daily intervals, depending on the prediction goal; that is, daily energy load cannot be used to provide accurate hourly predictions, but hourly data can be used to predict daily loads.

For this project, we rely on examples in the literature that most accurately predict building energy use with the least number of commonly available features. The following are key studies to support the analysis methodology:

Table 1: Review of Building Energy Prediction Studies

	Prediction	Parameters	Train Data	Test Data	Model	Performance Metric
Study 1 [16]	Hourly load	(1) Electric demand (2) Time (3) Electric Demand & time (4) Electric demand, time, & outside air temp	*Buildings:* 8 commercial, 24 residential. *Energy Data:* 18 months for batch, 3 months for moving horizon	*Buildings:* same as train *Energy Data:* 6 months for batch, 21 months for moving horizon	(1) Ridge (2) SVR (3) Decision Tree (4) k-NN	MAPE
Study 2 [18]	Daily load	(1) Outside air temp (2) Date (3) Time	*Buildings:* 537 commercial. *Energy Data:* 12 months	*Buildings:* same as train *Energy Data:* 12 months	10 models	NMBE; CV(RMSE)
Study 3 [10]	Daily load	(1) Day of year (2) Week of year (3) Weekday (4) Occupancy (5) Heating degrees	*Buildings:* 1 commercial *Energy Data:* 21 months	*Buildings:* same as train *Energy Data:* 3 months	Gaussian Process	Training accuracy & Validation accuracy

Lastly, we rely on some of the techniques used from the econometric analysis of Auffhammer and Aroonruengsawat [19], which forecast the building load impacts of climate change under various emissions scenarios, as the basis of the forecasting portion of this analysis.

Auffhammer and Aroonruengsawat fit coefficients to historical temperatures and other predictors and use new temperature data from downscaled global circulation models to forecast future energy consumption in different California regions.

1.4 Key contributions

This study contributes to the literature by comparing among a unique ensemble of machine learning methods, and applying the most appropriate method to a long-term forecast of building loads with new temperature forecasts that reflect climate warming scenarios.

2 Technical Description

2.1 Data Selection and Pre-Processing

Building Load Data Selection: A common challenge for building energy load forecasting is the lack of publicly available high resolution data, such as on building characteristics, square footage, and hours of operation or occupancy. We selected a dataset that had multiple buildings and high time resolution because the analysis objective is to develop a model selection and hourly forecasting methodology that can be adapted for a variety of buildings and employed by a utility planner with only access to building load and temperature data. We obtained a dataset from the Efficiency Valuation Organization (EVO) that includes 15-minute electricity consumption and outdoor air temperature over a two year period (2015 and 2016) for 277 commercial buildings in Fort Collins, Colorado [20]. The dataset does not provide any additional information about the buildings. The electricity consumption data is aggregated to hourly electricity consumption for this study.

Building Load Data Pre-Processing: We conducted a number of data-cleaning steps on the building data. First, of the total 277 commercial buildings, we selected a random subset of 50 buildings to use as our data for this analysis to maintain a reasonable computational time for model fit and forecasting. Second, we removed 4 buildings from the set of 50 because of many instances of missing data. Next, to account for load fluctuations unrelated to weather, we added predictors for day of week, month of year, and hour of day. We converted these categorical date-time variables into "dummies" through "one-hot encoding", removing the first binary variable of each category to prevent linearly dependent columns [21].

In addition, we applied scaling factors to the temperature and load data for each building based on the maximum values for each variable in the training data such that all variables had values between 0 and 1. Scaling is particularly important for regularization so that the shrinkage term λ will treat all variables on the same scale and not penalize variables with larger units [21]. We record the scaling parameters used for each building and variable in order to later inverse the scaling when calculating the predicted values and forecast values into original units. The scaling parameters are calculated on the training data and are then applied to the test and forecast values.

Climate Data: In order to forecast building energy load under future climate scenarios, we downloaded maximum and minimum daily temperatures that have been statistically

downscaled using the Bias Corrected Constructed Analog (BCCA) method [22] for the Colorado region from a number of Global Circulation Models that are part of the most recently available Coupled Model Intercomparison Project (CMIP5) ensemble for the IPCC Fifth Climate Assessment. The results of the climate models are archived and made available through a consortium of organizations including USGS, Bureau of Reclamation, Army Corps of Engineers, and Scripps Institute of Oceanography, among others [23]. The available data is for four scenarios that reflect different levels of emissions pathways and resulting climate forcing, which are called Representative Concentration Pathways (RCPs) ranging from the most aggressive climate mitigation scenario of RCP 2.5, to the least mitigation, or business-as-usual (BAU), RCP 8.5. There are also two mid-case scenarios, RCP 4.5 and RCP 6.0 [24].

Climate Data Pre-Processing: Our query of the CMIP5 climate models yielded outputs from 132 different models and model variants, for each year queried. We selected outputs for the mid-century time period, 2045 - 2065 when warming effects are expected to first be noticeable [24], and which is still within a reasonable lifetime of a building existing in 2015, which is our training year. The results are 20 time series of maximum and minimum daily temperature for all climate models. First, in order to account for variability between climate models, we averaged the daily outputs for each year across all the models. Next, we converted the daily temperature extremes to hourly temperatures, because our forecasting step required hourly time resolution data to match test and training data. We use the imposed offset/diurnal temperature conversion methodology broadly outlined by Guan, L. [25]. We follow a specific functional form for daytime and nighttime hourly temperatures devised by Linvill [26], simplified from Parton and Logan [27], that only requires the latitude and time of day.

For daytime hours, we converted to hourly data with a truncated sine function:

$$T(t) = (T_{max} - T_{min}) * sin[\frac{(\pi * t)}{(DL + 4)}] + T_{min} \tag{1}$$

where $T(t)$ is temperature at hour t after sunrise, T_{max} and T_{min} are daily temperature extremes, and DL is day length in hours.

For nighttime hours, we converted to hourly data with an exponential decay function:

$$T(t) = T_s - [\frac{(T_s - T_{min})}{(24 - DL)}] * ln(t) \tag{2}$$

where $T(t)$ is temperature at hour t after sunset, T_s is the sunset temperature from (1), T_{max} and T_{min} are daily temperature extremes, and DL is day length in hours. We used an R package called "chill R", which has created some functions to automate this conversion, including the calculation of the day length from the latitude.

To test the accuracy of this method, we construct hourly temperatures from observed 2015 temperatures and these estimated temperatures are only off from the observed data by 1.5 to 2 degrees Celsius, on average, each month. For the purposes of building simulation, we found this to be an acceptable accuracy.

Finally, for our temperature forecasts, to smooth out any outlying year, we average the resulting hourly temperatures for 2045-2065 to produce one year of hourly temperature data for each RCP, representing mid-century warming scenarios under different emissions futures

for Colorado. The temperature forecasts compared to 2015 temperatures are illustrated in Figure 1.

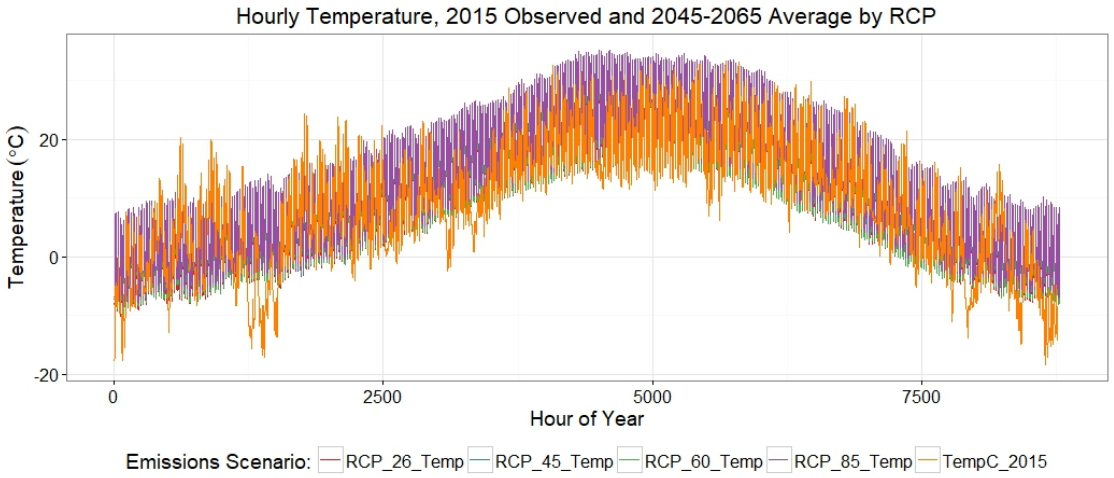


Figure 1: Historical vs. Forecast Hourly Temperatures

2.2 Model Types

Based on the literature review, we selected five candidate machine learning methods that were found to have reasonable computational times and predictive performance: (1) Mave, which is based on random forest and developed at UC Berkeley's Center for the Built Environment [13], (2) Ridge Regression, (3) Artificial Neural Network (ANN), (4) K-Nearest Neighbor (k-NN), and (5) Autoregressive with eXogenous inputs (ARX).

(1) Mave Mave uses a random forest machine learning technique with the inputs and outputs that can be set according to the available training data. Decision trees are popular because of their simplicity and interpretability. Random forests are an ensemble learning methodology where the performance of a number of weak learners is "boosted."

With a RF, there is an ensemble of C trees $T_1(X), T_2(X), ..., T_C(X)$, where $X = x_1, x_2, ..., x_m$ is a m-dimension vector of inputs. The resulting ensemble produces C outputs $\hat{Y}_1 = T_1(X), \hat{Y}_2 = T_2(X), ..., \hat{Y}_C = T_C(X)$. $\hat{Y}_C$ is the prediction value by decision tree number C. The output of all these randomly generated trees is aggregated to obtain one final prediction $\hat{Y}$, which is the average values of all trees in the forest. The decision trees in this project only consider outdoor air temperature and the previous value of energy consumption as input variables. Both these features were allowed to be tried in an individual tree, and the maximum depth of a tree is restricted to 4.

(2) k-NN The k-NN model uses the scikit-learn KNeighborsRegressor in Python, which is a regression based on k-nearest neighbors [28]. The model uses month of year, day of week, hour of day, normalized outdoor air temperature, and normalized hourly electricity data for training. The prediction uses the 15 closest neighbors ($k = 15$), based on Euclidean distance, and uniform weights, meaning all 15 nearest neighbors are weighted equally instead of weights by distance.

(3) Ridge Regression The ordinary least squares with L_2 regularization (aka ridge regression) fits a linear model while penalizing the size of parameters based on the L_2 norm.The input training variables (predictors) for ridge regression contain the scaled temperature, the dummy month of year, day of week and hour of day. A cross validation was conducted based on one building's data to determine the weighting term for the regularization penalty, and this weighting term was applied to the rest of the buildings. In this case, the weighting term $\lambda = 0.6$.

The least squares minimization model is given by:

$$\min_{w} \sum_{i} ||w^T x_i - y_i||_2^2 + \lambda ||w||_2^2 \tag{3}$$

where x_i are the predictors for each hour, w are the weights for each predictor. The model is fit with the closed form solution: $w^* = (X^T X + \lambda I)^{-1} X^T Y$

(4) ANN An Artificial neural network stores knowledge from past observations and makes it available for the prediction of future values. An ANN uses hidden layers where the number of hidden layers depends on the nature and complexity of the problem. ANN's do not require any information about the system as they operate like black box models and learn relationship between inputs and outputs. The schematic diagram below shows a feed-forward neural network architecture, consisting of two hidden layers. While there are different neural network strategies in literature e.g. feed-forward, Hopfield, Elman, self-organizing maps, and radial basis networks, this study uses a feed-forward neural network and back-propagation algorithm. The algorithm is summarized as follows:

1. Use the training data to propagate through the neural network to obtain the desired output.

2. Initialize weights using small threshold values.

3. Calculate input to the j-th node in the hidden layer using the equation below:

$$net_j = \sum_{i=1}^{n} w_i j x_i - \theta_j \tag{4}$$

4. Calculate output from the j-th node in the hidden layer using the two equations below:

$$h_j = f_h(\sum_{i=1}^{n} w_i j x_i - \theta_j) \tag{5}$$

$$f_h(x) = \frac{1}{1 + \exp(-\lambda_h x)} \tag{6}$$

5. Calculate the input to the k-th node in the hidden layer using equation below:

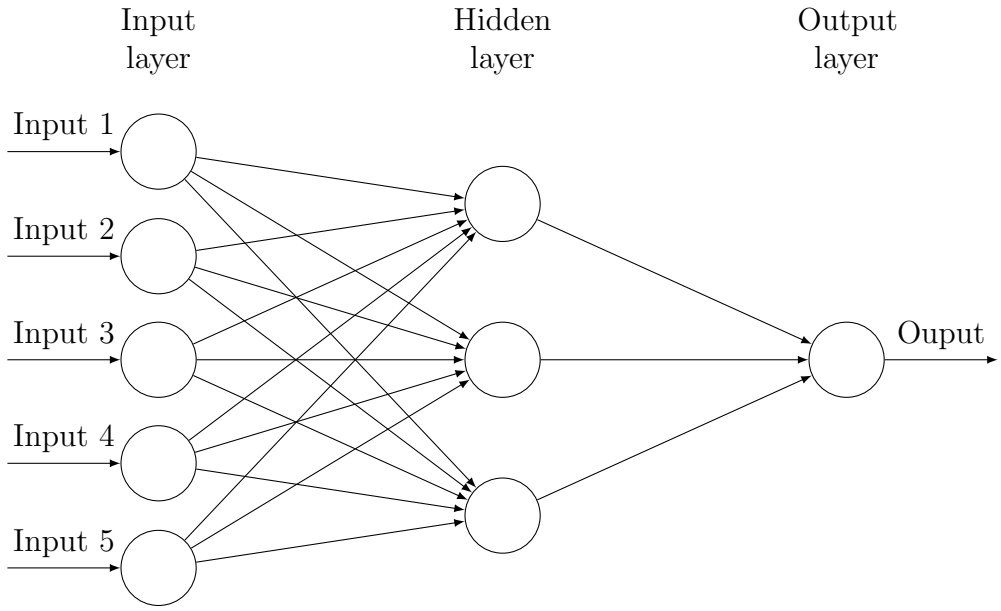
$$net_k = \sum_{j=1}^{n} w_k j x_j - \theta_k \tag{7}$$

6. Calculate the output of the k-th node of the output layer using equation below:

$$y_k = f_k(\sum_{j=1}^{n} w_k j x_k - \theta_k) \tag{8}$$

$$f_k(x) = \frac{1}{1 + \exp(-\lambda_k x)} \tag{9}$$

Schematic diagram of a feed-forward artificial neural network used is shown below.



(5) ARX An Autoregressive with exogenous Inputs Model combines information from a previous hourly observation with an exogenous input. In this case, the exogenous input is the average hourly load for a given hour of day and day of week, to capture the load variability unrelated to temperature. Rather than using the prior hours load data for the autoregressive component of the model, we use the prior hours of temperature data. This modification is in anticipation of the forecasting step; when forecasting into the future under climatic conditions, we do not have prior load data and only have new temperatures as inputs. This formulation also has a physical interpretation: the current energy loads depends on how much heating or cooling the building conducted in the previous hours in response to temperature.

The ARX model is given by:

$$\hat{P}_{arx}(t) = \sum_{l=1}^{L} \alpha_l * T(t-l) + \hat{P}_{avg}(t) \tag{10}$$

where $T(t-l)$ are the temperatures for l prior hours and the $\hat{P}_{avg}(t)$ is the average hourly load for the given hour of day and day of week from the training data. The model coefficients α_l are the weights for each prior hours of temperature, and the number of prior hours L is 3. The model is fit with the closed form solution: $\alpha^* = (X^T X)^{-1} X^T Y$ where X is the matrix of lagged temperatures $T(t-l)$ and Y is the observed hourly load minus the $\hat{P}_{avg}(t)$.

2.3 Model Fit, Model Gating, and Forecasting Methodology

After the data collection and pre-processing steps described in Section 2.1, the following steps and Figure 2 summarize the analysis methodology, which is described in more detail below.

1. Split dataset into 12 months of training and 12 months of testing data.
2. Train each model using appropriate predictors, including: month, day of week, hour of day, hourly outdoor air temperature (scaled by maximum value), and 2015 hourly electric load (scaled by maximum for each building).
3. Use test data to predict hourly electricity loads based on given datetime and normalized outdoor air temperature inputs.
4. Calculate and compare CV(RMSE) and MAPE for each model for each building.
5. The CV(RMSE) gate will select the model that performed best (produced the lowest CV(RMSE)) during the prediction step. Remove buildings which do not achieve at least 25% CV(RMSE) for future load forecasting.
6. Use model with lowest CV(RMSE) from step 5 to forecast hourly electricity load with modified outdoor air temperature inputs from four climate change scenarios.

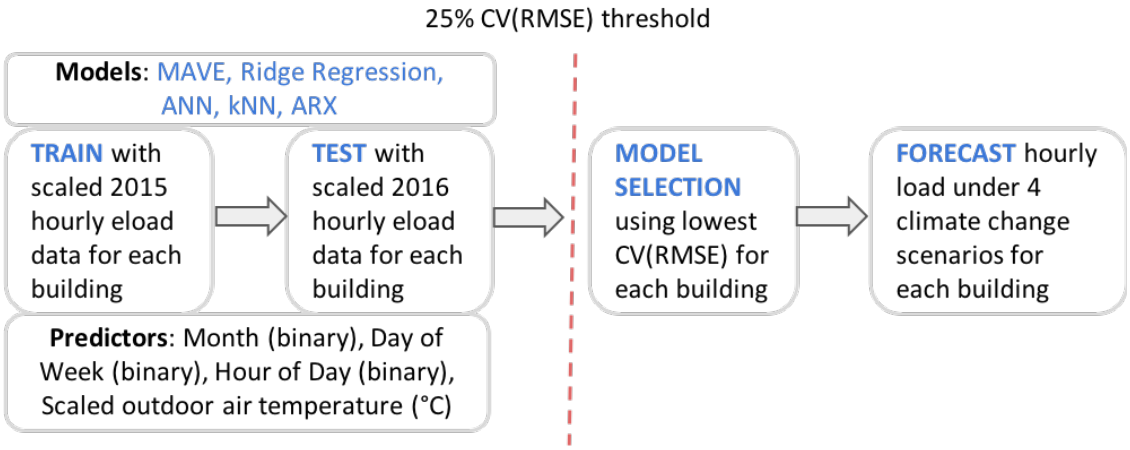


Figure 2: Analysis Methodology

In Step 1 we partition the pre-processed dataset of 46 buildings data into training and testing periods for each individual building, with 2015 data used for training and 2016 data used for testing. Overall the average hourly load and standard deviation of the test data is well-balanced with the training data, as illustrated by Figure 3.

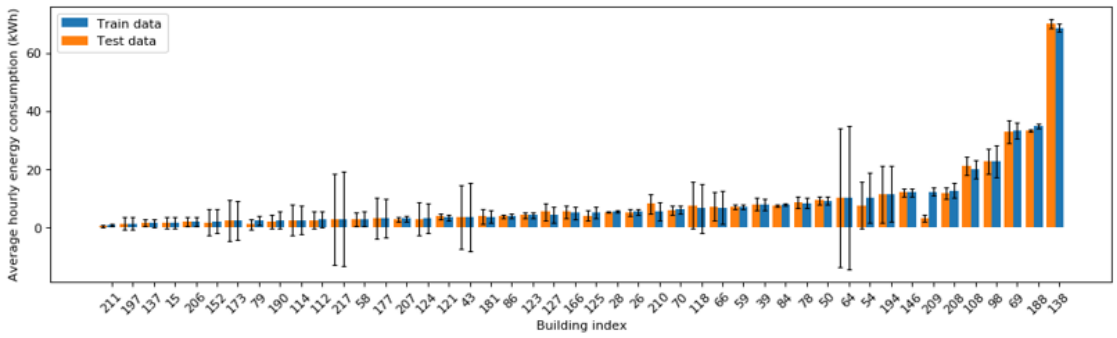


Figure 3: Average and SD Hourly Test and Training Energy Consumption by Building

In Step 2, for each building, we train and fit each of the 5 models described in Section 2.2 using the relevant date, time, and air temperature (scaled by maximum value) predictors. We record the fitted parameters and in Step 3, we use those fitted parameters to predict hourly energy loads for 2016. We inverse the scaling using the saved scaling parameters for each variable to return the predictions to the original kWh units.

In Step 4, we calculate the CV(RMSE) and the MAPE on each model's predictions compared to the test data for each building. The CV(RMSE) is derived by normalizing the RMSE with the mean of the data and has the advantage of providing a unit-less value that allows for performance comparison across models and buildings. The CV(RMSE) and MAPE are defined as follows:

$$CV(RMSE) = \frac{\sqrt{\frac{\Sigma_{i=1}^{n}(y_i - \hat{y_i})^2}{N}}}{\bar{y}} * 100 \tag{11}$$

$$MAPE = \frac{\frac{\Sigma_{i=1}^{n}(y_i - \hat{y_i})}{N}}{y_i} * 100 \tag{12}$$

We then perform a gating process in Step 5 that focuses on CV(RMSE): buildings for which none of the models obtain a CV(RMSE) equal to or below 25% are removed from the dataset for the climate change electricity load forecasting because they are deemed "unpredictable". The 25% CV(RMSE) threshold originates from the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) Guideline for Measurement of Energy, Demand, and Water Savings. Buildings may exceed the threshold for a number of reasons, such as variable building occupancy or some type of change that occurred during the data collection period (e.g. retrofit or change of tenant). Having no knowledge of building occupancy or construction, we removed all buildings which did not meet the threshold; this resulted in removal of 20 of the 46 buildings.

Finally, in Step 6, for each building which meets the threshold, the model with the lowest CV(RMSE) is selected to forecast building load under climate change. To do this, we use the fitted parameters from the best model for each building, and input the forecast hourly temperatures from each of the four RCP scenarios to forecast hourly loads. Implicit in this step is the assumption that the relationship between outdoor air temperature and building load remains constant over time, and that the only change is the outdoor air temperature

under climate change. We do not forecast any building loads with k-NN, because that is a clustering method that relies on observed load data for a time period close to the forecast period and is only appropriate for short-term and stable (i.e. consistent) energy loads.

3 Discussion

Based on the results of the CV(RMSE) 25% threshold and gating process, as seen in Figure 4, it is evident that not all buildings can rely on outdoor air temperature and datetime inputs for accurate electricity load predictions, and that no single machine learning method is the most robust under all situations. Among the models used in the analysis, there is approximately an even split between which model performed the best among the 46 buildings: MAVE (35% of buildings), Ridge Regression (19% of buildings), ANN (23% of buildings), and ARX (23% of buildings). ARX and ANN have prior hours of temperature data as inputs, and did equally better in the predictions. Even though Ridge Regression has the highest inputs utilization rate followed by k-NN, they did not win among other models. Additionally, the MAPE and CV(RMSE) across the buildings are high (greater than 100% in some instances), and approximately half (20 of 46) buildings did not meet the 25% CV(RMSE) threshold for any model. This suggests that outdoor air temperature and datetime inputs do not have a strong enough correlation with electricity loads for all commercial buildings. This could be due to non-weather-dependent loads, such as plug loads and equipment, representing a larger portion of building electricity consumption.

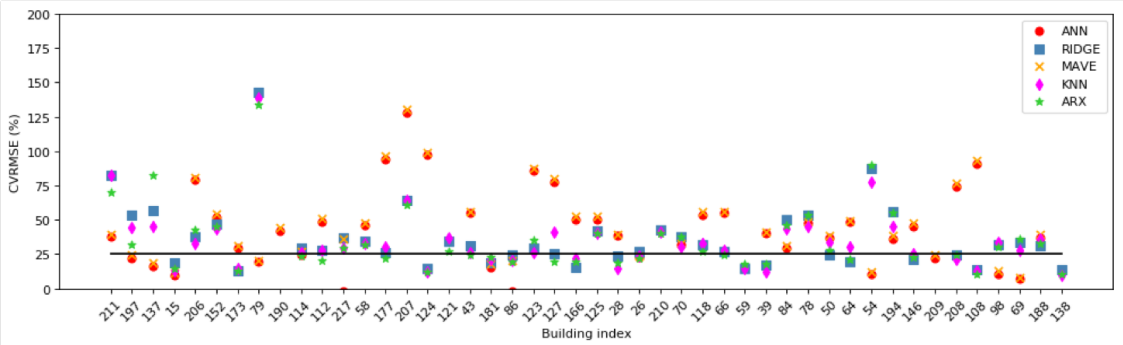


Figure 4: CV(RMSE) for all models by building

For the remaining 26 buildings that met the threshold, the climate change forecasting exercise results show that, on average, building electricity load is predicted to increase in both annual consumption and peak demand under increasing RCP scenarios (i.e. increasing outdoor air temperature). However, there are some buildings that see reduced overall electricity consumption and/or peak demand under the climate change scenarios. Looking into the building-specific data reveals that these particular buildings experienced peak demand during the winter months of the training data, and could be heating-dominated buildings (assuming heating is provided by an electric system), or at least *not* cooling-dominated buildings. Therefore, increasing outdoor air temperatures results in a predicted reduction in loads for those buildings and thus reduce overall electricity consumption. Building 15 in Figure

6 is one of the examples that had lower energy consumption in summer (i.e. warmer outdoor air temperatures) than in Winter, and low annual consumption overall. Under climate warming scenarios, its total energy consumption decreased because the model is trained to interpret warmer temperatures with lower electricity consumption. The other buildings in 6 are examples of predictions for mid and high annual consumption buildings. Without additional knowledge of these buildings' consumption patterns and characteristics, we can only speculate that some buildings will experience lower electricity demand in warming climates due to reduced heating needs, which again highlights the importance of data availability for prediction accuracy.

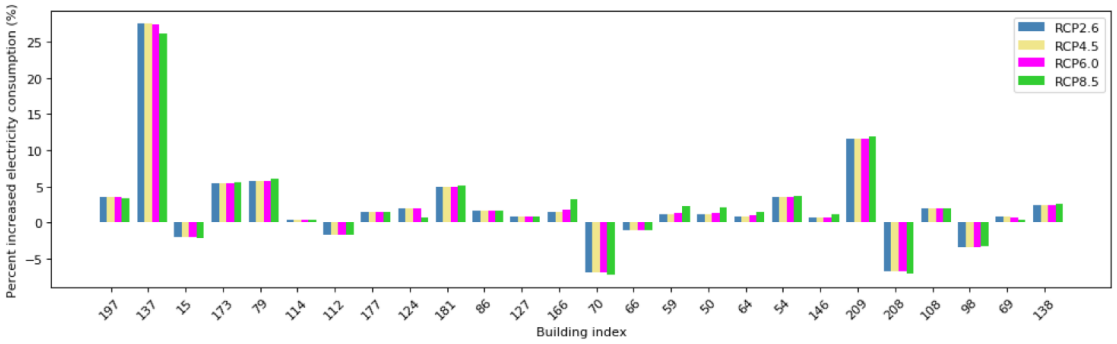


Figure 5: Percent change in total annual building electricity loads

There are a number of limitations of this study. The dataset only included commercial buildings from one zip code in Colorado, and those buildings may undergo retrofits, change in use, or even demolition prior to the forecast period of 2045 to 2065. However, the results more broadly imply that future climate may impact building electricity consumption differently depending on the scenario and underlying building characteristics. Utility planners should examine current buildings' peak demand usage when interpreting forecast results because strategies that are currently in place or being pursued may not be sufficient. Additionally, current building design strategies and codes could incorporate climate forecasting information to more appropriately design buildings and mechanical systems to be resilient or adaptable to changing temperatures and building cooling loads.

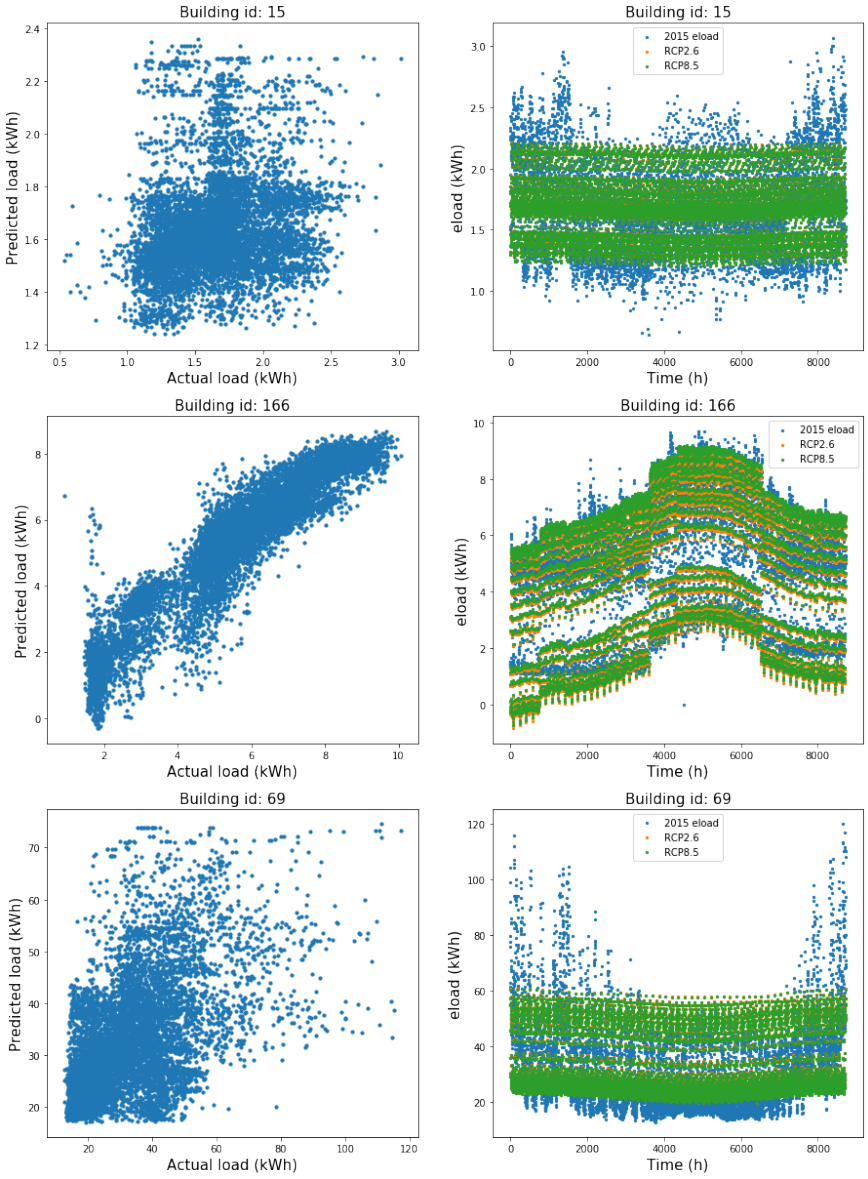


Figure 6: Predicted, actual, and forecasting load for 3 sample buildings

4 Summary and Future Work

While the specific results of this study only apply to a particular climate zone and set of buildings, we demonstrate a method of estimating a range of possible climate impacts on electricity loads for a set of commercial buildings, given publicly available data. The analysis could be repeated elsewhere in the U.S. and help planning authorities or utilities anticipate demand changes and also target energy efficiency measures in areas where the load would increase or change significantly under climate change. Our findings also show that no one particular machine learning method can generally be considered to be better than others, different machine learning models perform differently based on data, conditions

and a number of factors. It is best to either test a number of models and select the best based on a performance metric or use an ensemble of models to forecast building energy consumption. While we did the former, the latter could be an extension of this work.

Other extensions of this work include applying this method to datasets in other cities and to include gas data in the analysis to get a more comprehensive understanding of building energy consumption. This approach could also be used to identify temperature-sensitive buildings that need to be retrofitted in regions that face climate change impacts.

References

[1] U.S. Energy Information Administration [EIA]. (2012). Commercial Building Energy Consumption Survey: Energy Usage Summary, Table 1 (March 2016).

[2] Ettenson, L. & Heavey, C. (2015). California's Golden Energy Efficiency Opportunity: Ramping Up Success to Save Billions and Meet Climate Goals. Natural Resources Defense Council (NRDC) and Environmental Entrepreneurs (E2).

[3] Zhao, H. X., & Magoulès, F. (2012). A review on the prediction of building energy consumption. *Renewable and Sustainable Energy Reviews, 16(6), 3586-3592.*

[4] Westergren, K.E., Högberg, H., & Norlén, U. (1999). Monitoring energy consumption in single-family houses. *Energy and Buildings, 29(3):247–57.*

[5] Bauer, M., & Scartezzini, J.L. (1998). A simplified correlation method accounting for heating and cooling loads in energy-efficient buildings. *Energy and Buildings, 27(2):147–54.*

[6] Pfafferott, J., Herkel, S., & Wapler, J. (2005). Thermal building behaviour in summer: long-term data evaluation using simplified models. *Energy and Buildings, 37(8):844–52.*

[7] Dhar, A., Reddy, T.A., & Claridge, D.E. (1998). Modeling hourly energy use in commercial buildings with Fourier series functional form. *ASME Journal of Solar Energy Engineering, 120:217–23.*

[8] Lei, F., & Hu, P. (2009). A baseline model for office building energy consumption in hot summer and cold winter region. In: Proceedings of international conference on management and service science. p. 1–4

[9] Kimbara, A., Kurosu, S., Endo, R., Kamimura, K., Matsuba, T., & Yamada, A. (1995). On-line prediction for load profile of an air-conditioning system. *ASHRAE Transactions, 101(2):198–207.*

[10] Yan, B., Li, X., Shi, W., Zhang, X., & Malkawi, A. (2017). Forecasting Building Energy Demand under Uncertainty Using Gaussian Process Regression: Feature Selection, Baseline Prediction, Parametric Analysis and a Web-based Tool. `https://doi.org/10.26868/25222708.2017.142`

[11] Raftery, P. & Hoyt, T. (2016). Mave: software automated Measurement and Verification. Center for the Built Environment, University of California Berkeley, `https://github.com/CenterForTheBuiltEnvironment/mave`.

[12] Aydinalp, M., Ugursal, V.I., & Fung, A.S. (2002). Modeling of the appliance, lighting, and space cooling energy consumptions in the residential sector using neural networks. *Applied Energy, 71(2):87–110.*

[13] Li, Q., Meng, Q.L., Cai, J.J., Hiroshi, Y., & Akashi, M. (2009). Applying support vector machine to predict hourly cooling load in the building. *Applied Energy, 86(10):2249–56.*

[14] Zhao, H. X., & Magoulès, F. (2010). Parallel support vector machines applied to the prediction of multiple buildings energy consumption. *Journal of Algorithms & Computational Technology, 4(2):231–49.*

[15] Touzani, S., Granderson, J., & Fernandes, S. (2018). Gradient boosting machine for modeling the energy consumption of commercial buildings. *Energy and Buildings, 158:1533-1543.*

[16] Burger, E. M., & Moura, S. J. (2015). Gated ensemble learning method for demand-side electricity load forecasting. *Energy and Buildings, 109, 23-34.*

[17] Granderson, J., Price, P. N., Jump, D., Addy, N. & Sohn, M. D. (2015). Automated measurement and verification: Performance of public domain whole-building electric baseline models. *Applied Energy 144, 106–113.*

[18] Granderson, J., Touzani, S., Custodio, C., Sohn, M. D., Jump, D., & Fernandes, S. (2016). Accuracy of automated measurement and verification (M&V) techniques for energy savings in commercial buildings. *Applied Energy, 173, 296-308.*

[19] Auffhammer, M., & Aroonruengsawat, A. (2011). Simulating the impacts of climate change, prices and population on California's residential electricity consumption. *Climatic Change, 109(1), 191–210. https://doi.org/10.1007/s10584-011-0299-y*

[20] Efficiency Valuation Organization (EVO). (n.d.). [Efficiency Valuation Organization (EVO)]. Retrieved May 3, 2018, from *https://evo-world.org/en/*

[21] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). Introduction to Statistical Learning, with Applications in R. New York: Springer New York Heidelberg Dordrecht London.

[22] Gutmann, E., Pruitt, T., Clark, M. P., Brekke, L., Arnold, J. R., Raff, D. A., & M., R. R. (2014). An intercomparison of statistical downscaling methods used for water resource assessments in the United States. Water Resources Research, 50(9), 7167–7186. *https://doi.org/10.1002/2014WR015559*

[23] Downscaled CMIP3 and CMIP5 Climate and Hydrology Projections. (2018). Retrieved May 2, 2018, from *https://gdo-dcp.ucllnl.org*

[24] Lukas, J., Barsugli, J., Doesken, N., Rangwala, I., & Wolter, K. (2014). Climate Change in Colorado: A Synthesis to Support Water Resources Management and Adaptation. Western Water Assessment, Cooperative Institute for Research in Environmental Sciences (CIRES), University of Colorado Boulder.

[25] Guan, L. (2009). Preparation of future weather data to study the impact of climate change on buildings. *Building and environment, 44(4), 793-800.*

[26] Linvill, D. E. (1990). Calculating Chilling Hours and Chill Units from Daily Maximum and Minimum Temperature Observations. HortScience, 25(1), 14–16.

[27] Parton, W. J., & Logan, J. A. (1981). A model for diurnal variation in soil and air temperature. Agricultural Meteorology, 23, 205–216. *https://doi.org/10.1016/0002-1571(81)90105-9*

[28] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, É. (2012). Scikit-learn: Machine Learning in Python. arXiv:1201.0490 [Cs]. Retrieved from *http://arxiv.org/abs/1201.0490*

About the authors

Megan Dawe is an MS student in the Building Science program of the Architecture department.

Jin Pan is an MS student in the Energy, Civil Infrastructure and Climate (ECIC) program of Civil and Environmental Engineering.

Samuel Fernandes is a PhD student in the ECIC program of Civil and Environmental Engineering.

Julia Szinai is a PhD student in the Energy and Resources Group (ERG).

Air Conditioning Control System with Prediction of Occupant Flow

Lorna Chen, Yao Han, Sikang Hu, Xinhao Wang,

Abstract

HVAC system is an essential component providing thermal comfort and adequate air quality for individuals in built environment. However, the efficiency of HVAC system still has large potential for improvement. In this project, the aim is to design a responsive air conditioning(AC) system to improve the occupant comfort level and system economic efficiency based on the prediction of occupant flow by applying innovative control law. Traditionally, the response of AC system was defined by whether the room temperature exceeds desired range. In this model, for operating hours, the temperature control would be initiated ahead of time by predicting occupant flow from historical data and optimal indoor temperature would be achieved before occupants' arrival. This model is significant at the level to reduce extreme temperature fluctuation and to maximize thermal comfort hours for occupants.

Introduction

Motivation and Background

The AC system of today has not changed much from that back in 20th century with lack of control and responsive feedback. According to Quadrennial Technology Review from Department of Energy, more than 76% of all U.S. electricity use and more than 40% of all U.S. energy use as well as associated greenhouse gas (GHG) emissions were used to provide comfortable, well-lit, residential and commercial buildings and to provide space conditioning and lighting for industrial buildings. Nevertheless, the comfort level of customer is not closely monitored and energy waste can be prevented through sophisticated control strategies. The challenges for the control and optimization were to balance human comfort level and energy consumptions. In order to address this issue, a responsive system was needed to correctly predict the occupant flow in the built environment and allow optimal thermal experience for the occupants.

The current control strategy for commercial AC system is simply on/off based on whether the room temperature is beyond dead-band. However, there are two potential problems: 1) The dead-band strategy could not prevent the temperature from going out of comfort zone or would keep temperature zone too narrow compared with the comfort zone; 2) It's unnecessary

to keep temperature in break time. Our project aims to improve these two problems through customer flow estimation. With the customer flow estimation, our system can predict the room temperature evolution and keep temperature in comfort zone precisely.

Focus of this Study

The aim of this project was to design a responsive AC system to maximize the occupant comfort level and system economic efficiency based on the predicted occupant flow. From the nature of temperature state equation, dynamic programming (Chapter 5) was adopted in this project.

Literature Review

For the control technology part in AC realm, Zhao and Yu(2015) did a review on the application of advanced control technologies in AC system and mentioned four main technologies: PID(proprtional-integral-derivative) control, artificial neural network, fuzzy control and model predictive control. PID control extensively prevails in the modern AC control system, because implementing the other three would require replacement or modification of existing equipment. Besides, for artificial neural network, increasing computation and data size decreases convergence speed; for fuzzy control, different designers have different classification principles which depend on experience; for model predictive control, large online rolling calculation data causes data distortion and divergence.

For the occupant relevant AC control part, Godo, Haase, and Nishi(2017) proposed an AC control based on pressure sensor in the chair detecting occupant location in a library. Their idea on the control part is pretty simple: if there were occupants detected and the comfort level was below certain limit, the heating temperature would be set to $20°C$, otherwise it would be set to $15°C$. Wang, Feng and other eight researchers(2017) did predictive control of indoor environment based on occupant number detected by video data and CO_2 concentration. They incorporated occupant impact into their model by multiplying a factor decided by sensitivity analysis of cooling load to occupant number. Basically, they used real-time occupant data detected by sensors to adjust the AC on/off period to save energy consumption. Their idea achieved faster AC response and made the indoor environment more stable.

To summarize, the control technology used in the AC industry is fairly classic and there is certainly room for improvement. As for the occupant related control, current achievement is not satisfactory enough. With rapid development of computer technology, advanced building automation era is sure to come, along with a giant leap in AC control strategy.

Key contributions

Our project explored the algorithm-based strategy to optimize AC system control with occupant level in the built environment. The model includes the method of Markov Chain, a predictive and probabilistic model, to determine occupant flow. The model is able to

adjust the AC operation power beforehand, further improving response speed and indoor environment stability.

1 Modeling

1.1 Model Description

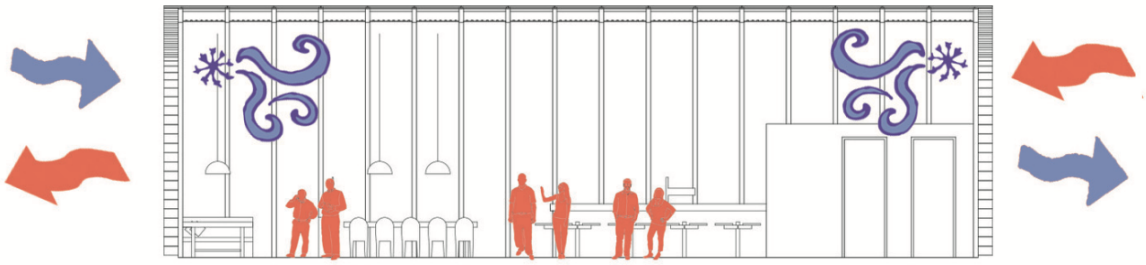


Figure 1: Project Overview

The setting was designated to build a simulation model for temperature control inside a café. Figure.1 above demonstrated the system boundary where three terms changed the observation states, 1) the energy exchange between interior and exterior state, 2) the working power of the air conditioning units and 3) the heat dissipated from occupants inside the space. The space was dimensioned to be 30 ft by 20 ft for length and width.

1.1.1 System Boundary

The system output is room temperature $T(t)$. The system inputs include predicted ambient environmental temperature $T_A(t)$, predicted number of customers $n(t)$ in the café and thermal power removed by the AC unit $P_e(t)$ and the AC mode $s(t) \in \{0, 1\}$ corresponding to off or on.

Notation	Description	Units
$T_A(t)$	predictable input t	$[^\circ C]$
$n(t)$	predictable input t	N/A
$s(t)$	controllable input t	N/A
$T(t)$	performance output t	$[^\circ C]$

The stock is thermal energy stored in the room. The level or state variable, indicating the amount of stored thermal energy, is $T(t)$.

1.1.2 Energy Equation based on Conservation Law

The room temperature $T(t)$ changes due to heat transfer with the outside environment and with the AC unit. The First Law of Thermodynamics, in the form of Newton's Cooling

Law, gives us:

$$C\frac{d}{dt}T(t) = \frac{1}{R}[T_A(t) - T(t)] - s(t)P_e + n(t)P_I$$

where parameter C is the room's thermal capacitance which was taken to be 0.5 $[kWh/°C]$ for this particular size of space. R is the thermal resistance between the room and outside, 25.3 $[°C/kW]$, and P_I is internal heat generation per person, 0.1 kW.

1.2 Data Processing

There were two sets of data we implemented as inputs in the model. One was the outdoor temperature in the surrounding area. The other was the occupant flow or amount of foot traffic entering the space. Both were recorded at hourly instances and then linearly interpreted to the time interval as desired. The temperature data was collected at Climate Data Online from agency of National Oceanic and Atmospheric Administration (NOAA). The document provides a descriptive suite of statistics including the dates and hours when it was collected, clouds coverage percentage for estimation of solar heat power if applicable and lastly the mean air temperature observed by the stations. The temperature data was referenced at the station in Oakland, CA. Due to the limitation of website service, only the statistics from year of 1981-2010 was available.

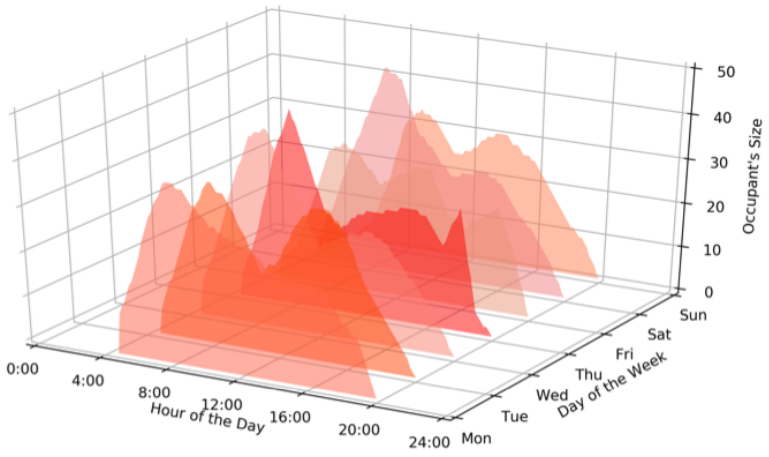


Figure 2: The Average Trends of Occupant Flow from Monday to Sunday

The data for the occupant flow in the commercial space is difficult to obtain and often the data is not accessible to the public. This data must be modeled to get a good estimate of parameters. Thus, the occupant flow was evaluated in combination of the Popular Times Graphs from Google, site visit and random sample generators based on Gaussian distribution. Google Popular Times only specifies 7 foot traffic distribution of the café from Monday to Sunday. From there, we recorded actual occupant data at a Starbucks Café at Albany, CA, for a total of 10-hour interval. Then these numbers were scaled based on the trend to map our average estimation as shown in Fig 2. For the simulation purpose, the occupant flow was

Level	Range of Number of People	Level	Range of Number of People
1	0-2	6	24-29
2	3-7	7	30-34
3	8-13	8	35-39
4	14-18	9	40-45
5	19-23	10	$\geqslant 46$

Table 1: Griding of Occupant Flow

then randomly generated from Gaussian distribution with means determined by our average model and standard deviation of 3.02 calculated from our 10 data points.

2 Simple Control

For the purpose of comparing and contrasting, a simple control model was incorporated to test the effectiveness of our optimization strategy. The optimal temperature range was defined to be from 24^oC (75^oF) to 26^oC (79^oF). The control rule is activated based on the optimal time temperature. When the temperature is below the lower bound, heating mode would be turned on. Alternatively, if the indoor temperature is higher than the upper bound, cooling mode would be turned on. The cost was characterized in the similar fashion as described in the Dynamic Programming section.

3 Dynamic Programming (DP)

3.1 Assumptions

In this project, the stochastic process for determining the trend of occupant flow was assumed to follow the Markov Property. That means future state of occupant level depends solely on present state. For the other random state variables such as outdoor temperature, it was assumed that a reliable prediction of data would be provided and the errors between actual data and predicted result were trivial.

3.2 Markov Chain Model

To begin with, the occupant flow was separated into k levels based on the principle that occupant flow in same level will have similar influence on indoor temperature. In this project, k is set to be 10, and the time interval is set to be 10 minutes (total time steps $N = 144$).

Since the flow of occupant follows Markov property, a series of matrices $P_n, n = 0, 1, \cdots, n-1$ can be formed to describe such stochastic process. P_n represent the transition matrix from time step n to $n+1$, in which p_{ij} is the probability that the system transfer to state j under current state of i.

$$p_{ij} = Pr(X_{n+1} = j | X_n = i)$$

$$P_n = \begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0k} \\ p_{10} & p_{11} & \cdots & p_{1k} \\ \vdots & \vdots & \ddots & \vdots \\ p_{k0} & p_{k1} & \cdots & p_{kk} \end{bmatrix}$$

As discussed before, different weekdays in a week have its own unique flow pattern and there is no big difference between different weeks. With the data of occupant flow in past year, those matrices can be calculated from dividing the frequency of specific transition by the number of weeks. For instance, p_{ij} in P_n on Monday is calculated by

$$Pr(X_{n+1} = j | X_n = i) = x_{ij} = \frac{m_{ij}}{m}$$

where m_{ij} is how many times the system transits from state i at timestep n, to state j at timestep $n+1$. m is the number of weeks. (There are 52 weeks available in this project.)

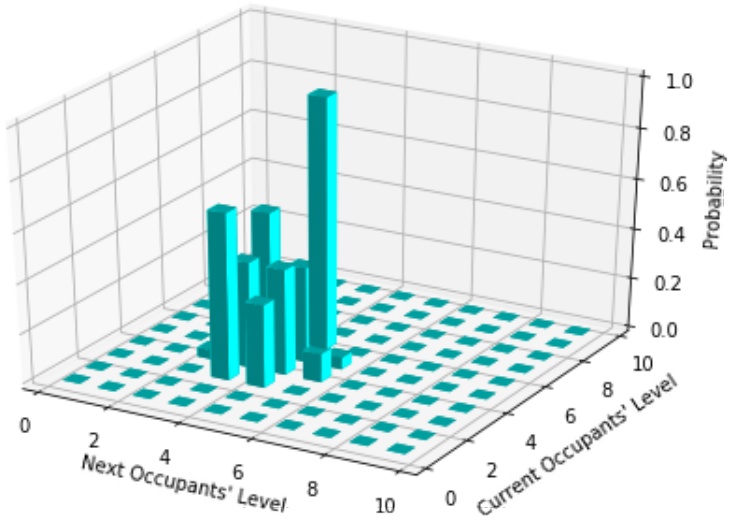


Figure 3: The Matrix of Markov Chain at $n = 35$

3.3 Stochastic Dynamic Programming

Given the Markov Chain model that dealt with the random process, a stochastic dynamic programming can be formulated to solve the optimization problem.

3.3.1 Objective Function

The objective of this optimization problem is to minimize the expected cost, namely the sum of electricity cost and penalty of temperature that went outside of comfort zone:

$$\min_{P_e(n), T(n), N(n)} J = \mathbb{E} \sum_{n=0}^{N-1} c_u(n) \Delta t P_e(k) + c_{comf}(T(n))$$

$N(n)$: Occupant flow level
$T(n)$: Current room temperature, $°C$.
$P_e(k)$: Input power of Air conditioner, kW.
Δt: Time step, hour
$c_u(n)$: Electricity charge at every time interval, $USD/kW \cdot h$
$c_{comf}(\cdot)$: Penalty of discomfort, USD

Subsequently, the penalty function $c_{comf}(\cdot)$ is a piecewise function which can be written as

$$c_{comf}(T) = \begin{cases} (T - T_{up})^2 & T > T_{up} \\ 0 & T_{low} \leqslant T \leqslant T_{up} \\ (T - T_{low})^2 & T < T_{low} \end{cases}$$

3.3.2 Constraints in the Model

1. To begin with, the AC units have maximum output for heating power and cooling power:

   $$-P_{cool}^{max} \leqslant P_{real}(n) \leqslant P_{heat}^{max}$$

   a negative sign is given to cooling power since in the dynamic equation, positive energy numerically results a raising state while negative energy leads to a drop.

2. A hard bound of interior temperature are also imposed on this system, which means even though there is leniency towards going out of comfort zone, ridiculous values (such as 5 $°C$ and 40$°C$) are never allowed. That is

   $$T_{min} \leqslant T(n) \leqslant T_{max}$$

3. There are also some equality constraints. First of all, the evolution of interior temperature satisfies the dynamic equation mentioned before:

   $$T(n+1) = f(T(n), P_{real}(n), N(n))$$

4. The input power can be calculated given output power and corresponding efficiency:

   $$P_e(n) = \begin{cases} \dfrac{P_{real}(n)}{\eta_{cool}} & P_{real} \leqslant 0 \\ \dfrac{P_{real}(n)}{\eta_{heat}} & P_{real} > 0 \end{cases}$$

3.3.3 Value Function and Principle of Optimality

The first step to implement a dynamic programming is to define the value function. Let $V_n(T(n), N_n)$ be the minimum expected cost-to-go from time step n to the end, time step N.

With such value function, the principle of optimality can be written as

$$\begin{aligned} V_n(T(n), N(n)) &= \min_{P_e(n)}\{c_u(n)\Delta t P_e(n) + c_{comf}(T(n)) + \mathbb{E}\left[V_{n+1}(T(n+1), N(n+1))\right]\} \\ &= \min_{P_e(n)}\{c_u(n)\Delta t P_e(n) + c_{comf}(T(n)) \\ &\qquad + \mathbb{E}\left[V_{n+1}(f(T(n), P_{real}(n), N(n)), N(n+1))\right]\} \\ &= \min_{P_e(n)}\{c_u(n)\Delta t P_e(n) + c_{comf}(T(n)) \\ &\qquad + \sum_{j\in S} p_{ij} V_{n+1}(f(T(n), P_{real}(n), N(n)), N(n+1) = j)\} \end{aligned}$$

The boundary condition is

$$V_N(T(N), N(N)) = 0$$

And the optimal control strategy can be acquired by

$$\begin{aligned} P_e^*(n) = \gamma_n(T(n), N(n)) = \arg\min_{P_e(n)}\{&c_u(n)\Delta t P_e(n) + c_{comf}(T(n)) \\ &+ \sum_{j\in S} p_{ij} V_{n+1}(f(T(n), P_{real}(n), N(n)), N(n+1) = j)\} \end{aligned}$$

In consideration of accuracy and computational complexity, the interval of temperature was set to $0.06°C$, and the size of power grid was set to 60, which means the interval was no more than 0.15 kilowatts. For the state variables that have no corresponding value in next time step, interpolation is used to approximate. And increasing the density of grids didn't change V_k so much, implying the reliability of previous griding. Under such setting, the time consumed for optimization was about 5 minutes.

3.4 Selection of AC Units

Besides the control strategy, model of AC units also has a significant influence on total cost. First of all, AC with insufficient output power is not able to keep interior temperature inside the comfort zone, leading to a high penalty. However, extremely powerful AC is not economic as well for the high price, even though it can always maintain a comfortable temperature. In order to cope with such trade-off between expense of AC and penalty of discomfort, this project is designed to make a comparison between four different kinds of AC units, and the table below shows the details.

No.	Cooling Power, kW	Cooling Efficiency	Heating Power, kW	Heating Efficiency	Price, USD
1	2.7	5.51	2.9	4.58	2260
2	3.2	5.33	4.0	4.02	2700
3	4.2	5.46	4.7	3.86	3240
4	5.1	5.09	6.3	3.36	3600

Table 2: Parameters of Different AC Units
(Efficiency $= \frac{P_{out}}{P_{in}}$)

4 Discussion & Results

A dataset with relatively large fluctuation in occupant flow level was selected as the test set. And the results were presented as follows.

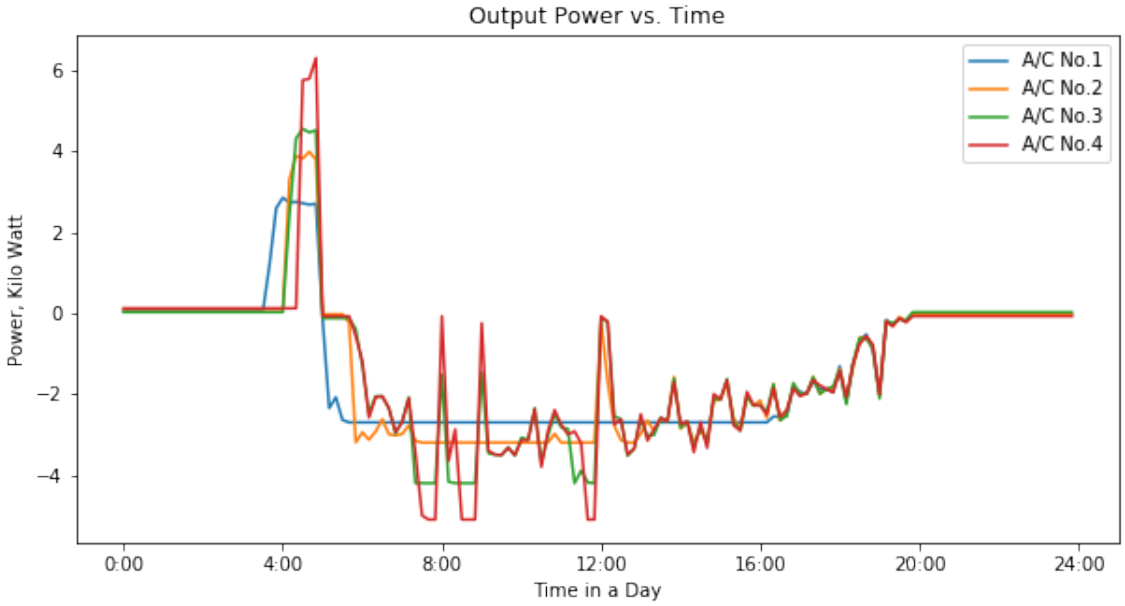


Figure 4: Output Power of AC Units

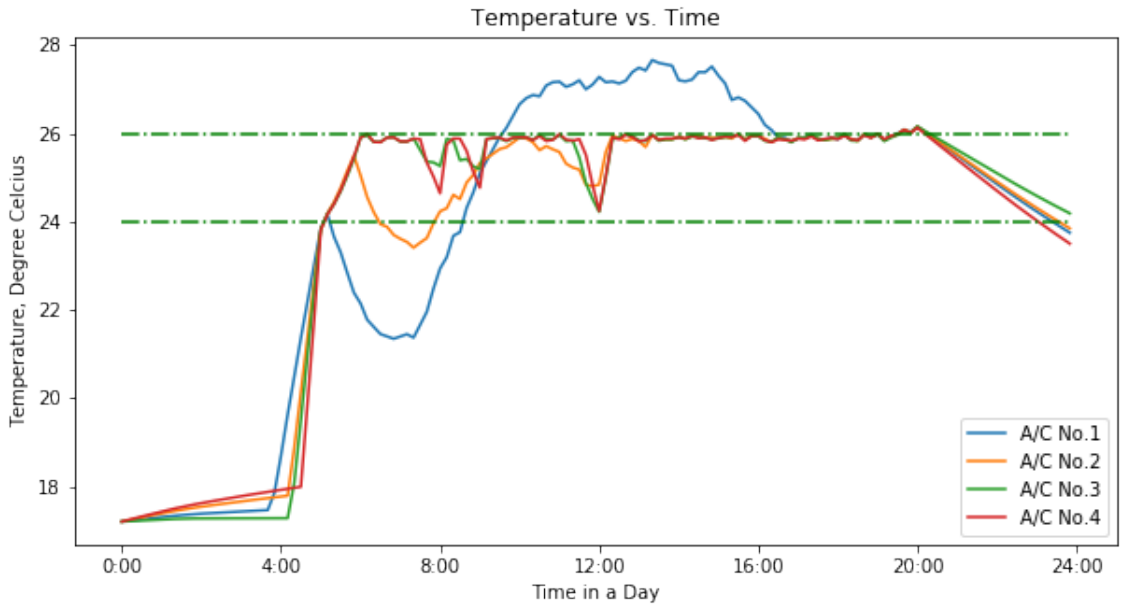


Figure 5: Interior Temperature under the Control of AC Units

No.	Electricity Cost USD/day	Economic Cost USD/day	Penalty USD/day	Total Cost USD/day
1	1.75	7.94	9.98	17.92
2	1.83	9.23	0.10	9.33
3	1.79	10.66	0.00	10.67
4	1.94	11.80	0.00	11.81

Table 3: Cost of Different AC Units
(Economic cost: sum of electricity cost and daily depreciation)

From Fig.4 and Fig. 5, the power of AC 3, 4 were not fully utilized during the process. AC 1 had the lowest daily cost, but it was not capable to handle the performance during the peak-time which led to extremely high penalty (even larger than the economic cost). AC 2 found a balance between two scenarios. On the one hand, it kept temperature inside comfort zone in most of the time. On the other hand, it was more cost effective and there was no waste on surplus capability. Thus, AC 2 had the lowest cost in total, and was the best choice for the setting of this project.

4.1 Comparing to Simple Control

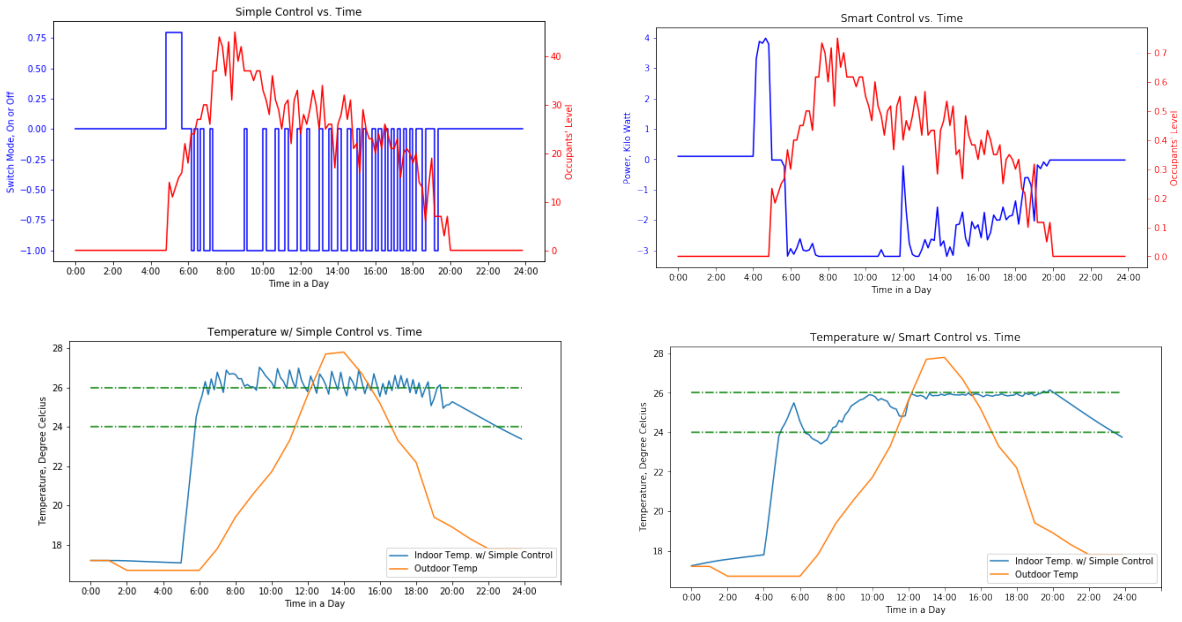


Figure 6: Comparison Between Simple Model and Dynamic Programming

Strategy	Electricity Cost USD/day	Penalty USD/Day	Total Cost USD/Day
Simple Control	3.80	9.21	13.01
Dynamic Programming	1.83	0.10	9.33

Table 4: Cost of Different Control Strategy

Under simple control, the delay of reaction to state changes caused unnecessary energy consumption, making it hard to maintain comfortable temperature. Compared with simple control, SDP control strategy significantly reduced electricity cost and discomfort. Armed with statistic of random process, SDP would take future variance into consideration. Therefore, SDP strategy made the AC warm up the room before opening time and adjusted the energy consumption according to fluctuant electricity rate.

4.2 Expected Cost vs. Real Cost

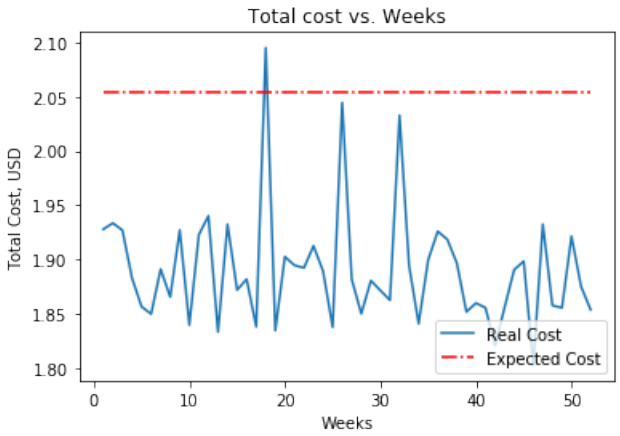


Figure 7: Expected Cost vs. Real Cost

Since the objective function in stochastic dynamic programming is expected cost, the real cost may have non-trivial deviation from it. Fig.7 shows the simulation result of 52 weeks based on SDP strategy. The strategy was found to be reliable because real cost is smaller than expected cost and the variance is reasonable.

4.3 Future Work

For the future consideration, the influence of solar radiation can be added in energy state equation. It plays a significant role in prediction with current data from NOAA's record about cloud percentage. Secondly, more optimization techniques can be implemented. For instance, the switching cost of shifting the AC power should be added to cost function. The incentives were to allow more of smooth temperature transition and to keep AC system for a longer service life. Overall, the model established was solid and it setups a framework for more optimization of HVAC system in the future.

Summary

The aim of the project was to design an autonomous control system for AC units to account for different levels of occupant flow in daily life. Therefore, dynamic programming algorithm coupled with Markov Chain model was employed in this project to determine the optimal strategy across the time frame. In the end, two types of control were compared and contrasted. One with simple strategy in which the system would be turned on once the temperature was beyond comfort range. The other was the optimal strategy calculated from SDP where we implemented cost function consisting of electric price, cost of AC units and penalty for temperature beyond comfort range. As of the results, the optimal solution from SDP has achieved what were stated in the cost function. Most of the temperature stayed in the optimal range and it saved 51.8% energy consumption compared to the simple control strategy.

References

Godo, S., Haase, J., & Nishi, H.(2015). Air Conditioning Control Using Self-Powered Sensor Considering Comfort Level and Occupant Location. *IECON 2015-41st Annual Conference of the IEEE.*

Wang, F., Feng, Q., Chen, Z., Zhao, Q., Cheng, Z., Zou, J.,...Reeve, H.(2017). Predictive control of indoor environment using occupant number detected by video data and CO2 concentration. *Energy & Buildings, 145*, 155-162.

Zhao, Z. & Yu, N.(2017). The application of advanced control technologies in air conditioning system – a review. *Advances in Building Energy Research, 11:1*, 52-66.

Appendix

Code of Markov Chain

```
## Identify Markov Chain
# Discretize irradiance into levels (you can pick more than 10 if you want)
level = 10
N_grid = np.linspace(0.0, np.max(N_10min), level)
TT = np.arange(N_data.shape[0])

t = np.remainder(TT/6, 24)

# Preallocate arrays to count transitions, and probability transition
# matrix
counts = np.zeros([level,level,24*6])
P = np.zeros([level,level,24*6])
Levels = np.arange(level)

# Given value of xi, return the value closest to it from x, y
def nearest_interp(xi, x, y):
```

```
    idx = np.abs(np.subtract(xi, x))
    return y[idx.argmin()]

for idx in range(N_data.shape[0]-1):

    ii = nearest_interp(N_data[idx], N_grid, Levels)
    jj = nearest_interp(N_data[idx+1], N_grid, Levels)
    kk = int(round(t[idx]*6))
    # Increment times irradiance level goes from ii to jj in time step kk
    counts[ii,jj,kk] += 1

for ii in range(level):
    for kk in range(24*6):
        # Compute fraction of times irradiance level goes from ii to jj in
   time step kk
        # out of ALL transitions out of level ii
        if np.sum(counts[ii,:,kk]) != 0:
            P[ii,:,kk] = counts[ii,:,kk] / np.sum(counts[ii,:,kk])
```

Code of Stochastic Dynamic Programming

```
# Stochastic Dynamic Programming

# Simple clock
start = time.time()
# Coefficient of discomfort penalty
alpha = 0.08
# Number of A/C unit candidates
n_ac = P_heat.size
# Number of occupant flow level
People_level = 10
# Time step in a day, interval is 10 minutes
time_range = 24 * 6
# Size of grid of temperature
grid_size_t = 170
# Size of grid of Power
grid_size_p = 60
# Temperature grid
temp_grid = np.linspace(min(t_min), max(t_max), grid_size_t)

# Matrix that stores value function
v = np.inf * np.ones((time_range + 1, grid_size_t, People_level, n_ac))

# Matrix that stores optimizor
u_star = np.zeros((time_range + 1, grid_size_t, People_level, n_ac))

# Initialize value function
v[time_range, :, :, :] = 0

save_err = np.seterr(all='call')

# Evaluate current temperature, gives a penalty if necessary
def comfort_cost_func(Temp):
    if Temp <= 26 and Temp >= 24:
```

```
        return 0.0
    elif Temp > 26:
        return alpha*(Temp -26)**2
    elif Temp < 24:
        return alpha*(Temp -24)**2

# Convert output power to input power
def pr2pe(pr, n_ac):
    if(pr >=0):
        return pr / ef_heat[n_ac]
    else:
        return - pr / ef_cool[n_ac]

# Begin SDP for 4 different A/C units
for n in range(n_ac):
    for i in range(time_range -1, -1, -1):
        for k in range(0, grid_size_t):
            for ii in range(0, People_level):
                # Calculate feasible field
                lb = max(P_cool[n], - T_10min[0][i] / R + (- C  + 1 / R) *
   temp_grid[k] + C * t_min[i] - p_I * N_grid[ii])
                ub = min(P_heat[n], - T_10min[0][i] / R + (- C + 1 / R) *
   temp_grid[k] + C * t_max[i] - p_I * N_grid[ii])

                if(lb >= ub):
                    v[i, k, ii, n] = np.inf
                    continue

                p_grid = np.linspace(lb, ub, grid_size_p)

                T_next = ((T_10min[0][i] - temp_grid[k]) / R + p_grid + p_I *
   N_grid[ii]) / C + temp_grid[k]


            # Calculate expected cost-to-go of next step
                v_next = np.zeros([People_level, grid_size_p])
                Ev_next = np.zeros(grid_size_p)

                for j in range(0, People_level):
                    v_next[j, :] = np.interp(T_next, temp_grid, v[i + 1, :, j,
    n])
                    for jj in range(0, grid_size_p):
                        if(np.isnan(v_next[j, jj]) or v_next[j, jj] == np.inf)
   :
                            v_next[j, jj] = 1e10

                    Ev_next = P[ii, j, i] * v_next[j, :] + Ev_next

                pe = np.zeros(grid_size_p)
                for j in range(grid_size_p):
                    pe[j] = pr2pe(p_grid[j], n)

                if(i < 5 * 6 or i >= 20 * 6):
                    v[i, k, ii, n] = min(c_u[i] * pe + Ev_next)
```

```
                    index = np.argmin(c_u[i] * pe + Ev_next)
                else:
                    v[i, k, ii, n] = min(c_u[i] * pe + comfort_cost_func(
    temp_grid[k]) + Ev_next)
                    index = np.argmin(c_u[i] * pe + comfort_cost_func(
    temp_grid[k]) + Ev_next)

                u_star[i, k, ii, n] = p_grid[index]

    end = time.time()
    print(end - start)
```

Acknowledgments

In this semester, all of us enjoyed studying CE295 a lot. We learned a variety of tools for control and optimization. First of all, we would like to thank our instructor, Prof. Scott Moura, not only for the interesting and inspiring materials, but also his rigorous attitude during the course. Prof. Moura was so passionate and enthusiastic that we were immersed in lectures and developed a deep understanding of all the materials. Secondly, we want to express our thanks to our GSI Bertrand Travacca for all the extra workshops which supplemented the course with some essential knowledge.

About the authors

All authors of this project are Master of Engineering student from UC Berkeley. Lorna Chen, Yao Han, Sikang Hu are from Civil and Environmental department, majored in Systems Engineering. Xinhao Wang is from Mechanical Engineering department majored in Energy.

Part IV

Batteries and Energy Storage

Minimization of cumulative aging in batteries: a grid-based approach

Yu-Hsin (Bryant) Huang, Yaser Marafee, Raja Selvakumar

Abstract

In this study we uniquely focus on the translation of portable size battery kinetic information to grid level analysis for a time-variant system. We characterize the effective charging policies based on fluctuating demand and grid supply. We have constructed a dynamic programming framework to understand the effects of drawing power from the grid when compared against the cumulative aging of the battery. We present a formulation that includes two time-variant states and one control variable. We observe that with increasing the tuning parameter α, we delay the charging/discharging of batteries and therefore observe the impact of cycle life on the battery. A capacity fade of less than 1% is observed over a 24h time period, and fade of 7% is observed over a full week. Future work would be keyed on simplifying some of the coupled and nonlinear constraints to additionally include temperature dynamics and multiple battery nodes.

Introduction

Motivation and Background

Electrification of renewable energy integration and automobile transportation is essential towards the reduction of greenhouse gas emission and therefore the impact of global warming [1] The importance of energy storage devices can be seen in consumer electronics, electric vehicles, and grid storage. With increasing diversification in renewable and intermittent power supplies, larger integration of energy storage systems is imminent.

The main issue with batteries is aging during their lifetime due to the decrease in capacity, which leads to voltage decay and loss of power. There are numerous electrochemical mechanisms to describe aging [2, 3, 4]. However, characterization is challenging due to diverse time scales and complex nonlinearities in these models. Dynamic parameter estimation and battery state-of-charge (SOC) analysis remain complex topics.

As Figure 1 demonstrates, some of the studied impacts on battery aging is the cycle time, depth-of-discharge (DOD), and storage life. It is observed that amongst these parameters, it is the cycle lifetime that affects the lithium evolution the strongest. A more involved discussion is presented in the Relevant Literature section; the purpose of this early statement is to illustrate the direction of our research interests.

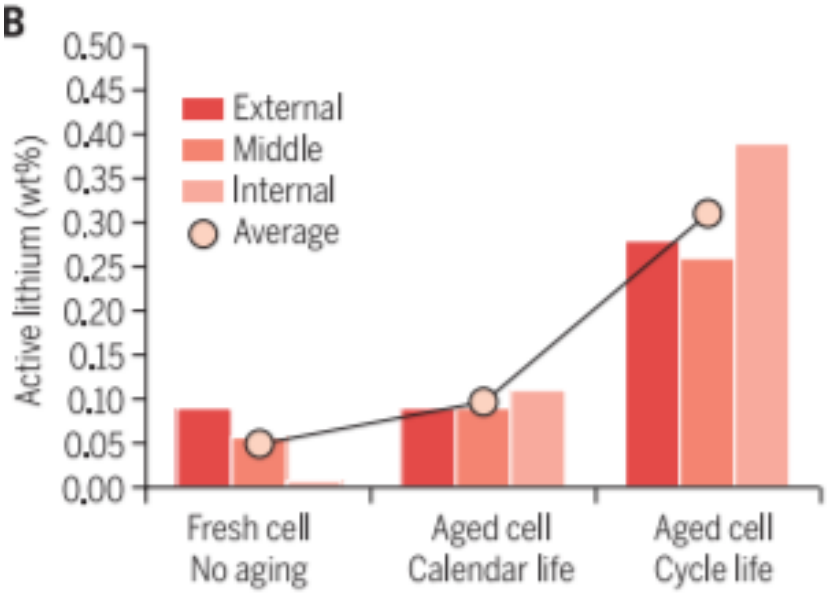


Figure 1. *Evolution of active lithium upon aging of a Li-ion cell. Note that lithium evolution at the anode increases more prominently with battery cycling.* [1].

To that end, this project focuses on development of an optimization model which combines the two challenging aspects of battery modelling: the grid storage and cell aging mechanisms. With regards to the grid storage framework, we remodel the conventional grid network as a source-sink problem. The input $G(k)$ is defined as the grid power generated at time k, which represents the variable power derived from combining nonrenewable and renewable energy sources. Subsequently $d(k)$ represents the time-variant demand response that depends on the change in hourly demand from residential, commercial, and industrial buildings. This transformation is shown in Figure 2. Note that this formulation is robust for multiple batteries, although computational time scales exponentially with the addition of each state. This progression is shown in the Appendix.

The second element of our project involves using a dynamic programming (DP) framework to minimize cumulative aging of the batter(ies) presented in the system. We try to investigate the capacity fade from parameters such as time, temperature, depth of discharge (DOD), and discharge rate as presented in Equation 1:

$$Q_{loss} = B \exp(\frac{-E_a}{RT})(A_h)^z \tag{1}$$

where Q_{loss} is the percentage of capacity loss, B is the pre-exponential factor, E_a is the activation energy, R is the gas constant, T is the absolute temperature, and A_h is the amp-hour throughput, which is expressed as $A_h = (\text{cycle number}) \cdot (\text{DOD}) \cdot (\text{full cell capacity})$, and z is the power law factor. The work done by Wang et. al [5] presents a relationship for the capacity fade from temperature and C-rate effects. The estimated values for a LiFePO4 battery are provided in the Technical Description section.

Our group consists of three chemical engineering masters students in the Product Development Program. We have all taken classes in electrochemical systems, ranging from

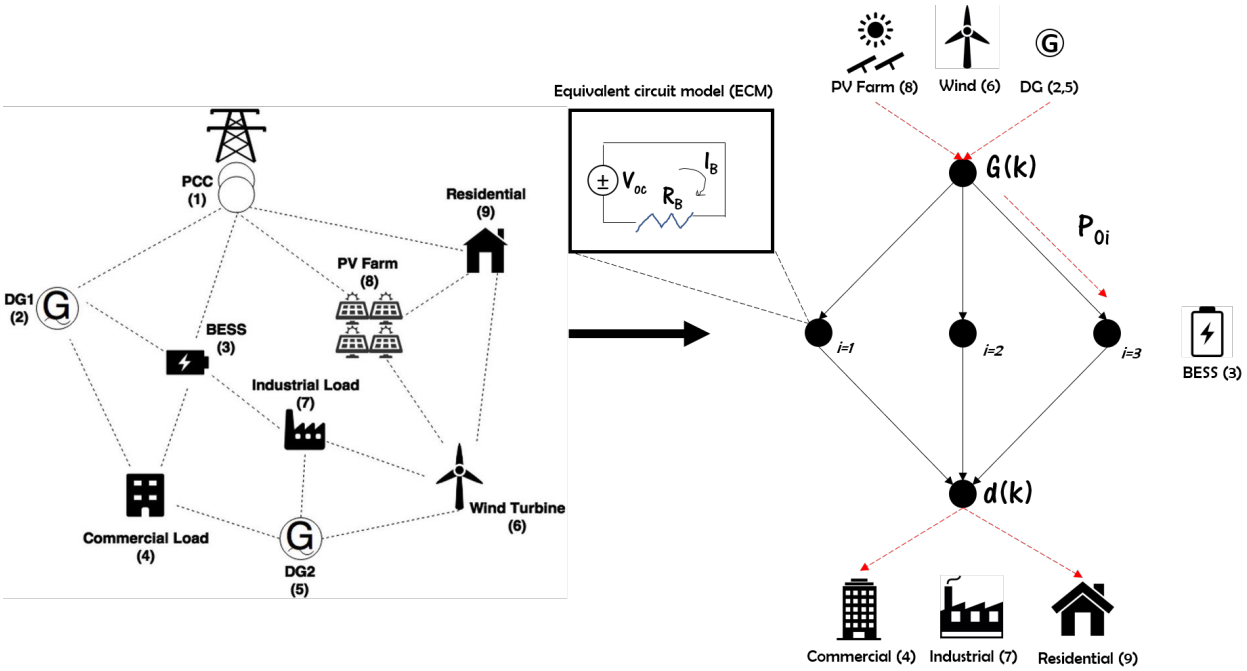


Figure 2. *Grid transformation of a typical microgrid network into a max flow problem. The inherent assumption is that we use prior knowledge on the evolution of $G(k)$ and $d(k)$ as supply and demand. The remaining surplus or shortage will be taken from the batteries at each node i. Figure adapted from* [8].

mathematical fundamentals to first steps of solid-electrolyte interphase (SEI) modelling. These experiences provide us an advantage to test the differences between black-box and white-box modelling approaches. As a team, our goal is to learn advanced control and parameter estimation techniques, so we hope to demonstrate this in our project results.

.

Relevant Literature

Battery aging can be classified into two main categories: calendar aging and cycle aging [6]. The former is associated with the phenomena and the consequences of battery storage and cycle aging corresponds to the influence of battery utilization time. Calendar aging is the irreversible process of lost capacity during storage. The battery is degraded due to self storage. Temperature effects greatly contribute to calendar aging as side reactions are facilitated and cause capacity fade, as shown in Figure 3. The other principal variable under investigation in calendar aging is the State of Charge (SOC). The state of charge is equivalent to the ratio of the current battery energy to the maximum capacity. While operating at an increased SOC may result in greater energy throughput, this also results in larger capacity fade as indicated earlier in Equation 1. These mechanisms are described by electrochemical models, performance based models, and equivalent circuit based (ECM) models.

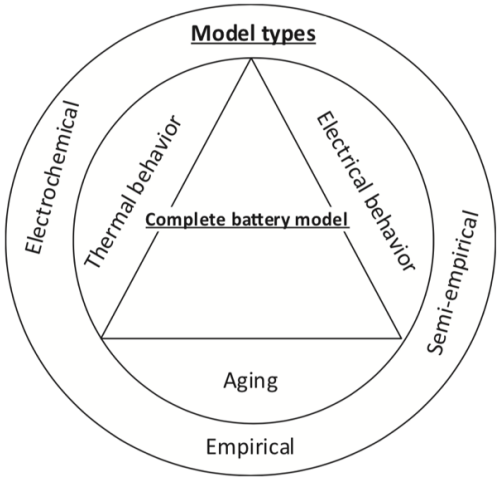


Figure 3. *Different battery modelling schemes.* [7]

Focus of this Study

For the purposes of this study, we aim to consider the effects of temperature and cycle aging on the battery SOC. As existing grid optimization models operate on the premise of minimizing cost of electricity usage [10] or fuel usage [11] or some affine combination of the two. We present here the first consideration of evaluating costs for a aging of larger scale grid storage battery systems by using microkinetic analysis of effects of cycle time and temperature on cumulative aging.

Key contributions

In this study we uniquely focus on the translation of portable size battery kinetic information to grid level analysis for a time-variant system. We characterize the effective charging policies based on fluctuating demand and grid supply.

Technical description

Assumptions

We first itemize the assumptions we have made for this model:

- Power can be drawn from the main power source, and we assume it has a maximum capacity G_{max} over all time N
- Power flow from the source is unidirectional (i.e. we only consider charging)
- The node represents a power distribution center, where the balance from power demanded and power transmitted is the power storage
- Battery temperature is constant with ambient conditions

- Power throughput is controlled at a constant C-rate[1]
- Grid batteries used are Li-ion
- Final state of charge is 0.3 (at the end of the simulation time step)
- Only one battery node is considered for the scope of the results of this project, while the formulation is written for many nodes.

Formulation

We now define our grid and the appropriate parameters. Our operating framework was first presented in Figure 2. Table 1 includes all the definitions for the variables and parameters used in this formulation.

The objective function is to minimize the cumulative aging across all nodes $i \in [1, I]$, as shown in Equation 2. Note that I represents the total nodes in the grid space. As aforementioned, however, implementing a model for multiple nodes grows rapidly in complexity without use of techniques like Approximate Dynamic Programming (ADP) and decoupling the coupled affine constraint as given by Equation 7.

$$\min \sum_{k \in N} \sum_{i \in I} \left(\alpha \ln Q_i(k) + c(k) P_{0i}(k) \right) \tag{2}$$

$$\ln Q_i(k) = \ln(B) - \frac{E_A}{R T_i(k)} + z \ln \frac{W_{h,i}(k)}{V_i(k)} \quad \forall k \in 1..N, i \in 1..I \tag{3}$$

$$W_{h,i}(k+1) = W_{h,i}(k) + |p_{b,i}(k)| \Delta t \quad \forall k \in 1..N, i \in 1..I \tag{4}$$

$$E_i(k+1) = E_i(k) - p_{b,i}(k) \Delta t \quad \forall k \in 1..N, \forall i \in 1..I \tag{5}$$

$$P_{0i}(k) = d(k) - p_{b,i}(k) \quad \forall k \in 1..N, \forall i \in 1..I \tag{6}$$

$$\sum_{i \in I} P_{0i}(k) \leq G(k) \quad \forall k \in 1..N, \forall i \in 1..I \tag{7}$$

$$E_{\min} \leq E_i(k) \leq E_{\max} \quad \forall k \in 1..N, i \in 1..I \tag{8}$$

$$0 \leq W_{h,i}(k) \leq W_{h,\max} \quad \forall k \in 1..N, i \in 1..I \tag{9}$$

$$-P_{\text{batt, max}} \leq p_{b,i}(k) \leq P_{\text{batt, max}} \quad \forall k \in 1..N, i \in 1..I \tag{10}$$

Equation 2 represents the cost minimization equation. α here is the key tuning parameter that has thus far not been determined in literature surveys. In particular, if one tunes the $\frac{\alpha}{c(k)}$ ratio, this will present a minimization frontier that exists for a range of time evolving values. While a Pareto Frontier wasn't calculated for this project, this is an additional step that we recommend.

[1]This is based on the relationship between power and energy as: $\frac{P}{E}$ = C-rate. Based on the work by Wang et. al, we follow a C-rate of C/2

Equation 3 is the linearized logarithmic expression for the percent capacity loss. Note mainly here that we have transformed the current throughput $A_{h,i}$ to the power throughput $W_{h,i}$. This is primarily done to describe the battery charging dynamics that are presented in Equation **??**. We describe the evolution of the battery dynamics in Equations 4-6. In order to maintain some understanding of the energy state E_i, this transforms into a state variable. Equation 7 now poses the main challenge for our model as with increasing numbers of batteries, the optimization function will represent coupled dynamics. For future work, the authors recommend starting with simple test cases (i.e. 2 nodes) with smaller grid sizes to account for two-state dynamics repeated across two nodes. Equations 8-10 are simply the state constraints on the variables.

$$\rho C V_B \dot{T}_i(k) = hA_s\left(T_i(k) - T_\infty\right) + R_B\left(\frac{p_{b,i}(k)}{V_i(k)}\right)^2 \quad \forall k \in 1..N, i \in 1..I \tag{11}$$

$$T_i(k) + \dot{T}_i(k)\Delta t = T_i(k+1) \quad \forall k \in 1..N, i \in 1..I \tag{12}$$

$$T_{\min} \leq T_i(k) \leq T_{\max} \quad \forall k \in 1..N, i \in 1..I \tag{13}$$

$$V_i(k) = V_{oc,i}(k) - I_i(k)R_B \quad \forall k \in 1..N, i \in 1..I \tag{14}$$

$$Q_{\text{cap, i}}(k) = Q_{\max}\left(1 - Q_i(k)\right) \quad \forall k \in 1..N, i \in 1..I \tag{15}$$

$$0 \leq I_i(k) \leq I_{\max} \quad \forall k \in 1..N, i \in 1..I \tag{16}$$

The above equations represent the temmperature dynamics of the system, which is where the major complications arise in the battery state dynamics. We present the formulation but recognize that due to computational limitations, we are unable to pursue the effects of temperature evolution on battery aging. A further study could linearize and otherwise relax some of the provided constraints.

We now discuss the dynamic programming (DP) framework for this project. Let $V(k)$ represent the cumulative capacity fade and power generation from time step k to total time N. We define control variables $p_{b,i}(k)$ as u_k $\forall i$ and state variables $W_{h,i}(k), E_i(k)$ as x_k $\forall i$:

$$V_k(x_k) = \min_{u_k, x_k}\left\{\sum_{i\in I}\left(\alpha \cdot Q_i(k) + c(k)\cdot\left(d_i(k) - u_i(k)\right)\right) + V(k+1)\right\} \quad \forall k \in 1..N \tag{17}$$

We finally establish the boundary condition:

$$V(N+1) = 0$$

Table 1 presents a cumulative list of all the parameters and values used in the formulation.

Discussion

We now present the main conclusions from this project. Figure 4 represents the battery state evolution over a full day with only considering fluctuation in grid energy price, and not considering the battery aging cost. This policy minimizes the use of the battery in general

Table 1. List of parameters and variables used in formulation

Symbol	Description	Units	Value
α	Fixed cost of capacity loss	[$]	50
$c(k)$	Grid energy cost at time k	[$/kW]	0.02[a]
$Q_i(k)$	% capacity loss of node i at k	[-]	
B	Pre-exponential factor	[-]	30,330
E_A	Activation energy	[J/mol]	-31,500
R	Gas law constant	[J/mol*K]	8.314
T_∞	Ambient temperature	[K]	298
z	Power law factor	[-]	0.552
$V_i(k)$	Voltage at node i at time k	[J]	
$W_{h,i}(k)$	Power throughput	[J]	
$E_i(k)$	Energy state	[J]	
$p_{b,i}(k)$	Power surplus/shortage	[W]	
$P_{0i}(k)$	Power delivered from grid to i	[W]	
$G(k)$	Grid variable power supply	[W]	50E3[b]
$d(k)$	Variable demand	[W]	25E3
Δt	Simulation time step	[h]	1
N	Total simulation time	[h]	24
E_{min},E_{max}	Min and max energy limits[c]	[W-h]	792, 79200[d]
$W_{h,min}$,$W_{h,max}$	Min and max power throughput limits	[W-h]	0, 24·E_{max}
$P_{batt,max}$	Maximum battery charge/discharge limit	[W]	0.9·E_{max}/dt

[a]Time variant cost calculated from hourly electricity given in CE295 HW3, Spring 2018
[b]Value estimated based on existing grid capacities, this is the max value
[c]This is derived from the C-rate.
[d]Determined using the A-h throughput from Wang et. al [5].

and just meets the demand as required. This is different from the results shown in Figure 5, where as α increases, there is a noticable delay in charging/discharging from the battery. This indicates that the DP recognizes the aging equation accounts for the time evolution of the battery state and tries to draw more power from the grid (as long as its within the the operating limits of the battery). This is a remarkable finding that shows the power of the minimization function. Finally, Figure 6 proves this hypothesis by showing for a week-long optimization, in the first day the demand does not draw power from the battery at all! So cumulative aging is decreased because overall usage of the battery decreases itself. We'd also like to note here that the calculated values of 7% capacity fade over a week long period matches what we expect from literature values [5].

We would now like to enumerate the key takeaways from this project and further discussion items:

- The equation for capacity loss is based on a 2.2Ah battery which lead to an unreasonable capacity loss when the battery that is being minimized has a much larger capacity.

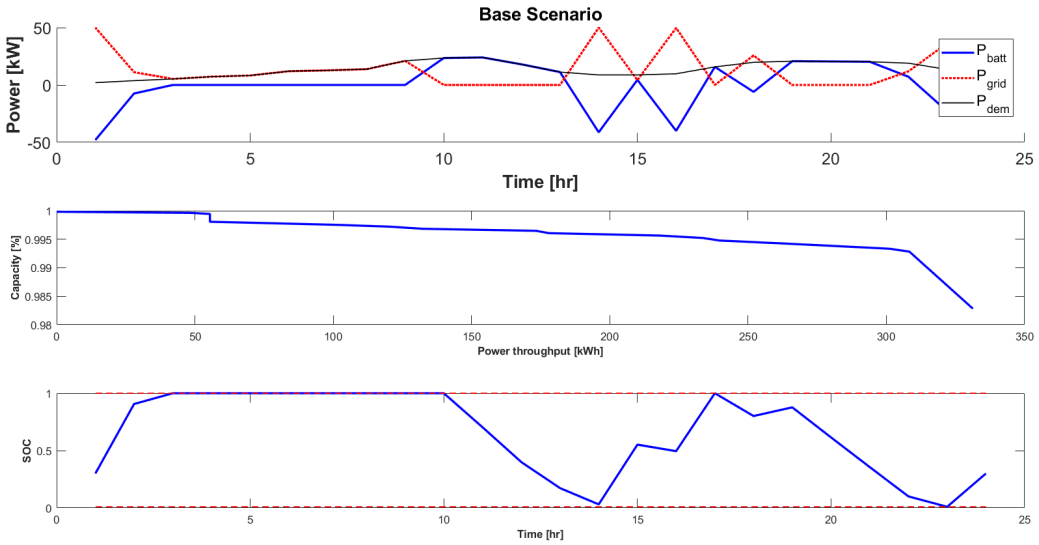


Figure 4. *Grid based optimization using* $\alpha = 0$

Thus, we had to tune the equation to give reasonable capacity losses that would yield similar quantitative results to the plots provided in literature.

- In order to incorporate more batteries in the system, we would need to account for charge/discharge power for each battery in the constraints. We faced a great challenge in trying to implement this in dynamic programming. Perhaps using a ADP framework and relaxing some constraints could lead to convex optimization.

- We were able to corporate temperature as a third state, however, due to high computational requirements, the grid size had to be reduced which resulted in inaccurate interpolations because the grid size is small. Since the grid size ranges of the energy state, power throughput, and temperature are all different, this yields different grid sizes and therefore parameter sensitivity would be quite high.

- One would consider the practical values used in the simulation. Of course we have estimated parameter values based on previous homeworks and some literature survey, but this system should be robust to handle more realistic parameters. If not, the authors advise exploring a modified capacity loss equation that may include some post-exponential "correction factor" that accounts for this.

Executive Summary

In this study we uniquely focus on the translation of portable size battery kinetic information to grid level analysis for a time-variant system. We characterize the effective charging policies based on fluctuating demand and grid supply. We have constructed a dynamic programming framework to understand the effects of drawing power from the grid

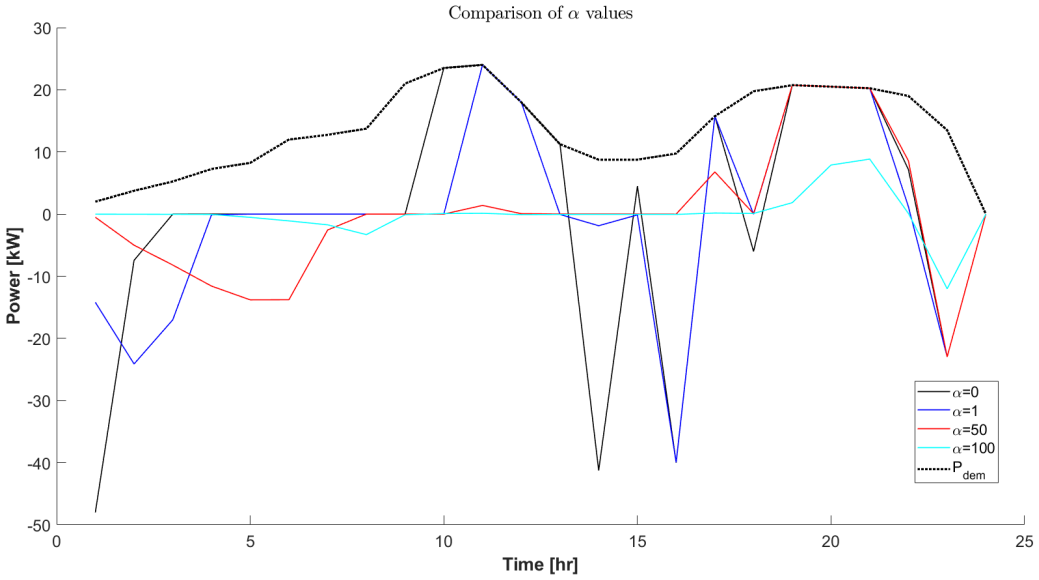


Figure 5. *Grid based optimization using different values of* α

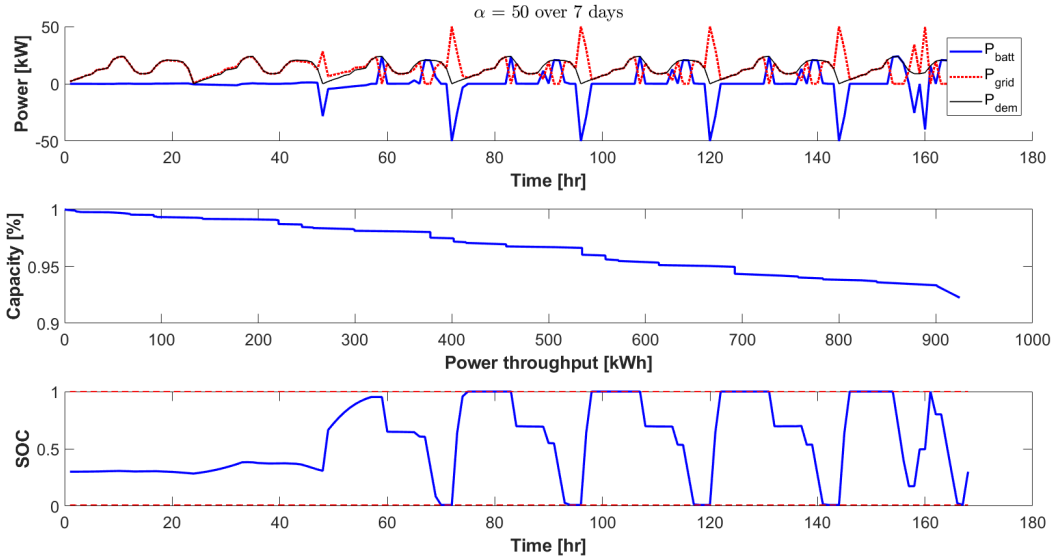


Figure 6. *Week long optimization based on* $\alpha = 50$

when compared against the cumulative aging of the battery. We present a formulation that includes two time-variant states and one control variable. Our results indicate control policies that fluctuate with the tuning parameter α, which represents the cost of losing power to capacity fade. We observe that with increasing this tuning parameter, we delay the charging/discharging of batteries and therefore observe the impact of cycle life on the battery. Future work would be keyed on simplifying some of the coupled and nonlinear constraints to additionally include temperature dynamics and multiple battery nodes.

References

[1] Palacin, M. R.; de Guibert, A., "Why do batteries fail?" Science 2016, 351 (6273), 7.

[2] Baek, K.W., Hong, E.S. & Cha, S.W. Int. J Automot. Technol. (2015) 16: 309. https://doi.org/10.1007/s12239-015-0033-2

[3] Pinson, M. B., Bazant, M. Z (2013). "Theory of SEI Formation in Rechargeable Batteries: Capacity Fade, Accelerated Aging and Lifetime Prediction." Journal of The Electrochemical Society. Volume 160. Pages A243-A250.

[4] Vetter, J.; Novak, P.; Wagner, M. R.; Veit, C.; Moller, K. C.; Besenhard, J. O.; Winter, M.; Wohlfahrt-Mehrens, M.; Vogler, C.; Hammouche, A., "Ageing mechanisms in lithium-ion batteries." J. Power Sources 2005, 147 (1-2), 269-281.

[5] Wang, J.; Liu, P.; Hicks-Garner, J.; Sherman, E.; Soukiazian, S.; Verbrugge, M.; Tataria, H.; Musser, J.; Finamore, P., "Cycle-life model for graphite-LiFePO4 cells." J. Power Sources 2011, 196 (8), 3942-3948.

[6] Barre, A.; Deguilhem, B.; Grolleau, S.; Gerard, M.; Suard, F.; Riu, D., "A review on lithium-ion battery ageing mechanisms and estimations for automotive applications." J. Power Sources 2013, 241, 680-689.

[7] Jaguemont, J.; Boulon, L.; Dube, Y., "A comprehensive review of lithium-ion batteries used in hybrid and electric vehicles at cold temperatures." Applied Energy 2016, 164, 99-114.

[8] Plett, G. L., "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs." Journal of Power Sources 2004, 134, 262-276

[9] de Azevedo, Ricardo Cintuglu, Mehmet Ma, Tan Mohammed, Osama. (2016). Multiagent-Based Optimal Microgrid Control Using Fully Distributed Diffusion Strategy. IEEE Transactions on Smart Grid. 1-1. 10.1109/TSG.2016.2587741.

[10] G. C. Lazaroiu, V. Dumbrava, G. Balaban, M. Longo and D. Zaninelli, "Stochastic optimization of microgrids with renewable and storage energy systems," 2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC), Florence, 2016, pp. 1-5. 10.1109/EEEIC.2016.7555486

[11] F. A. Mohamed and H. N. Koivo, "Online Management of MicroGrid with Battery Storage Using Multiobjective Optimization," 2007 International Conference on Power Engineering, Energy and Electrical Drives, Setubal, Portugal, 2007, pp. 231-236. 10.1109/POWERENG.2007.4380118

Appendix

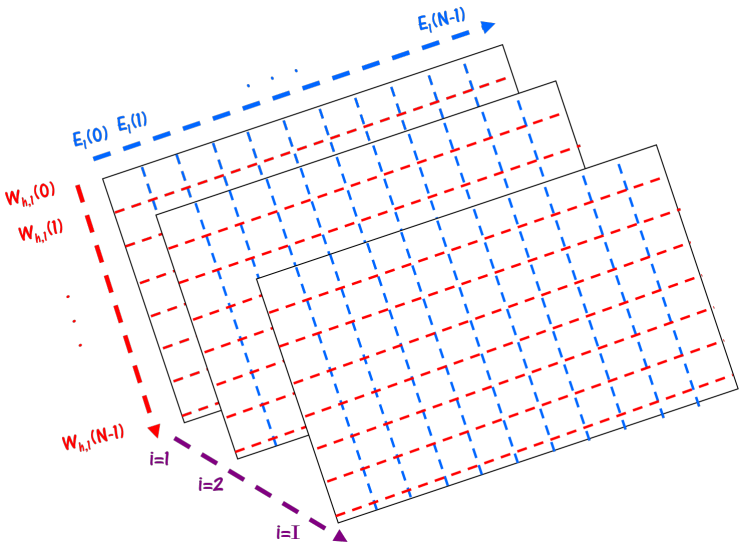


Figure A 1. *Computational scaling of increasing number of states and number of nodes processed. Increasing the number of states increases the complexity exponentially.*

Acknowledgments

We'd like to thank Professor Scott Moura for his extensive feedback and help in developing the formulation for this project. Most helpful was breaking the problem down into solvable portions and determining what is feasible and what is practical in emulating battery state dynamics.

We'd also like to acknowledge Bertrand Travecca for providing astute observations on simplifying some constraint of the problem and how to further develop the project analysis.

About the authors

Yu-Hsin (Bryant) Huang is a MS candidate in Chemical Engineering under the Product Development Program (PDP) at UC Berkeley. His career interests include PHEV and battery management.

Yaser Marafee is a MS candidate in Chemical Engineering under the Product Development Program (PDP) at UC Berkeley. His career interests include process optimization and sustainable fuels.

Raja Selvakumar is a MS candidate in Chemical Engineering under the Product Development Program (PDP) at UC Berkeley. His career interests lie in the intersection of machine learning and chemical process design.

States Estimation of Li-ion Battery

Junzhe Shi, Franklin Zhao, Ruitong Zhu, and Xin Peng

Abstract

Batteries are ubiquitous in all forms of electronics and transportation, and a key to the store of clean and secure energy. For different kinds of batteries, Li-ion battery is the most prominent one for their superior gravimetric and volumetric energy density. For the safe operation of Li-ion battery, the state of charge (SOC) and state of health (SOH) estimation are of great significance. Hence, the goal of the project is to design a robust observer which can estimate the SOC and SOH of Li-ion batteries. In the project, the equivalent-circuit model is used for the battery modeling with current and ambient temperature as inputs and voltage as the measured output. The equivalent-circuit model includes three parts which are an electrical model, a thermal model, and an aging model. To ensure the accuracy of states estimation, the Extend Kalman Filter (EKF) is applied and examined in the project. The battery system is constructed and simulated using MATLAB. The best observer built in this project is a Voltage-Temperature (VT) observer which can accurately observe SOC with great robustness, while SOH can be observed using open-loop observer. The robustness of designed observer is tested using the wrong initial estimates and wrong model parameters.

Introduction

Motivation and Background

The identification of battery operation and aging in real life has been a long-desired yet challenging goal, which includes multiple complex processes in complicated operating conditions and environments. An accurate method to observe SOC and SOH of Li-ion battery is in need. Meanwhile, batteries invariably work at varying thermal and aging conditions. Thus, it is necessary for us to build a battery observation system to monitor operation and aging of battery. The potential challenges also exist. We need to express a multi-control problem via mathematical equations and combine electrical model, thermal model and aging model. Besides, since all team members are major in Civil Systems, a lack of background knowledge in electrical engineering can be a big challenge. However, our previous course CE 291F, Control and Optimization of Distributed Parameters Systems can be helpful for the project. It gave us a background knowledge of partial differential equations, conservation laws, linear stability, Kalman filter and so on. We all have experiences of building, controlling and optimizing systems, including quench process, heat diffusion and Lighthill Whitham Richards model.

Focus of this Study

In this project, we will focus on the SOC and SOH of Li-ion batteries. Based on equivalent-circuit, the electrical, thermal and aging models will be developed for the observing system. Since battery monitoring and management can be the key to allowing innovation in future designs because of their limit properties, our system may play an important role in such an area, and significantly contribute to the energy saving and efficiency.

Literature review

The behavior of batteries is difficult to predict because of its non-linearity, and there has been quite a few attempts to model and estimate the inner state of the system. To better describe the behavior of batteries, a lumped-parameter electro-thermal model was introduced to capture the correlation between the thermal and electric behavior [1], a semi-empirical cycle-life model was established to investigate the attributes of capacity loss [2]. As a combination of the two models above, a coupled electro-thermal-aging model is developed in [3], which captured the systematic dynamics for lithium-iron-phosphate batteries. This model provides a decent assumption for the battery behaviors, which we will go through in detail at the section of mathematical model, as well as an open-loop observer for the Sate of Charge (SOC) and State of Health (SOH). Inspired by the course material of CE 295, we tried to go further to investigate more possible approaches to estimate the battery states.

Key contributions

Based on the thermal-electrical-aging model as discussed above, we build a robust observer for the model, which provides users with an efficient method to monitor State of Charge and State of Health for battery.

1 Technical Description

1.1 Mathematical Model

Our analysis is based on a coupled electro-thermal-aging model for lithium-iron-phosphate batteries, which is introduced in [3]. The model consists of a two RC pair electrical model, a two-state thermal model and a semi-empirical aging model.

1.1.1 Electrical Model

As shown in Figure 1, the electrical comprises an open-circuit voltage (OCV, V_{OC}), two resistor-capacitor (RC) pairs (R_1, C_1, R_2, C_2), and an ohmic resistor (R_0). The state-space model is given by:

$$\frac{\mathrm{d}SOC}{\mathrm{d}t}(t) = \frac{I(t)}{C_{bat}} \tag{1}$$

$$\frac{\mathrm{d}V_1}{\mathrm{d}t}(t) = -\frac{V_1(t)}{R_1 C_1} + \frac{I(t)}{C_1} \tag{2}$$

$$\frac{\mathrm{d}V_2}{\mathrm{d}t}(t) = -\frac{V_2(t)}{R_2C_2} + \frac{I(t)}{C_2} \tag{3}$$

$$V_t(t) = V_{OC}(SOC) + V_1(t) + V_2(t) + R_0 I(t) \tag{4}$$

where C_{bat} is the nominal capacity of the battery, $I(t)$ is the current (positive for charging), and $V_t(t)$ denotes the terminal voltage. Three state variables are SOC and voltages across the two RC pairs V_1, V_2.

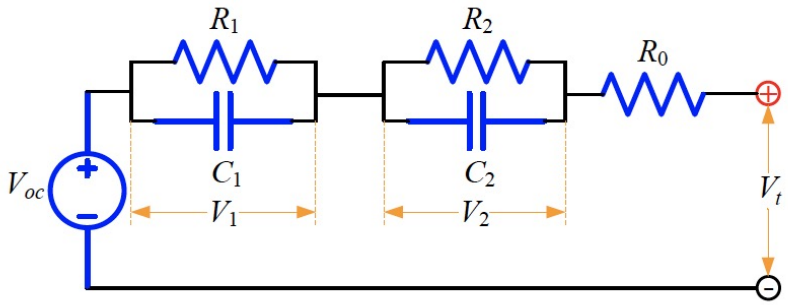


Figure 1: Electrical Model [3]

The electrical parameters are identified in [4]. In our model, we follow the equations listed in the appendix to derive these parameters based on the state of charge ($I < 0$) or discharge ($I \geq 0$).

1.1.2 Thermal Model

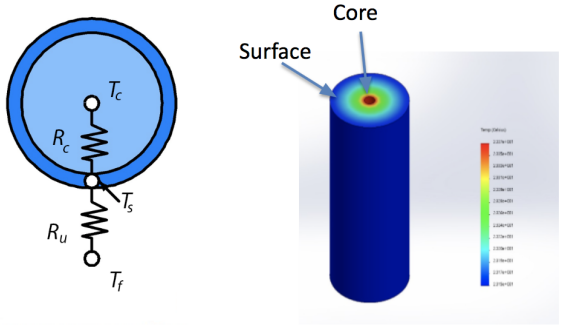


Figure 2: Two-state Thermal Model

Since the core temperature can be higher than the surface temperature under high current rates [4], a two-state thermal system was hereby introduced to capture both core and surface temperature dynamics. As sketched in Figure 2, the radial heat transfer dynamics of a cylindrical battery can be described as follow.

$$\frac{\mathrm{d}T_c(t)}{\mathrm{d}t} = \frac{T_s(t) - T_c(t)}{R_cC_c} + \frac{Q(t)}{C_c} \tag{5}$$

$$\frac{\mathrm{d}T_s(t)}{\mathrm{d}t} = \frac{T_f(t) - T_s(t)}{R_u C_s} + \frac{T_s(t) - T_c(t)}{R_c C_s} \tag{6}$$

R_c, R_u, C_c and C_s represent the heat conduction resistance, convection resistance, core heat capacity and surface heat capacity respectively, with their values shown in Table 1; two state variables are core temperature T_c and surface temperature T_s; the ambient temperature T_f is treated as uncontrollable input.

Table 1: Thermal Parameters

$R_c(KW^{-1})$	$R_u(KW^{-1})$	$C_c(JK^{-1})$	$C_s(JK^{-1})$
1.94	3.08	62.7	4.5

$Q(t) = |I(V_{OC} - V_t)|$ is heat generation including joule heating and energy dissipated by electrode over-potentials, based on equation (4), we can rewrite equation (5) as

$$\frac{\mathrm{d}T_c(t)}{\mathrm{d}t} = \frac{T_s(t) - T_c(t)}{R_c C_c} + \frac{I(t)(V_1(t) + V_2(t) + R_0 I(t)}{C_c} \tag{7}$$

1.1.3 Aging Model

The aging model is based upon a matrix of cycling tests from [2]. The experiment results suggest that capacity fade depends strongly on C-rate and temperature in the cell at low charge/discharge rates, while the sensitivity to depth-of-discharge is negligible. The semi-empirical life model adopted the following equation to describe the correlation between the capacity loss (ΔQ_b, in %) and the discharged Ah throughput(A, depends on C-rate),

$$\Delta Q_b = M(c)\exp\left(\frac{-E_a(c)}{RT_c}\right) A(c)^z \tag{8}$$

where $M(c)$ is the pre-exponential factor as a funciton of C-rate, which is denoted by c. The relation between the pre-exponential factor $M(c)$ and C-rate are shown in Table 2. The activation energy E_a and the power-law factor z are given by

$$E_a(c) = 31700 - 370.3c \quad z = 0.55 \tag{9}$$

Table 2: Pre-exponential Factor as a function of the C-rate

C-rate c	0.5	2	6	10
M	31630	21681	12934	15512

The model consider a capacity loss of 20% as the end-of-life (EOL) for an automotive battery. The corresponding Ah throughput A_{tol} and the number of cycles N are therefore calculated as below.

$$A_{tol}(c, T_c) = \left[\frac{20}{M(c)\exp\left(\frac{-E_a(c)}{RT_c}\right)}\right]^{\frac{1}{z}} \tag{10}$$

$$N(c, T_c) = \frac{3600 A_{tol}(c, T_c)}{C_{bat}} \tag{11}$$

Each cycle correspondes to $2C_{bat}$ charge throughput, and since A_{tol} is discharged Ah throughput, the total throughput including both charged and discharged Ah should be $2A_{tol}$. Based on this, the battery State-of-Health (SOH) is defined as:

$$SOH(t) = SOH(t_0) - \frac{\int_{t_0}^{t} |I(\tau) \mathrm{d}\tau}{2N(c, T_c) C_{bat}} \tag{12}$$

where t_0 denotes the initial time. SOH varies among $[0, 1]$, $SOH = 1$ correspondes to a brand new battery and $SOH = 0$ means 20% capacity loss, as known as EOL. The derivative of SOH yields the battery aging model

$$\frac{\mathrm{d}SOH}{\mathrm{d}t}(t) = -\frac{|I(t)|}{2N(c, T_c) C_{bat}} \tag{13}$$

1.1.4 Model Coupling

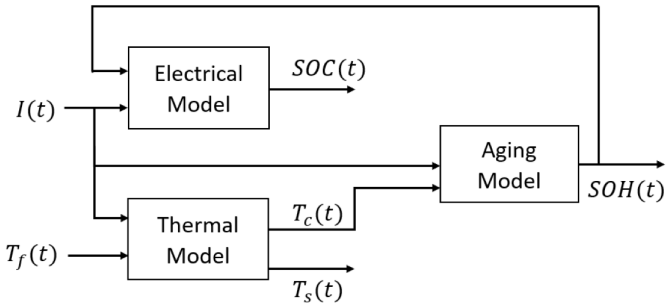


Figure 3: Electro-Thermal-Aging Model Coupling

Combining the above three subsystems, the model dynamics are summarized as below.

$$\frac{\mathrm{d}SOC}{\mathrm{d}t}(t) = \frac{I(t)}{C_{bat}} \tag{14}$$

$$\frac{\mathrm{d}V_1}{\mathrm{d}t}(t) = -\frac{V_1(t)}{R_1 C_1} + \frac{I(t)}{C_1} \tag{15}$$

$$\frac{\mathrm{d}V_2}{\mathrm{d}t}(t) = -\frac{V_2(t)}{R_2 C_2} + \frac{I(t)}{C_2} \tag{16}$$

$$\frac{\mathrm{d}T_c(t)}{\mathrm{d}t} = \frac{T_s(t) - T_c(t)}{R_c C_c} + \frac{I(t)(V_1(t) + V_2(t) + R_0 I(t)}{C_c} \tag{17}$$

$$\frac{\mathrm{d}T_s(t)}{\mathrm{d}t} = \frac{T_f(t) - T_s(t)}{R_u C_s} + \frac{T_s(t) - T_c(t)}{R_c C_s} \tag{18}$$

$$\frac{\mathrm{d}SOH}{\mathrm{d}t}(t) = -\frac{|I(t)|}{2N(c,T_c)C_{bat}} \tag{19}$$

Inputs in this model include current $I_(t)$, which is controllable, and ambient temperature $T_f(t)$, which is uncontrollable.

1.2 Observer & Simulation

All the parameters used in this project base on the parameter of an A123 Li-ion battery. In the 1 hour of simulation, the controllable input is charging rate which is a constant value $0.9C$. The uncontrollable input, ambient temperature, is assumed as a half sine wave. The input values are presented in Figure 4.

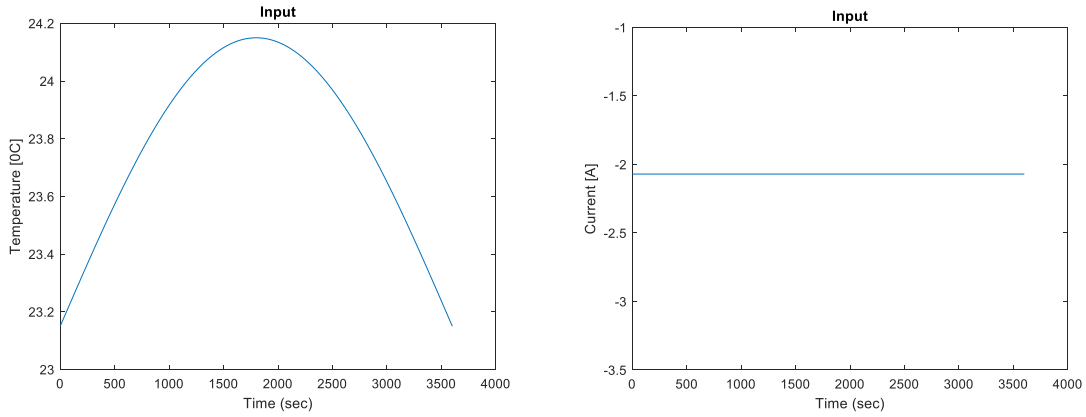


Figure 4: Systems inputs

In the simulation outputs, SOC reached 90% after 1-hour charging. The terminal voltage also increases as the SOC increases. SOH decreased as the time goes by. The core and surface temperatures of Li-ion battery changed under the effects of the current and the ambient temperature. The simulation outputs are showed in Figure 5.

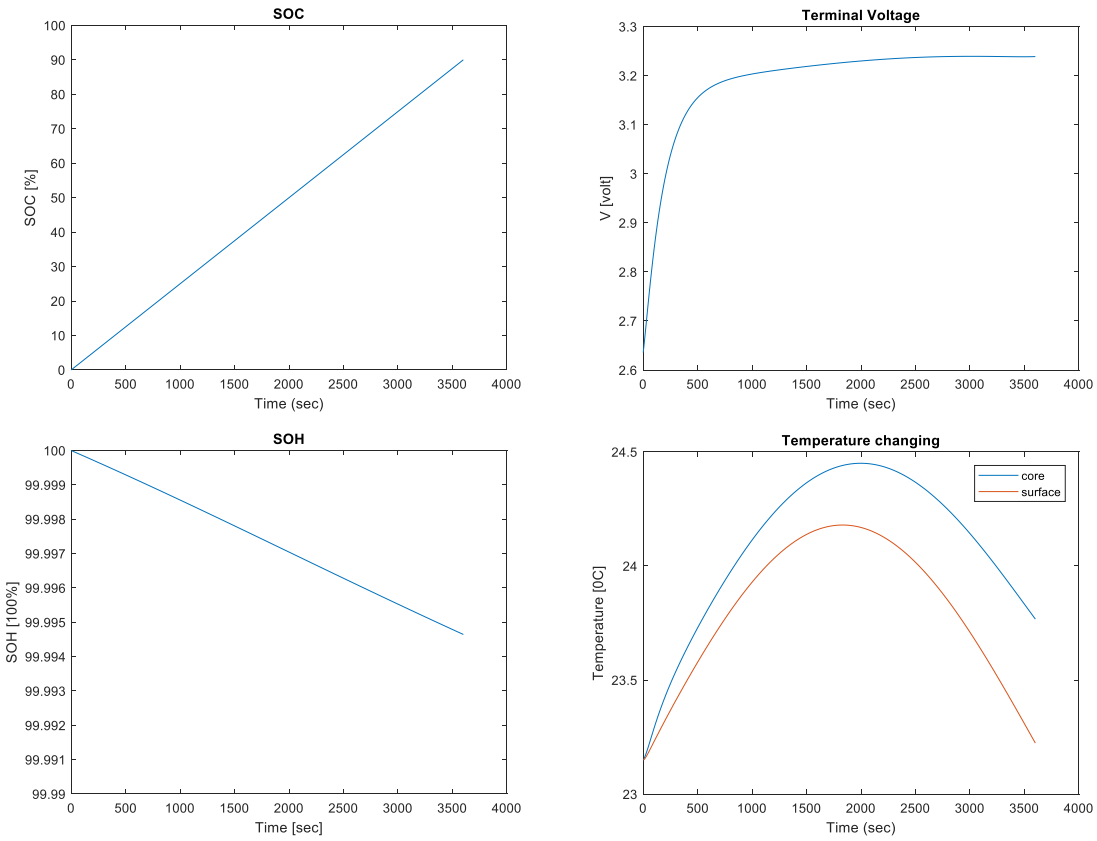


Figure 5: Systems outputs

By adding white noises in the terminal voltage and surface temperature data with SNR is equal to 60, the synthetic data is generated from this simulation model to produce measurements of terminal voltage and surface temperature. The synthetic measurements are presented in Figure 6.

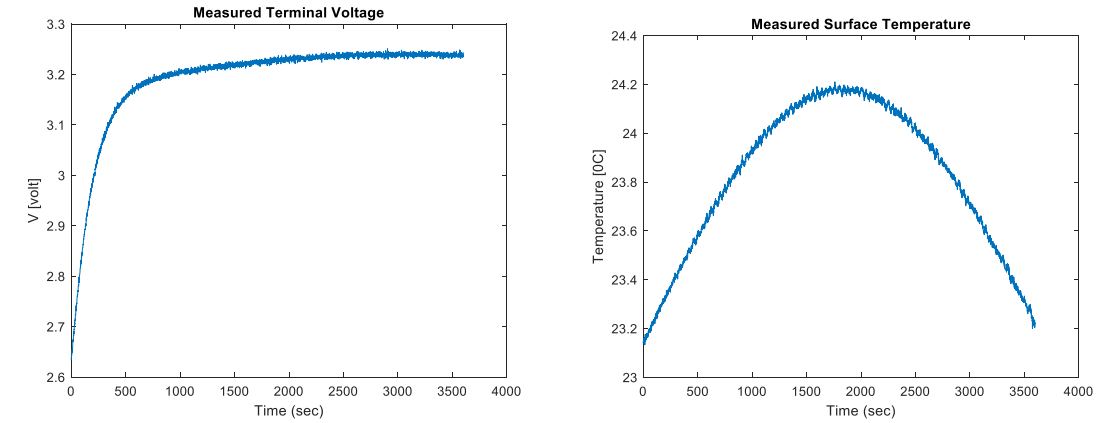


Figure 6: Measured outputs

By using Extended Kalman Filter method, three different observers are created. The

components of these three observers are showed on Table 3.

Table 3: Three observers

Observers	Control inputs	Measured inputs
Voltage-measured observer	T_f, Current	V_T
Temperature-measured observer	T_f, Current	T_S
VT-measured observer	T_f, Current	V_T, T_S

In order to test the robustness of the three observers, these observers were tested with wrong initial estimates and wrong parameter values respectively.

1.2.1 Voltage-measured observer

The test results of the Voltage-measured observer with wrong initial estimates and wrong parameter values in showed in Figure 7 and Figure 8 respectively.

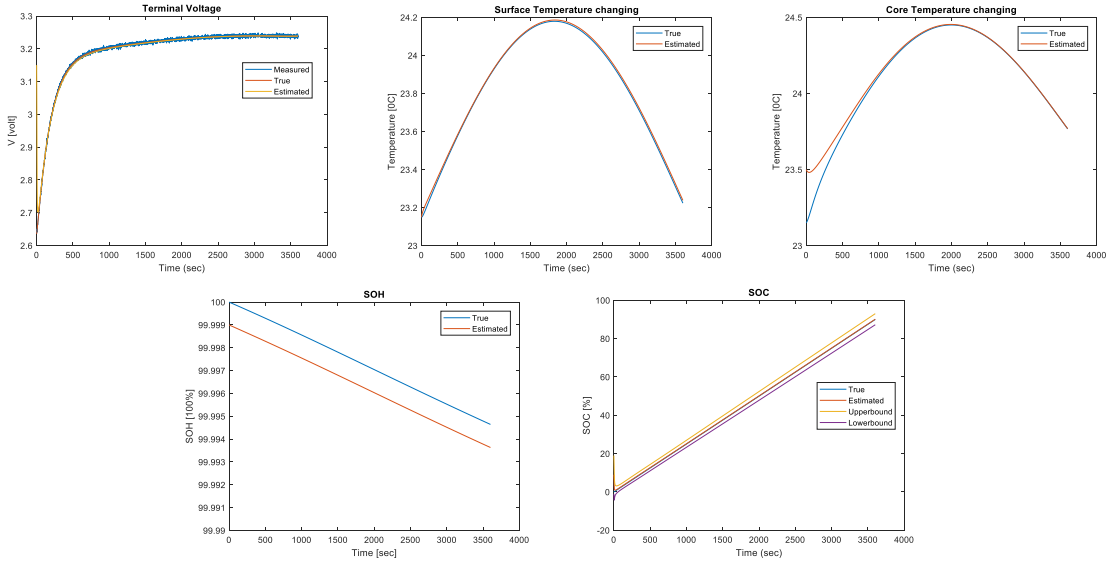


Figure 7: Estimation test of voltage-measured observer with wrong initial values

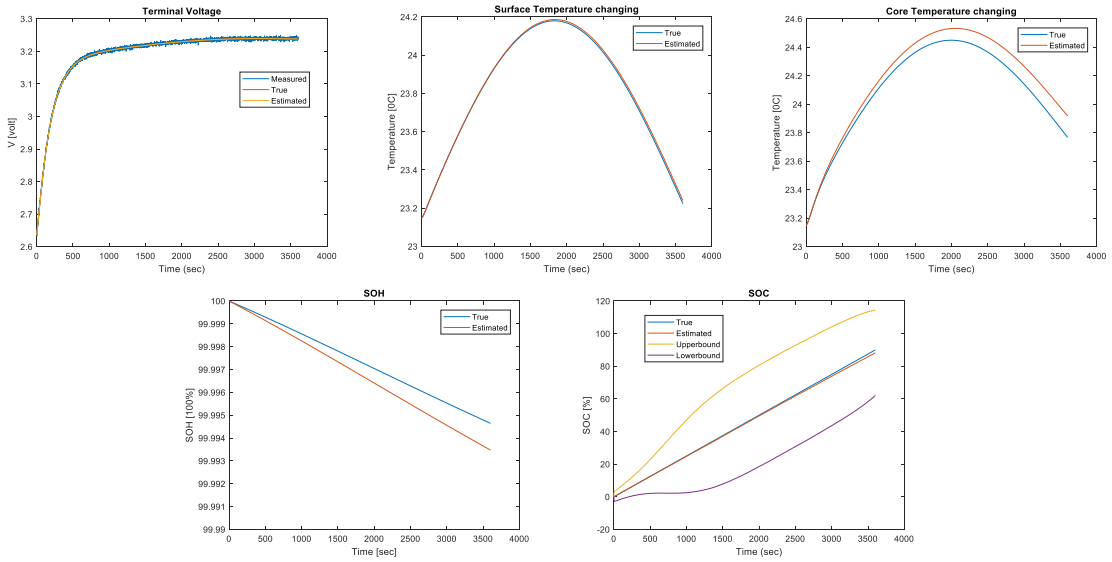


Figure 8: Estimation test of voltage-measured observer with wrong parameters

For the voltage-measured observer, the SOC and terminal voltage are well estimated. The estimated surface and core temperatures of battery converge with their true values after 1000 seconds. The estimated SOH does not converge to it true value at all. The result shows that SOH is unobservable. It is because the observability matrix of the voltage-measured observer is not full rank. The null space to the observability matrix is span($[0,0,0,1,0,0]^T$, $[0,0,0,0,1,0]^T$, $[0,0,0,0,1]^T$). It means the states of T_c, T_s, and SOH are unobservable for this observer. However, when the observer's model is very accurate, the temperatures will converge to their true values, even if temperatures are unobservable for the voltage-measured observer. It is because the thermal system is asymptotically stable, the surface and core temperatures will reach their equilibrium states after a long time.

1.2.2 Temperature-measured observer

The test results of the Temperature-measured observer with wrong initial estimates and wrong parameter values in showed in Figure 9 and Figure 10 respectively.

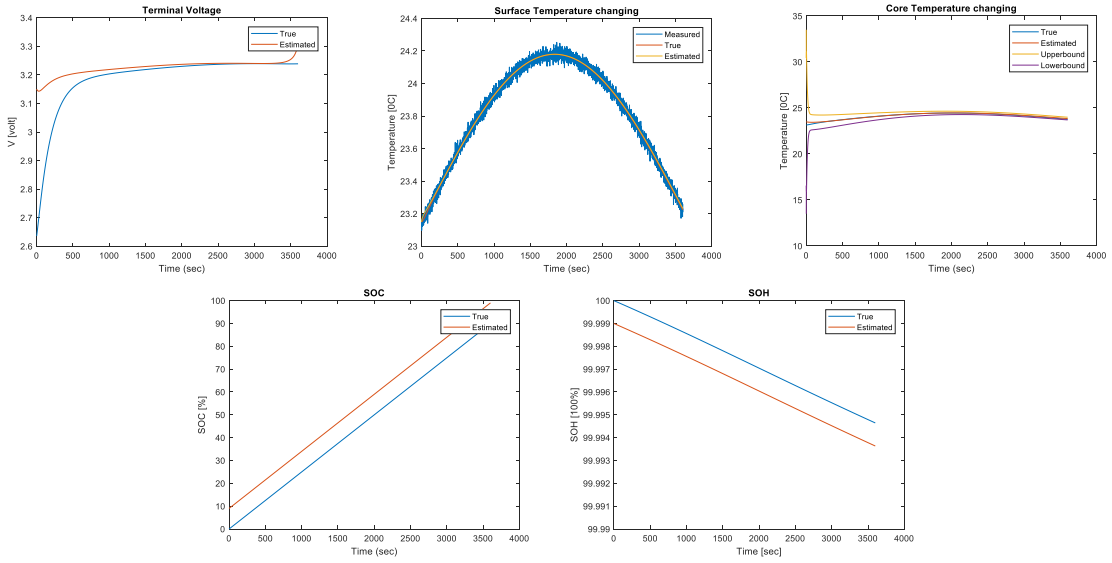


Figure 9: Estimation test of temperature-measured observer with wrong initial values

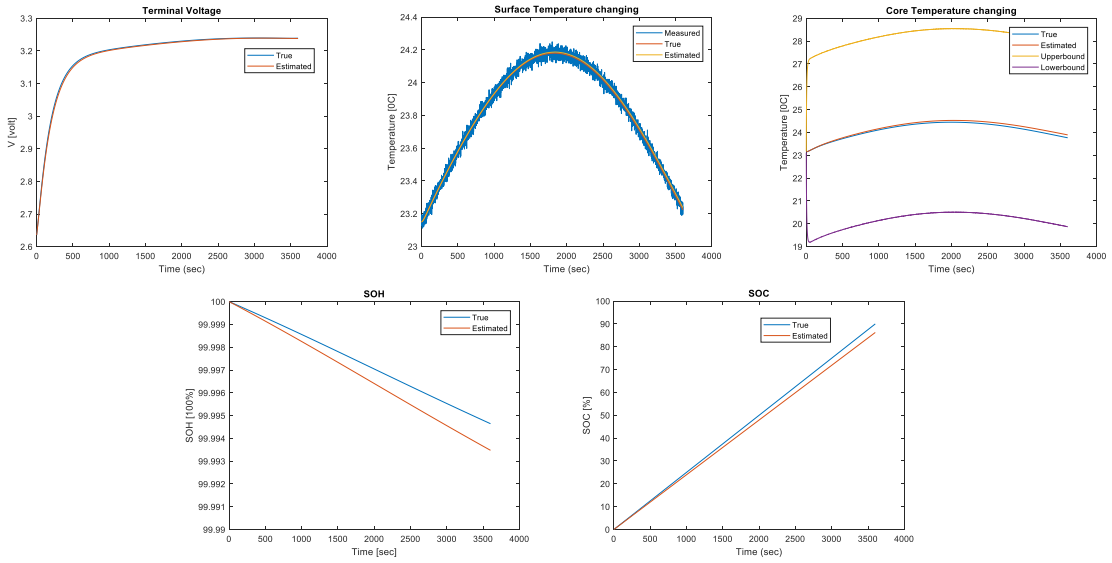


Figure 10: Estimation test of temperature-measured observer with wrong parameters

For the temperature-measured observer, the surface and core temperatures of the battery are well estimated. The estimated SOC and SOH does not converge to their true values. The test result shows that SOC and SOH are unobservable for the temperature-measured observer. The rank of the observability matrix of the observer is 4 which is not full rank. The null space to the observability matrix is span$([1,0,0,0,0,0]^T, [0,0,0,0,0,1]^T)$. The null space also indicates the unobservability of SOC and SOH for temperature-measured observer.

1.2.3 Voltage and temperature (VT) -measured observer

The test results of the VT-measured observer with wrong initial estimates and wrong parameter values in showed in Figure 11 and Figure 12 respectively.

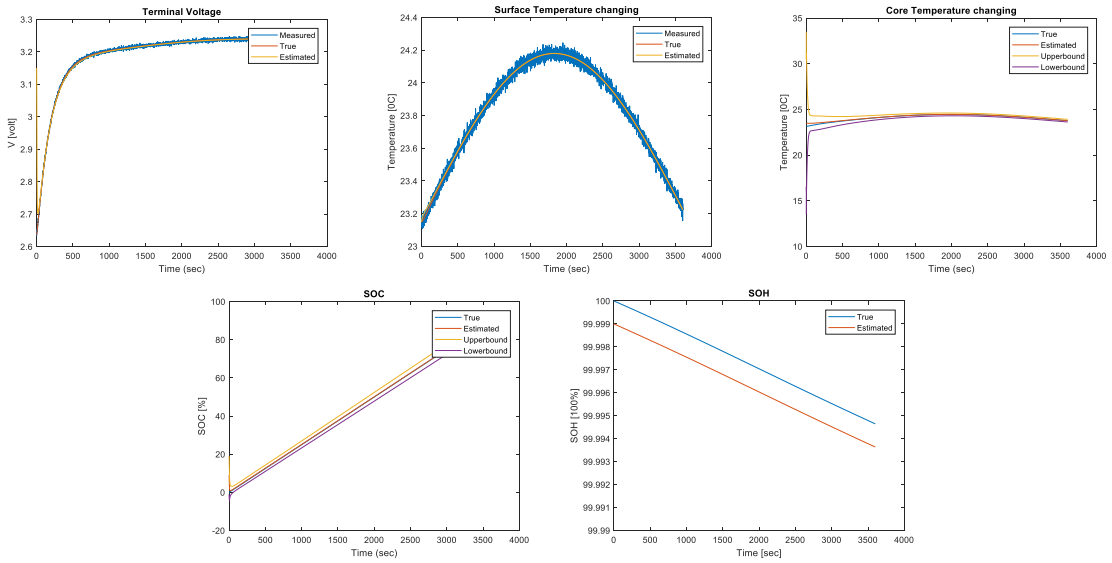


Figure 11: Estimation test of VT-measured observer with wrong initial values

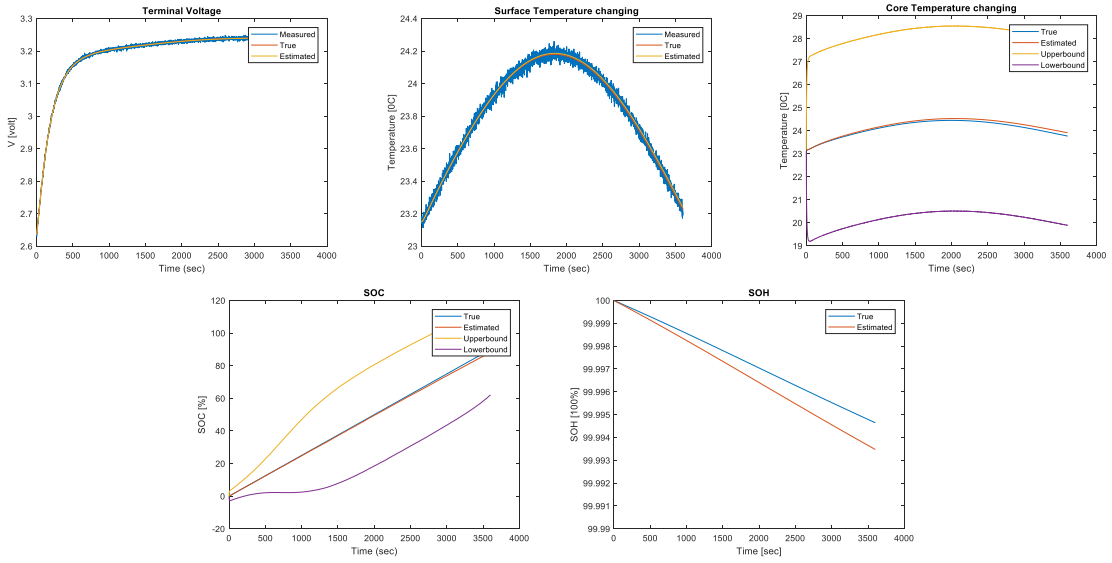


Figure 12: Estimation test of VT-measured observer with wrong parameters

For the VT-measured observer, V_T, T_C, T_S, and SOC of the battery are well estimated, except the SOH. The rank of the observability matrix of the observer is 5. The null space to the observability matrix is the span$([0, 0, 0, 0, 0, 1]^T)$ which also indicates the unobservability of the SOH.

1.2.4 Comparsion

To better evaluate the performance of these three kinds of observers, the observability of 6 battery states of the observers are presented on Table 4. The root-mean-squared error (RMSE) of three observers and two tests are showed on Table 5 and Table 6.

Table 4: The observability three observers

Observers	**SOC%**	$\mathbf{V_1}$	$\mathbf{V_2}$	$\mathbf{T_S}$	$\mathbf{T_C}$	**SOH%**
Voltage-measured observer	Yes	Yes	Yes	No	No	No
Temperature-measured observer	No	Yes	Yes	Yes	Yes	No
VT-measured observer	Yes	Yes	Yes	Yes	Yes	No

Table 5: RMSE of Extended Kalman Filter with wrong initial values

Observers	$\mathbf{V_T}$	$\mathbf{T_S}$	$\mathbf{T_C}$	**SOC%**	**SOH%**
Voltage-measured observer	0.0163	0.0060	0.0607	0.2848	0.1009
Temperature-measured observer	0.0861	0.0055	0.0554	51.3642	0.1004
VT-measured observer	0.1063	0.0059	0.0594	0.2758	0.1009

Table 6: RMSE of Extended Kalman Filter with wrong parameters in models

Observers	$\mathbf{V_T}$	$\mathbf{T_S}$	$\mathbf{T_C}$	**SOC%**	**SOH%**
Voltage-measured observer	0.0004	0.0090	0.0878	0.7590	0.0669
Temperature-measured observer	0.0034	0.0078	0.0798	51.4638	0.0669
VT-measured observer	0.0004	0.0083	0.0848	0.7551	0.0669

2 Discussion

As the results shown in the previous part, for temperature observer, SOC, SOH is unobservable. For voltage observer, T_C, T_S, SOH are unobservable. However, the temperature can be affected by the resistance of the circuit. On the other hand, the resistance will be affected by the temperature as well. Hence, the voltage observer can actually "observe" T_c and T_S by an indirect approach since after a long time, the temperature would converge.

Hence, we dicided to combine the templeature observer and voltage observer together as a VT-observer. As the results are shown, VT-measured observer has an overwhelming advantage. The observer can estimate most of the states because it the combines both V-measured observer and T-measured observer. In this project, there is not any observer can estimate the SOH. However, if an accurate battery model and initial values can be provided, the SOH values can be well estimated by an open-loop observer.

In addition, from the simulation result we notice that SOC has a positive linear relationship with charging time while SOH has a negative linear relationship with charging time. In the simulation, it requires nearly 2 hours to replenish the SOC from 0% to 90%, with the associated 0.005% SOH decay. The highest SOC is 87.4%, and the lowest SOH is 99.995% compared to the original status. Such a trend indicates that charging time and battery health is a trade off. This is interesting, since a balance between efficiency and safety, is

raised. If the highest efficency is needed, then the charging time should be minimized; If the highest safety is needed, then the aging condition should be minimized. However, in real-word cases, a balance point need should be find in different scenarios, which leads to a new topic – optimal control for battery charging. Since batteries are storing most of the energy that we use in our daliy life, the observer we developed, as such a useful tool for the optimal control problems, will definitiely advance sustainablity in energy systems. Hence it can be noticed that state estimation is a really powerful tool for maintaining the energy system. In the future, a student team can focus on optimal control of the battery charging, based on the observer we developed.

Executive Summary

In this project, we built the electric model, thermal model, and aging model, and implemented the open-loop simulation as well as state estimation using EKF. The goal of this project is to develop a battery observing system with high efficiency and robustness. Since with the accurate initial values and model parameters, the open-loop observer can observe the SOC and SOH pretty well, the trend of those curves are exactly as we expected. However, with wrong inital estimates and wrong parameter values, SOC cannot be observed by temperature observer, and SOH cannot be observed by both observers. Hence, for SOC in this scenario, it can be well-observed using our VT-observer; SOH can be observed using open-loop observer with correct initial estimates and parameters. Future work will focus on the optimal control of battery charging based on these observers we developed.

References

[1] C. R. Gould, C. M. Bingham, D. A. Stone, and P. Bentley, "New battery model and state-of-health determination through subspace parameter estimation and state-observer techniques," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 3905–3916, 2009.

[2] J. Wang, P. Liu, J. Hicks-Garner, E. Sherman, S. Soukiazian, M. Verbrugge, H. Tataria, J. Musser, and P. Finamore, "Cycle-life model for graphite-lifepo4 cells," *Journal of Power Sources*, vol. 196, no. 8, pp. 3942–3948, 2011.

[3] H. E. Perez, X. Hu, S. Dey, and S. J. Moura, "Optimal charging of li-ion batteries with coupled electro-thermal-aging dynamics," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 9, pp. 7761–7770, 2017.

[4] H. E. Perez, J. B. Siegel, X. Lin, A. G. Stefanopoulou, Y. Ding, and M. P. Castanier, "Parameterization and validation of an integrated electro-thermal cylindrical lfp battery model," in *ASME 2012 5th Annual Dynamic Systems and Control Conference joint with the JSME 2012 11th Motion and Vibration Conference*. American Society of Mechanical Engineers, 2012, pp. 41–50.

Appendix

The parameters of equation (1)-(4) are identified as follow [4], which varied across the state of charge ($I < 0$) or discharge ($I \geq 0$).

$$R_1 = \begin{Bmatrix} R_{1d} & I \geq 0 \\ R_{1c} & I < 0 \end{Bmatrix} \tag{20}$$

$$R_{1_*} = (R_{10_*} + R_{11_*}(SOC) + R_{12_*}(SOC)^2) exp\left(\frac{T_{refR_1*}}{T_m - T_{shiftR_1*}}\right) \tag{21}$$

Table 7: PARAMETRIC R_1 FUNCTION PARAMETERS

R_{10_d}	R_{10_c}	R_{11_d}	R_{11_c}	R_{12_d}
7.1135e-4	0.0016	-4.3865e-4	-0.0032	2.3788e-4
R_{12_c}	T_{refR_1d}	T_{refR_1c}	T_{shiftR_1d}	T_{shiftR_1c}
0.0045	347.4707	159.2819	-79.5816	-41.4578

$$R_2 = \begin{Bmatrix} R_{2d} & I \geq 0 \\ R_{2c} & I < 0 \end{Bmatrix} \tag{22}$$

$$R_{2_*} = (R_{20_*} + R_{21_*}(SOC) + R_{22_*}(SOC)^2) exp\left(\frac{T_{refR_2*}}{T_m}\right) \tag{23}$$

Table 8: PARAMETRIC R_2 FUNCTION PARAMETERS

R_{20_d}	R_{20_c}	R_{21_d}	R_{21_c}
0.0288	0.0113	-0.073	-0.027
R_{22_d}	R_{22_c}	T_{refR_2d}	T_{refR_2c}
0.0605	0.0339	16.6712	17.0224

$$C_1 = \begin{Bmatrix} C_{1d} & I \geq 0 \\ C_{1c} & I < 0 \end{Bmatrix} \tag{24}$$

$$C_{1_*} = C_{10_*} + C_{11_*}(SOC) + C_{12_*}(SOC)^2 + (C_{13_*} + C_{14_*}(SOC) + C_{15_*}(SOC)^2)T_m \tag{25}$$

Table 9: PARAMETRIC C_1 FUNCTION PARAMETERS

C_{10_d}	C_{10_c}	C_{11_d}	C_{11_c}
335.4518	523.215	3.1712e+3	6.4171e+3
C_{12_d}	C_{12_c}	C_{13_d}	C_{13_c}
-1.3214e+3	-7.5555e+3	53.2138	50.7107
C_{14_d}	C_{14_c}	C_{15_d}	C_{15_c}
-65.4786	-131.2298	44.3761	162.4688

$$C_2 = \begin{Bmatrix} C_{2d} & I \geq 0 \\ C_{2c} & I < 0 \end{Bmatrix} \tag{26}$$

$$C_{2_*} = C_{20_*} + C_{21_*}(SOC) + C_{22_*}(SOC)^2 + (C_{23_*} + C_{24_*}(SOC) + C_{25_*}(SOC)^2)T_m \tag{27}$$

Table 10: PARAMETRIC C_1 FUNCTION PARAMETERS

C_{20_d}	C_{20_c}	C_{21_d}	C_{21_c}
3.1887e+4	6.2449e+4	-1.1593e+5	-1.055e+5
C_{22_d}	C_{22_c}	C_{23_d}	C_{23_c}
1.0493e+5	4.4432e+4	60.3114	198.9753
C_{24_d}	C_{24_c}	C_{25_d}	C_{25_c}
1.0175e+4	7.5921e+3	-9.5924e+3	-6.9365e+3

Acknowledgments

The authors would like to thank Professor Scott Moura for guidance in this project.

About the authors

Junzhe Shi is a Systems Engineering student (Master of Science) at University of California, Berkeley.

Franklin Zhao is a Systems Engineering student (Master of Science) at University of California, Berkeley.

Ruitong Zhu is a Systems Engineering student (Master of Science) at University of California, Berkeley.

Xin Peng is a Systems Engineering student (Master of Engineering) at University of California, Berkeley.

Part V

Health and Environment

Mathematical Modeling of An Ecosystem Network

Kyra Chang, Tiffany Chang, Emily Farrar, Nate Tsang

Abstract

Complex systems are present in every facet of our environment. The vast biodiversity of life and its interconnected networks are an area of research for ecologists and modelers alike. In our project, we demonstrate the effects of various controllable inputs, such as adding or removing species through introduction and hunting, on the population dynamics of an ecosystem as well as three different types of species' interaction: predator-prey, competition, and commensal. The main research of the paper is focused on modeling predator-prey relationships using food chain dynamics. In our first mathematical model, we predict how populations evolve over time given interactions with each other and the environment. Our second model utilizes dynamic programming to determine the optimal control strategy that achieves a target population size given a set of boundary conditions.

Introduction

Motivation and Background

The stability of complex systems is central to the understanding of networks and communities that have many interacting components. Examples of interactions include species in ecology, human groups in sociology, markets in economics, computers in a cyber-physical system, and neurons in our brains. The complexity of interactions between species within an ecosystem and the question of dynamic stability is a topic of both theoretical and empirical research in population ecology. In 1958, Charles Elton proposed the idea that simple communities are less stable than diverse ones in that they are more vulnerable to invasions and more likely to experience larger fluctuations as a result. There exists a complexity-stability paradox where randomly generated ecological systems decreased in stability as they increased in diversity in theory, but the opposite pattern has been observed in nature, leading to the debate between diversity versus stability that has been studied by ecologists for the past 45 years. Recent studies are starting to explore dynamic interactions between the species to more accurately depict the stability of realistic food web configurations and structures.

The relative state of a system is rated by three factors: richness, diversity, and complexity. Richness refers to the total number of species in the system. Diversity is an index that evaluates both species richness as well as the relative abundance of each species. Complexity is a combined measure of the number of species and the strength of interaction and level of connectivity between the species. To measure stability, the main factors to consider include

equilibrium, resistance, and resilience of the system. The system is considered to be stable if it returns to equilibrium after encountering a perturbation. This can be a local as well as global equilibrium point. The resistance of the system is the ability of it to resist change when perturbed. A system with a higher level of resistance will be able to resist change more easily than a system with a lower level of resistance. Lastly, the resilience of a system measures the ability of the system to recover after being perturbed. A system with higher resilience will return to equilibrium quicker than a lower resilience system.

Relevant Literature

The predator-prey model was first proposed by Alfred Lotka and Vito Volterra in the 1920s. This model has become the foundation of mathematical ecology and is widely used to model predator-prey relationships. The Lolta-Volterra (LV) model is a pair first-order, nonlinear differential equations that describe the dynamics of biological systems in which two species interact, shown in (1, 2) [4].

$$\dot{x} = \alpha x - \beta xy \tag{1}$$

$$\dot{y} = -\gamma y + \delta xy \tag{2}$$

Where:
x is the number of prey
y is the number of predators
α is the growth rate constant for prey
β is the death rate constant for prey
δ is the growth rate constant for predators
γ is the death rate constant for predators

LV is a simple, deterministic model that relies on many assumptions, such as the following:

- Considers only two species.
- Prey have unlimited food so there is no interspecies competition.
- Predator's food supply depends solely on the number of prey. As such, the predators feed on prey and the food that the prey eat.
- The rate of change of population size is proportional to population size.
- Predators have unlimited appetite.
- The environment is held constant.

Since its proposal, researchers have altered the LV model to represent phenomena not previously considered, such as spatial heterogeneity, stochasticity, intra-species competition, and the dynamics of two or more species. Interestingly, much of the recent literature continues to rely on the basic LV model that is described in (1) and (2), without any augmentations.

In the paper by Gravel et. al (2015) [1], the authors study the trade-off between species diversity, connectance, and interaction strength in determining the stability of a simulated

complex ecosystem. For each simulation, random interaction coefficients were applied before solving for the equilibrium of the matrix. When only the positive species densities remained, they subjected each matrix to a gradient of connectivity (termed 'dispersal' in the paper) before numerically solving for the largest eigenvalue. They found that their meta-ecosystem dynamics stabilized because of the effects of dispersal on the Jacobian matrix and its corresponding eigenvalues. Cases with low dispersal are linearly affected, but in cases with high dispersal, the effects are dependent on the ecological size of the metacommunity. Factors such as species interaction and spatially structured landscapes (e.g., fragmentation and/or habitat loss) can affect biodiversity in the ecosystem.

The stability of species coexistence is an important factor in maintaining the diversity of an ecosystem. A paper by Chesson (2000)[7] evaluates how intra- and interspecies interactions affect diversity maintenance, defined as the coexistence of species having similar ecology in the same spatial region. Coexistence mechanisms can be defined in two main ways: an equalizing mechanism, which minimizes average fitness differences between species, or a stabilizing mechanism which increases competition between members of a single species relative to competition between different species. Most studies focus exclusively on equalizing mechanisms, but Chesson argues that models of unstable coexistence would be more robust if they also included mechanisms that model interactions between different species.

The paper by Chesson (2000) uses the LV model to discuss the basic principles of competition. In terms of competition coefficients, the LV equations are written as (3):

$$\frac{1}{N_i} \times \frac{dN_i}{dt} = r_i(1 - \alpha_{ii}N_i - \alpha_{ij}N_j) \quad i = 1, 2 \quad j \neq i \tag{3}$$

Where α_{ii} and α_{ij} are absolute intraspecific and interspecific competition coefficients.

When parameterized in this method, intraspecific competition must be greater than interspecific competition. While LV models growth rates are linear functions of density, models with nonlinear growth rates can be written in the above form by making the competition coefficients functions of density (α_{ij}=$f_{ij}N_i, N_j$) as long as the resident is at equilibrium and the invader is at zero. A species is defined as an invader if it is at a low density compared to the rest of the community.

A paper by Rafikov et al. (2005)[5] introduces two steps to solve the the optimal pest control problem. The first step is to move the ecosystem pest at an equilibrium state below the economic injury level by introducing the Pontryagin Maximum Principle. The Hamilton-Jacobi-Bellman equation, based on the theory of dynamic programming, is used in this step to obtain the pest control strategy through natural enemies' introduction and stabilize the pest density. Both of the steps are based on the predator-prey model.

For our implementation of optimal control problem, there is a complication with the actual feasible value function. Sundström et al.[6] addresses this issue by introducing a method for the optimal control of a one-dimensional dynamic model that avoids landing in an infeasible solution space. In particular, after the preallocation of the Value Function table, a new boundary based on the target number of species at time N is calculated by proceeding backwards in time for $k = N, N-1, \ldots, 0$, which is $x_{high}(t)$ and $x_{low}(t)$ in (1).

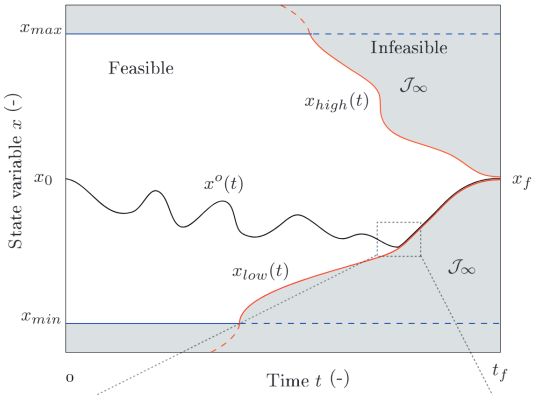


Figure 1: Value Function table with a new boundary

Focus of this Study

The goal of our research is to observe how ecological communities evolve over time subject to chosen inputs and perturbations. We will be focusing on the effects of how the stability of a system changes as a function of increasing complexity (e.g., network size, connectivity, and interaction strength). We will also delve into how ecosystems respond to external management actions.

Technical Description

Our goal is to determine the optimal control strategy to achieve a desired populations in a complex ecosystem. To achieve this, we will first develop a mathematical model to represent the dynamical system. We will then optimally control this system to achieve a target population size at a final time N with minimal intervention u.

Phase 1: Develop equations to describe the dynamical system.

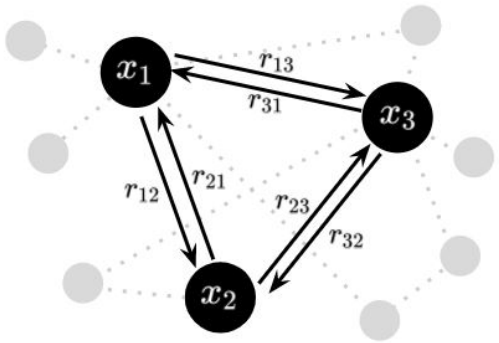


Figure 2: Schematic diagram of a general three-species ecosystem

Population Relationships

We represent an ecosystem as a directed graph network with n nodes as shown in Figure 2. Each node represents a population of species x_i, while each edge represents a relationship between species x_i and species x_j. As such, this model is more general and complex than a food chain, which has a radial nature. There are three types of inter-species relationships that we will explore: predator-prey, competition, and commensal.

The parameters that describe the inter- and intraspecies relationships are organized into a matrix R, as shown in (4) [2]. The diagonal entries r_{ii} represent the birth or death rates for species x_i. It is assumed that only the species at the bottom of the food chain has a natural birth rate, while all other species have natural death rates.

$$R = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ r_{21} & r_{22} & \dots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \dots & r_{nn} \end{bmatrix} \qquad \text{where } r_{ij} \in \mathbb{R} \tag{4}$$

Interspecies relationships are characterized by the off-diagonal parameters r_{ij}. Positive parameters (r_{ij}) indicate that populations of x_j are beneficial to x_i, while negative parameters ($-r_{ij}$) indicate that populations of x_j are adversarial to x_i. For example, if we consider a predator-prey relationship between two species x_1 (e.g. elk) and x_2 (e.g. wolves), r_{12} is positive, while $-r_{21}$ is negative. This is represented by the original Lotka-Volterra model, represented in equations (5, 6). For equations (5-10), x_1 and x_2 must be positive real numbers.

$$\dot{x}_1 = x_1(r_{11} - r_{12}x_2) \tag{5}$$

$$\dot{x}_2 = x_2(-r_{22} + r_{21}x_1) \tag{6}$$

Commensalism is a +/+ relationship where each species benefits from the other's existence (e.g. birds and flowers). In this scenario, both r_{12} and r_{21} are positive. This phenomena is represented with the equations (7, 8).

$$\dot{x}_1 = x_1(r_{11} + r_{12}x_2) \tag{7}$$

$$\dot{x}_2 = x_2(-r_{22} + r_{21}x_1) \tag{8}$$

Finally, competition is a -/- relationship where both species are negatively impacted by the other's existence. An example of competition would be two animals that eat the same food or occupy the same physical space. In this scenario, both $-r_{12}$ and $-r_{21}$ are negative. This phenomena is represented with the equations (9, 10).

$$\dot{x}_1 = x_1(r_{11} - r_{12}x_2) \tag{9}$$

$$\dot{x}_2 = x_2(-r_{22} - r_{21}x_1) \tag{10}$$

State-Space Equations

In a directed network, each species may exhibit one or more of the relationships described above. As such, the population dynamics for each species x_i in a system can be represented using the general equation (11).

$$\dot{x}_i = \sum_{i \neq j} (r_{ij} x_i x_j) + r_{ii} x_i \tag{11}$$

We describe the networked system using the generalized state-space equations for three species in (12). The following two examples shows how (12) can be utilized to explain different network types.

$$\begin{aligned} \dot{x}_1 &= x_1(r_{11} + r_{12}x_2 + r_{13}x_3) \\ \dot{x}_2 &= x_2(r_{21}x_1 + r_{22} + r_{23}x_3) \\ \dot{x}_3 &= x_3(r_{31}x_3 + r_{32}x_2 + r_{33}) \end{aligned} \tag{12}$$

Next, we adapt the general state-space equations to reflect two types of networks: commensal and food-chain.

Example 1: Three species commensal network

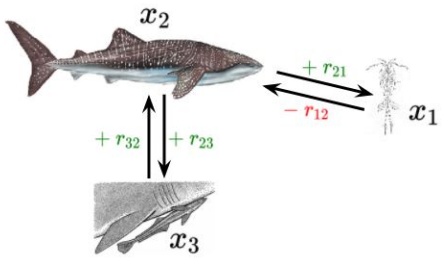


Figure 3: Commensal Network

Figure 3 is an example of a commensal relationship between a whale shark and remora, as well as a predator-prey relationship, where the whale shark eats the krill. In this example, the remora receives scraps of prey dropped by the shark and consumes parasites on the shark. In return, it is protected from predators and receives a constant flow of water across its gills. This relationship is represented by two positive relationship parameters, in green, to represent that they both benefit from the relationship. We assume there is no interaction between the remoras and krill. The dynamics of this network can be described by (13).

$$\begin{aligned} \dot{x}_1 &= x_1(r_{11} - r_{12}x_2) \\ \dot{x}_2 &= x_2(r_{21}x_1 + r_{22} + r_{23}x_3) \\ \dot{x}_3 &= x_3(r_{32} + r_{33}) \end{aligned} \tag{13}$$

Example 2: Three species food-chain network

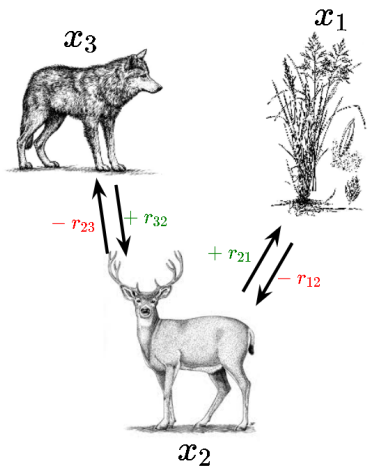


Figure 4: Food Chain Network

The three species example in Figure 4 consists of coyote, a deer, and a blue grass population. In a food chain, the top predator and bottom prey do not directly interact with each other. The colored r_{ij} values in Figure 4 represent the intraspecies dynamics.

$$\begin{aligned} \dot{x}_1 &= x_1(r_{11} - r_{12}x_2) \\ \dot{x}_2 &= x_2(r_{21}x_1 - r_{22} - r_{23}x_3) \\ \dot{x}_3 &= x_3(r_{32}x_2 - r_{33}) \end{aligned} \tag{14}$$

(14) is the set of state-space equations for the blue grass, deer, and coyote populations. Unlike the commensal example where the remora and the shark have positive interspecies relationships, parameter r_{23} is negative, indicating that the deer population decreases as the coyote population increases. The remainder of this study will consider a food chain network for either two or three species.

Phase 2: Evaluate Equilibrium and Stability.

Using the food chain scenario for three species, we can find where populations reach a steady-state by identifying the equilibrium points. This is achieved by setting (14) to zero, and solving for x_1^{eq}, x_2^{eq} and x_3^{eq} as seen in (15).

$$\begin{aligned} 0 &= x_1^{eq}(r_{11} - r_{12}x_2^{eq}) \\ 0 &= x_2^{eq}(r_{21}x_1^{eq} - r_{22} - r_{23}x_3^{eq}) \\ 0 &= x_3^{eq}(r_{32}x_2^{eq} - r_{33}) \end{aligned} \tag{15}$$

There are two sets of equilibrium points, shown in (16) and (17), which are a function of the parameters and are consistent with [3].

$$\text{Set 1: } x_1^{eq} = 0,\ x_2^{eq} = 0,\ x_3^{eq} = 0 \tag{16}$$

$$\text{Set 2: } x_1^{eq} = \frac{r_{22}}{r_{21}},\ x_2^{eq} = \frac{r_{11}}{r_{12}},\ x_3^{eq} = 0 \tag{17}$$

Next, we determine the stability of these equilibrium points by evaluating the Jacobian (18) for the food chain model with respect to the equilibrium values. The system is unstable if the eigenvalues have both negative and positive real parts, asymptotically stable if they have all negative real parts, and marginally stable they have all non-positive real parts.

$$\mathrm{J} = \begin{bmatrix} r_{11} - r_{12}x_2 & -r_{12}x_1 & 0 \\ r_{21}x_2 & r_{21}x_1 - r_{22} - r_{23}x_3 & -r_{23}x_2 \\ 0 & r_{32}x_3 & r_{32}x_2 - r_{33} \end{bmatrix} \tag{18}$$

Substituting in the first set of equilibria points (16) into the Jacobian yields the matrix (19) and eigenvalues (20). This is unstable because the first eigenvalue is positive, while the second and third are negative.

$$\mathrm{J} = \begin{bmatrix} r_{11} & 0 & 0 \\ 0 & -r_{22} & 0 \\ 0 & 0 & -r_{33} \end{bmatrix} \tag{19}$$

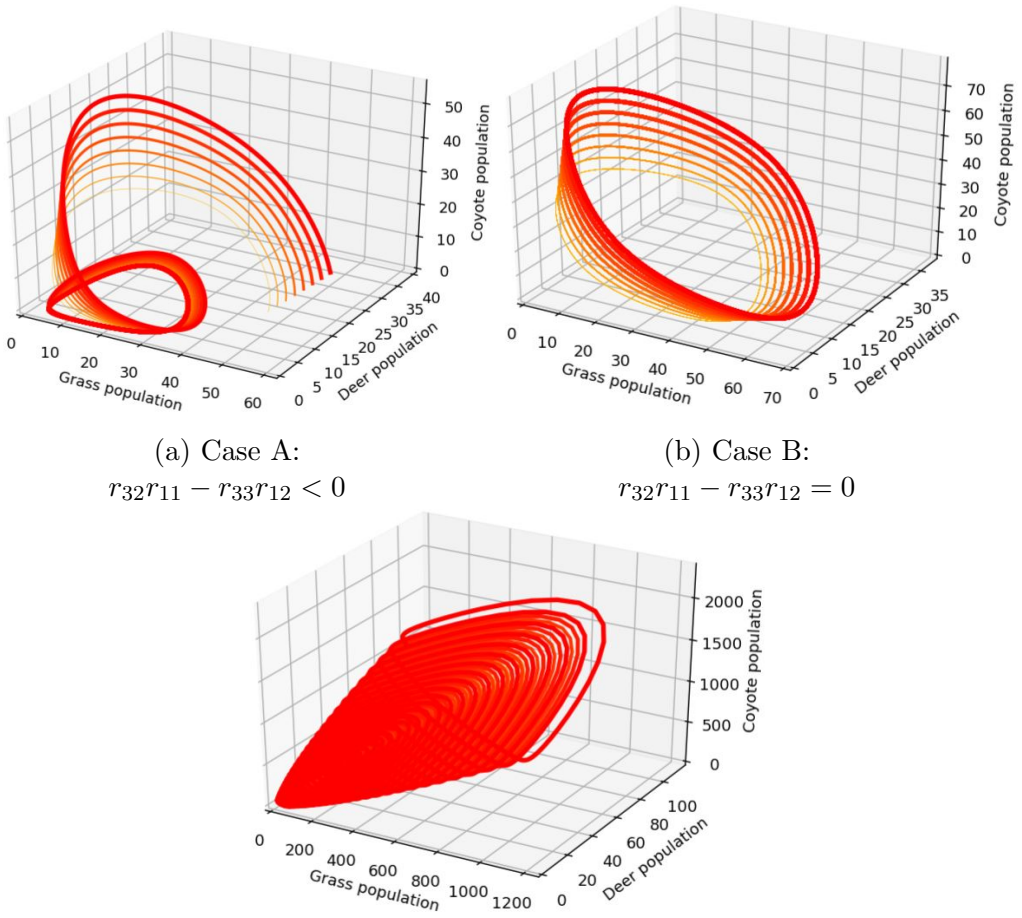
$$\lambda(J) = (r_{11}, -r_{22}, -r_{33}) \tag{20}$$

Substituting in the second set of equilibria points (17) into the Jacobian yields the matrix (21) and eigenvalues (22). The stability of the system is dependent on the third eigenvalue. In particular, it is marginally stable when $r_{32}r_{11} - r_{33}r_{12} \leq 0$ and unstable when $r_{32}r_{11} - r_{33}r_{12} > 0$.

$$\mathrm{J} = \begin{bmatrix} r_{11} - \frac{r_{12}r_{33}}{r_{32}} - \frac{r_{13}r_{22}}{r_{23}} & 0 & 0 \\ \frac{-r_{21}r_{33}}{r_{32}} & 0 & \frac{-r_{23}r_{33}}{r_{32}} \\ \frac{-r_{22}r_{31}}{r_{23}} & \frac{-r_{22}r_{32}}{r_{23}} & 0 \end{bmatrix} \tag{21}$$

$$\lambda(J) = i\sqrt{r_{11}r_{22}},\ \text{-}i\sqrt{r_{11}r_{22}},\ \frac{r_{32}r_{11} - r_{33}r_{12}}{r_{12}} \tag{22}$$

The phase space diagrams in Figure 5 shows the population dynamics for two marginally stable cases and one unstable case. In Cases A and B, the populations oscillate around the equilibrium point due to its non-real eigenvalues. In Figure 5a, when $r_{32}r_{11} - r_{33}r_{12} < 0$, the trajectory of the graph spirals downwards, changing from a 3D to a 2D system, depicting the extinction of the coyote population. However, as shown in Figure 5b, when $r_{32}r_{11} - r_{33}r_{12} = 0$ (indicating a lower death rate for the coyote population), the system oscillates in a 3D system without damping. Figure 5c is unstable ($r_{32}r_{11} - r_{33}r_{12} > 0$) and shows that the populations spiral towards infinity as they deviate from the equilibrium. In all graphs, the different colored orbits indicate different initial population sizes.



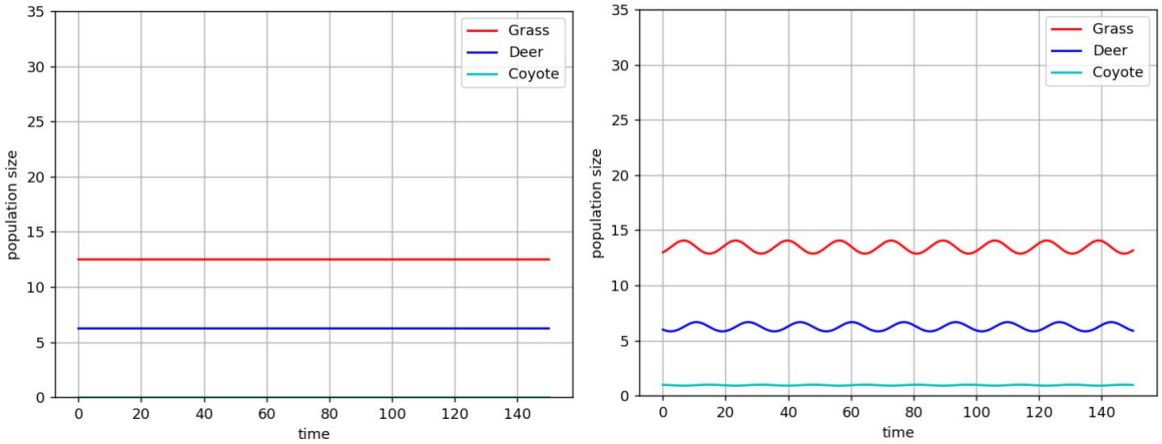
(a) Case A:
$r_{32}r_{11} - r_{33}r_{12} < 0$

(b) Case B:
$r_{32}r_{11} - r_{33}r_{12} = 0$

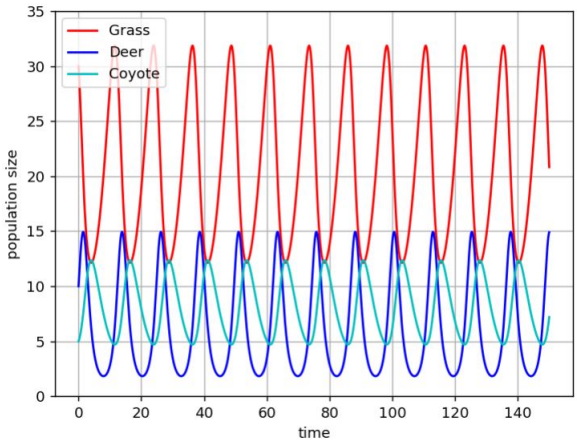
(c) Case C:
$r_{32}r_{11} - r_{33}r_{12} > 0$

Figure 5: Phase space for a three-species food chain with marginally stable (Case A, B) or unstable (Case C) equilibria.

As demonstrated in Figure 6, the farther the initial populations are from the equilibrium point, the larger the oscillations and the greater the period. Figure 6a reflects populations that initialize at the equilibrium, while 6b and 6c demonstrate non-equilibrium initial populations $x(0)$. This information also explains how the magnitude of perturbations from the equilibrium impact the dynamics.



(a) $x(0) = (12.5,\ 6.25,\ 0)$ (b) $x(0) = (13,\ 6,\ 1)$



(c) $x(0) = (30,\ 10,\ 5)$

Figure 6: Impact of initial population size on oscillatory dynamics for Case B. The more the initialization deviates from the equilibrium, the more oscillatory the dynamics.

Phase 3: Minimize optimal controls to achieve a target population at the final time step.

Optimization Problem - Two species system

Humans are not included as part of the ecosystem network, but can perturb the system through a variety of controls (i.e., management techniques). For this problem, we consider the possibility that predator x_2 individuals can be removed through hunting or introduced into the population at any time step. The removal of individuals is denoted $-u$, while the introduction of individuals is denoted u. We aim to optimize the number of population members added or removed to achieve a target population for two species at the final time step N. The optimal control u^* for ecosystem management is subjective and will vary based on the specific scenario. For our purposes, we define the optimal control strategy as one that minimizes the square value of the Euclidean norm. As such, the optimization problem

is formulated as follows:

$$\min_{u_k} J = \sum_{k=0}^{N-1} ||u_k||_2^2 \tag{23}$$

With equality constraints:

$$\begin{aligned} x_0^i &= x_{init}^i && \forall i = 1, 2 \\ x_{k+1}^1 &= \Delta t \cdot x_k^1[r_{11} + r_{12}x_k^2] + x_k^1 && \forall k = 0, 1, \ldots, N-1 \\ x_{k+1}^2 &= \Delta t \cdot x_k^2[r_{21}x_k^1 + r_{22}] + x_k^2 + \Delta t \cdot u_k && \forall k = 0, 1, \ldots, N-1 \end{aligned}$$

And inequality constraints:

$$\begin{aligned} u_{min} &\leq u_k \leq u_{max} && \forall k = 0, 1, \ldots, N-1 \\ x_{min} &\leq x_k \leq x_{max} && \forall k = 0, 1, \ldots, N \\ x_N^{1,min} &\leq x_N^1 \leq x_N^{1,max} \\ x_N^{2,min} &\leq x_N^2 \leq x_N^{2,max} \end{aligned}$$

Where:

k is the discrete time index
N is the time horizon
Δt is the size of each timestep
$x_k = [x_k^1, x_k^2]$ is the state at time k
x_k^i is the state for species i at time k, $i \in 1, 2$
u_k is the control decision applied at time k
x_{init}^i is the state for species i at initial time $k = 0$, $i \in 1, 2$
x_{min}, x_{max} are the minimum and maximum population for every species for all time k
u_{min}, u_{max} are the minimum and maximum control value for species 2 for all time k
$x_N^{i,min}$, $x_N^{i,max}$ are the minimum and maximum population for species i at time N, $i \in 1, 2$

Dynamic Programming

Since the dynamics of the system evolve with time, Dynamic Programming is used to break the multistage decision problem into subproblems. The value function for this problem is defined as follows:

Let $V_k(x_k)$ denote the minimum control of species x_k^2 from time step k to N, given that the current state is the population sizes for each species $x_k = [x_k^1, x_k^2]$.

The Principle of Optimality (PoO) can be written in recursive form as:

$$V_k(x_k) = \min_{u_k}\{||u_k||_2^2 + V_{k+1}(x_{k+1})\} \tag{24}$$

With the boundary conditions:

$$V_N(x_N) = \begin{cases} 0 & \text{if } x_N^{1,min} \leq x_N^1 \leq x_N^{1,max}, x_N^{2,min} \leq x_N^2 \leq x_N^{2,max} \\ \text{inf} & \text{otherwise} \end{cases} \tag{25}$$

The optimal control action is:

$$u^*(x_k) = arg\min_{u_k}\{||u_k||_2^2 + V_{k+1}(x_{k+1})\} \tag{26}$$

This formulation ensures that at every timestep in the time horizon k to N, the control input u_k and the corresponding states x_k are optimized. The dynamic program calculates 3D value table, with the dimensions time k, the state of species x_1 and the state of species x_2. To limit the computational effort, the value table is discretized into a grid. Therefore, the recursive PoO (24) involves computing x_{k+1}, and interpolating of $V_{k+1}(x_{k+1})$ onto the grid.

1. Computing x_{k+1}
   For each timestep, all possible controls u_k are evaluated for each possible state x_k. To simplify the formulation of u_{grid}, we rearrange each equality constraints as an upper and lower limit on u_k.

$$x_{min} \leq x_{k+1}^2 = \Delta t \cdot u_k + \Delta t\cdot \tag{27}$$

$$x_{min} \leq x_{k+1}^2 = \Delta t \cdot u_k + \Delta t\cdot \tag{28}$$

$$x_k^2[r_{21}x_k^1 + r_{22}] + x_k^2 \leq x_{max} \tag{29}$$

$$x_{min} \leq \Delta t \cdot u_k + \Delta t\cdot \tag{30}$$

$$x_k^2[r_{21}x_k^1 + r_{22}] + x_k^2 \leq x_{max} \tag{31}$$

$$x_{min} - x_k^2 - \Delta t \cdot x_k^2[r_{21}x_k^1 + r_{22}] \leq \Delta t \cdot u_k \leq x_{max} - x_k^2 - \Delta t \cdot x_k^2[r_{21}x_k^1 + r_{22}] \tag{32}$$

$$\frac{x_{min} - x_k^2}{\Delta t} - x_k^2[r_{21}x_k^1 + r_{22}] \leq u_k \leq \frac{x_{max} - x_k^2}{\Delta t} - x_k^2[r_{21}x_k^1 + r_{22}] \tag{33}$$

2. Interpolation of $V_{k+1}(x_{k+1})$
   For each x_{k+1} that is not on a grid-point (i.e., is not an integer), a 2D interpolation is required to calculate $V_{k+1}(x_{k+1})$.

Boundary Conditions Towards the Final Target Size

Inspired by [6], the boundaries for our model can be calculated using the following algorithm:

1. Initialize with the lower bound of the partially constrained final state. $x_{k,low}^1 = x_{N,min}^1$ and $x_{k,low}^2 = x_{N,min}^2$
2. Proceed backward in time for $k = N-1, \ldots, 0$:
   (a) Assume the states are unconstrained and solve the fixed point problem as shown:
      i. Initialization:

$$x_{k,low}^{1,st=0} = x_{k+1,low}^1 x_{k,low}^{2,st=0} = x_{k+1,low}^2 \tag{34}$$

ii. Iterate over st until a specified tolerance is achieved:

$$x_{k,low}^{1,st+1} = x_{k+1,low}^{1} - \Delta t * x_{k,low}^{1,st} * (R_{11} + R_{12} * x_{k,low}^{2,st})$$
$$x_{k,low}^{2,st+1} = x_{k+1,low}^{2} - \max_{u_k}\{\Delta t * (x_{k,low}^{2,st} * (R_{22} + R_{21} * x_{k,low}^{1,st}) + u_k)\}$$

(b) Check whether the solution violates the state constraints.

(c) If the constraints are violated, solve the general problem:

$$x_{k,low}^{1} = x_{min} \tag{35}$$

$$\begin{aligned} &\max_{u_k}\{\Delta t * (x_{k,low}^{2,st} * (R_{22} + R_{21} * x_{k,low}^{1,st}) + u_k)\} \\ s.t. \quad &\{\Delta t * (x_{k,low}^{2,st} * (R_{22} + R_{21} * x_{k,low}^{1,st}) + u_k)\} + x_{k,low}^{2,st} = x_{k+1,low}^{2,st} \\ &u_{min} \le u_k \le u_{max} \\ &x_{min} \le x_{k,low}^{i,st} \le x_{max} \quad \forall i = 1,2 \end{aligned}$$

(d) Store the solution $x_{k,low}$ with the respective $u_{k,low}$ and the cost-to-go $V_{k,low}$.

3. The upper bound of states x_1 and x_2 can be found using similar concepts as shown in the previous steps.

To implement our boundary conditions in dynamic programming, it is necessary to check if (36) fits, in which $x_{k,next}^{i}$ is all possible next states calculated from system dynamics. Next, we only consider the feasible states from $x_{k,next}^{i}$ that will optimize the problem. If the chosen state is on the boundary line, both the u^* and $V_k(x_k)$ will need to take the value from $u_{k,low/high}$ and $V_{k,low/high}$.

$$x_{k,low}^{i} \le x_{k,next}^{i} \le x_{k,hi}^{i} \quad \forall i = 1,2 \tag{36}$$

Results

The parameters in our dynamic programming problem is as listed: N = 200, Δt=0.1, x_{min}=0, x_{max}=200, u_{min}=-30, u_{max}=30, x_{init}=[10,5].

The system dynamics without optimal control is shown in Figure 7. The Figure 8 is the result for $38 \le x_N^1 \le 42$, $13 \le x_N^2 \le 17$. The simulation stops at $x_N^{*1} = 39$ and $x_N^{*2} = 14$. The Figure 9 is the result for $18 \le x_N^1 \le 22$, $18 \le x_N^2 \le 22$. The simulation stops at $x_N^{*1} = 19$ and $x_N^{*2} = 23$.

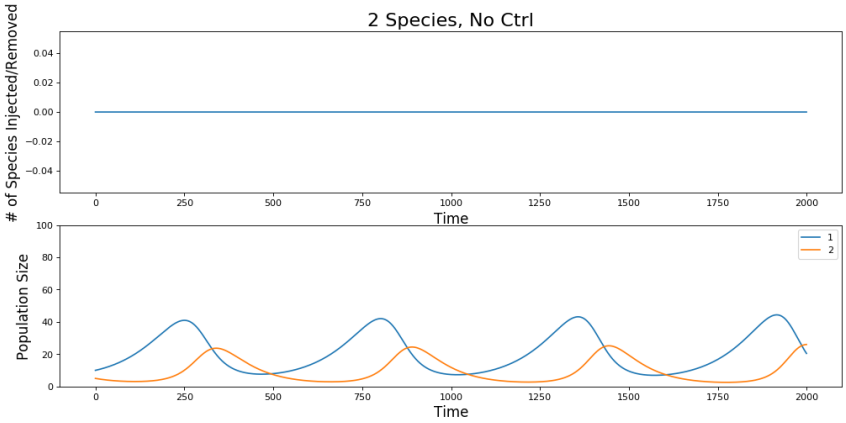


Figure 7: Original System Dynamics Without Control

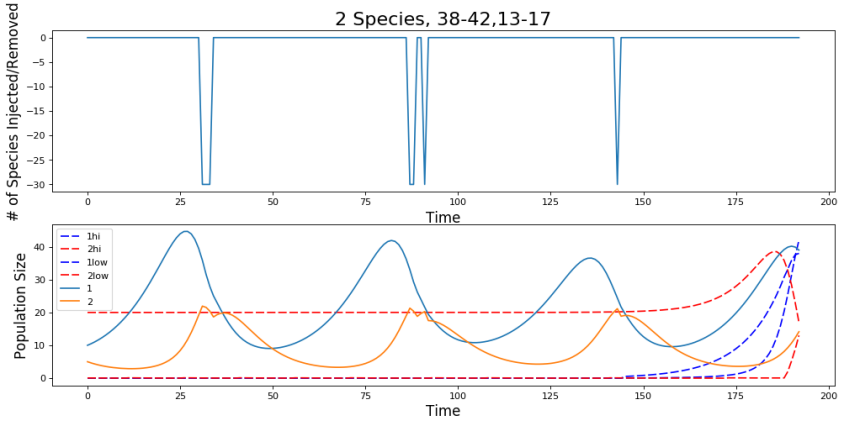


Figure 8: Simulation Result for $38 \leq x_N^1 \leq 42$, $13 \leq x_N^2 \leq 17$

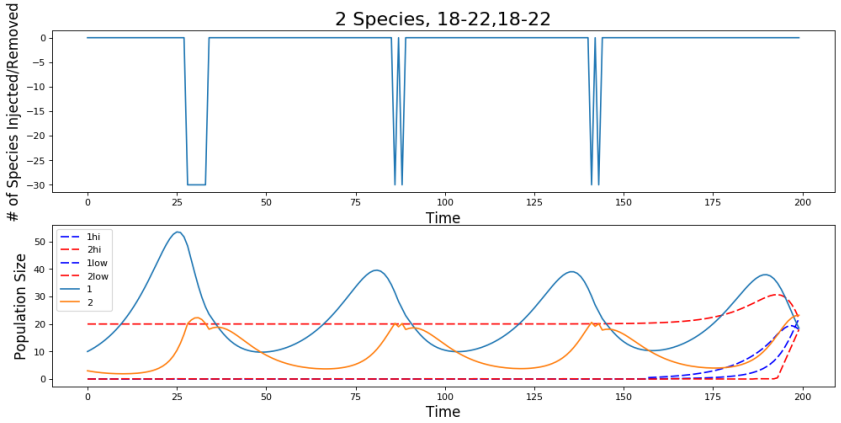


Figure 9: Simulation Result for $18 \leq x_N^1 \leq 22$, $18 \leq x_N^2 \leq 22$

Discussion

As shown in Figure 7, the two species modeled without the controls follow a stable trajectory, with the prey species x_1 peaking at around 42 members and the predator species x_2 peaking at around 25 members at the final time step. However, in using optimal control, the model aims to have a prey species between 38 to 42 members, and a predator population between 13 to 17 members. Figure 8 displays the simulation result for the dynamic programming model with optimal control; the control was applied to species x_2, in this case the predator species. The optimal control scenario involves removing predator members (the orange line in the population size graph) as the predator population peaks in the oscillatory period. The dashed blue and red lines represent the solution space upper and lower boundaries for species x_1 and x_2, respectively. The upper boundary line is the highest state $x_{i,high}$ at each of the time steps that allows us to obtain the final state (and vice versa for the lower boundary line) [6]. Figure 9 is a variation on the first optimal control model; in this scenario, both predator and prey populations must be between 18-22 members at the final time step. Similarly to the first optimal control problem, members of species x_2 are removed after each of the high points in the period.

Summary

One of the main takeaways from conducting a literature review and from our own experimentation with stability and equilibrium analysis is that populations change in an oscillatory pattern. A stable population, as modeled using the Lotka-Volterra model, will exhibit periodic behavior with bounded trajectories [8]. In addition, the equilibrium trajectories and the controllable inputs are highly sensitive to inter- and intra-species relationship parameters. Slight variations in population growth rates and species interactions could cause destabilization of the model. One limitation to our work is that the relationship parameters are not well understood, which is reflected in the literature. We also found that modeling with dynamic programming is exponentially more complicated to model as the number of species modeled increases. The Lotka-Volterra model is not easily transferred to models with multiple species. For example, a food chain with a odd number of species must include a logistic term, while models of an even number should not contain a logistic term [8].

Acknowledgments

We would like to extend our gratitude to Professor Scott Moura and Bertrand Travacca for their help throughout this process.

About the authors

Kyra Chang is a first-year Systems graduate student in Civil and Environmental Engineering at the University of California, Berkeley.

Tiffany Chang is a first-year Energy, Civil Infrastructure, and Climate graduate student in Civil and Environmental Engineering at the University of California, Berkeley.

Emily Farrar is a first-year Energy, Civil Infrastructure, and Climate graduate student in Civil and Environmental Engineering at the University of California, Berkeley.

Nate Tsang is a first-year Energy, Civil Infrastructure, and Climate graduate student in Civil and Environmental Engineering at the University of California, Berkeley.

References

[1] D. Gravel, F. Massol, and M. Leibold. "Stability and complexity in model meta-ecosystems". In: *Nature Communications* 7.12457 (2016). DOI: 10.1038/ncomms12457.

[2] L. Devireddy. "Extending the Lotka-Volterra Equation". In: (2016), pp. 2–10.

[3] E. Chauvet, J. Paullet, J. Previte, and Z. Walls. "A Lotka-Volterra three-species food chain". In: *Mathematics Magazine* 75.4 (2002), pp. 243–255. DOI: 10.2307/3219158.

[4] A. Lotka. "Analytical Note on Certain Rhythmic Relations in Organic Systems". In: *Proceedings of the National Academy of Sciences of the United States of America* 6.7 (1920), pp. 410–415. DOI: 10.1073/pnas.6.7.410.

[5] M. Rafikov and J. Balthazar. "Optimal Pest control problem in population dynamics". In: *Computational Applied Mathematics* 24.1 (2005), pp. 65–81. DOI: 10.1590/S0101-82052005000100004.

[6] O. Sundstrom, D. Ambuhl, and L. Guzzella. "On Implementation of Dynamic Programming for Optimal Control Problems with Final State Constraints". In: *Oil Gas Science and Technology - Rev. IFP* 65.1 (2010), pp. 91–102. DOI: 10.2516/ogst/2009020.

[7] P. Chesson. "Mechanisms of maintenance of species diversity". In: *Annual review of Ecology and Systematics* 31.1 (2000), pp. 343–366. DOI: 10.1146/annurev.ecolsys.31.1.343.

[8] S. Harris, J. Paullet, J. Previte, and J. Ranola. "A Lotka-Volterra Four Species Food Chain". In: *Mathematics Subject Classification* 92.01 (2005), pp. 1–10.

Blood Glucose Prediction

Mallika Bariya, Mohini Bariya

Abstract

In this project we develop a model for predicting blood glucose dynamics using auto-regressive handling of the input variables - dietary glucose intake and physical activity - and measured output (blood glucose levels as monitored by a continuous glucose meter). We build a training and test data set by carefully recording time-series data for the input variables and measuring blood glucose for an individual over the span of a week. The model we develop is a starting point for predicting blood glucose levels in pre-diabetic patients, potentially informing preventive therapies to stave off the onset of diabetes. Our predictive model could also be incorporated into insulin injection control loops for diabetic patients.

Introduction

Motivation and Background

Diabetes is a chronic metabolic disease in which the body is unable to support normal mechanisms for regulating blood glucose (BG) levels, either because it cannot endogenously produce the insulin needed to store and absorb sugars, or because it develops insulin resistance. As a result, sugars from food accumulate in the blood without a mechanism for removal by absorption into tissues and cells, causing BG levels to become dangerously elevated after eating. Diabetes can also result in sudden precipitous declines in blood glucose, and is in general marked by high BG uctuations that can be life-threatening. As one of the most abundant chronic diseases, with over 30 million Americans aicted and 1.5 million more being diagnosed each year [1], there is an urgent need to understand how blood glucose levels are impacted by daily activities and develop models for predicting blood glucose with the vision of enabling preventive therapies. While much eort has historically been dedicated to understanding the complex interplay of glucose and insulin in diabetic patients, it is also valuable to model glucose dynamics in healthy individuals with functioning glucose regulation systems. Non-diabetics do not rely on insulin injections or medication to stabilize changing glucose levels as endogenous insulin production is metered internally by a complex control system involving the pancreas, liver, and other organs. A similar system is at work for pre-diabetics who, though able to generate insulin, show developing insulin resistance that diminishes the eect of the internal control system. A predictive model for blood glucose levels in healthy individuals could thus be extrapolated to pre-diabetics with minimal restructuring, and could be used to inform lifestyle changes and local preventive care to stave

o diabetes. Almost 85 million Americans are pre-diabetics [1], but with a healthcare system that is focused on reactive as opposed to preventive treatment, there are few technological or medical solutions to aid in monitoring developing health states. A predictive blood glucose model that can capture the diminished response of the endogenous glucose control system would thus be hugely impactful for curtailing the rising number of diabetes diagnoses.

Focus of this Study

The focus of this study is to build a predictive model for the blood glucose evolution of a healthy, non-diabetic individual to explore the impact of influencing factors like physical activity and food intake. This model will be based on longitudinal data from a single subject to capture personalized model parameters.

Literature review

Many methods have been demonstrated for blood glucose prediction. Some studies, such as by Lehmann et al, favor a first-principles approach, using compartment models with simple differential equations to capture the interplay of food intake, physical activity, insulin and blood glucose [2]. Others use time-series methodologies, relying on historic measurements to predict future ones; for example, Eren-Oruklu et al use recursive linear models to estimate parameter values that can change to accommodate most recent data and any new disturbances [3]. Other studies use data-driven approaches and place less importance on the dynamics of individual variables. Sandham et al use 1 artificial neural networks on vast quantities of mined data to estimate weighting parameters for the different factors in uencing blood glucose [4]. Naumova et al use meta-learning, creating algorithms that learn which parameters are important before learning their actual values to improve portability of the model across patients and even applications [5]. Using simple compartment models is effective when the body produces no insulin and has no internal glucose control system. Then, injected insulin acts as a glucose regulator and is a measurable input, allowing key components of the governing differential equations to be known. In contrast, for healthy individuals it is impossible to directly measure the actions of the body's internal control mechanisms, complicating the use of simple differential equations. Using machine learning tools like neural networks poses challenges as well, requiring vast quantities of data to prevent overfitting while also sacrificing model explainability and the ease with which we can slightly alter the model structure to achieve better prediction-to-measurement coherence. As we are working with limited data, we instead choose a simple model structure built on an intuitive idea of how a glucose control system might work, while finding least squares estimates for parameter values. Our linear model is simple to apply and understand, allowing an instructive and potentially useful attempt at blood glucose prediction in healthy individuals.

Key contributions

While various complex physiological and data-driven models have been proposed for blood glucose prediction, none achieve very successful results. We seek to create a simple, intuitive model based on known influencing factors to determine each of their impact on blood glucose

evolution. While we cannot hope to achieve very accurate predictions with this approach, our contribution is mainly towards understanding how food intake and activity individually impact blood glucoes.

1 Data Collection

In our simplified model, we assume that blood glucose levels can be accurately predicted based on food intake and activity. We collect food intake, activity, and blood glucose measurements over 6 days in March. These measurements comprise the time series dataset on which our models are trained and tested. In the following section, we describe how each type of measurement is made, and how the different measurements are converted to time series on the same discrete time grid (i.e. with identical sampling periods). In addition, for each data source, we describe potential sources of error in the measurement process and data pre-processing that could contribute to error in our final blood glucose predictions.

The sampling period of all the time series, denoted dT, was chosen to be 5 minutes, as this was the resolution of the blood glucose data. For diabetic applications, a 5 minute prediction time-step is reasonable for the purposes of glucose prediction as we would expect the blood glucose to remain reasonably stable over this time period. However, if this is not the case, a a shorter time step is desired, our models could be trained on higher resolution data, or our 5 minute resolution predictions could be interpolated to a finer time scale. For reference, with $dT = 5$, 6 days of data contain a total of 1728 data points per measurement time series.

1.1 Activity Data

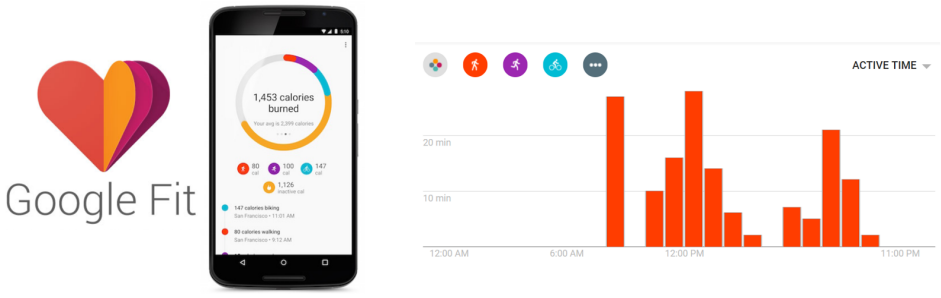


Figure 1: The Google Fit app estimates various activity metrics based on smart phone sensor measurements. Estimates are streamed to a web interface.

We use Google Fit, a cellphone application, to record activity [6]. Google Fit estimates steps walked and calories burned using GPS, gyroscope, and accelerometer measurements from a smart phone. For our project, we choose to use Google Fit's estimates of calories burned, rather than steps walked, as representative of the subject's activity. We believe calories burned should be more directly linked to the removal of glucose from the blood as

sugars are used to fuel cell activity. Since the subject did not do any exercise during the test period, and especially no stationary exercise such as on an exercise bike or treadmill, the activity data predominantly tracks walking. Figure shows the Google fit interface. Google fit records estimates every 1 hour, producing a time series $c_h[k]$. We interpolate these onto a $dT = 5$ min. time grid by assuming that calories are burned at a fixed rate every hour. This produces the time series $c[k]$ which is related to $c_h[k]$ as follows:

$$c[k] = \frac{1}{12} c_h \left[\left\lfloor \frac{k}{12} \right\rfloor \right]$$

Potential errors introduced by the activity data would largely consist of inaccurate estimates of calories burned. Google Fit uses its own algorithm to convert sensor measurements to calories burned, taking into account the self-reported height and weight of the user. These algorithms are unlikely to be perfect. However, we did ballpark checks that confirm the caloric estimates are reasonable.

1.2 Food Data

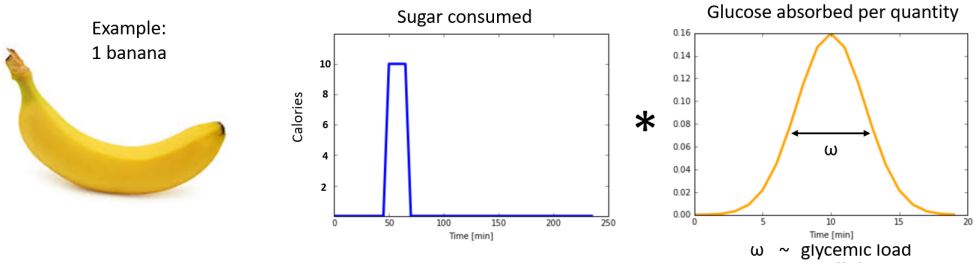


Figure 2: An example of converting qualitative food data to a quantitative time series for one banana.

In our simplified model, food is the input source of blood glucose. To create a time series of food data, we start by recording the the type and quantity of every item of food consumed, as well as the time period over which it was consumed. Some sample entries from our records are included in Table 1.

Table 1: Sample food intake logs

Day	Time	Item
3/15/18	3:45pm-3:48pm	half a doughnut with chocolate glaze
3/15/18	5:45pm-5:48pm	1 banana

The qualitative food data must be converted into a quantitative time series. The impact of a certain food on blood glucose levels is related to two important properties of the food item. These are the *caloric content* and the *glycemic load* [7]. The significance of the caloric content is intuitive as it indicates how much sugar is contained in the food, and therefore

should be directly related to the quantity of sugar entering the blood stream. The glycemic load reflects how *quickly* the sugars from a certain food can be released into the blood. This depends on such aspects as the structure of the food, the amount of fiber it contains, etc [8]. Two foods may have the same caloric content, but if one has a larger glycemic load, it will lead to a more rapid increase in blood sugar. Since they depend on such a variety of factors, glycemic load values are determined through experimentation, with white bread setting the standard with a glycemic load of 100.

We use both caloric content and glycemic load to generate our quantitative food intake time series. To begin, we convert the quantitative food data into a time series of $rect()$ functions. Each $rect()$ corresponds to one food item. The width of the $rect()$ is equal to the duration over which the food was consumed, while the height is chosen so that the area under the $rect()$ is equal to the caloric content of the food item. To account for the glycemic load, we convolve each $rect()$ with a Gaussian, where the width of the Gaussian is related to the glycemic load. An example for one banana is shown in Figure 2.

1.3 Blood Glucose Data

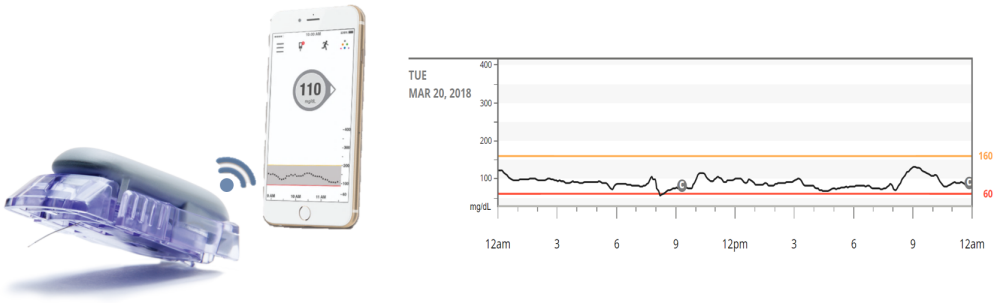


Figure 3: The Dexcom Blood Glucose Sensor and online measurement reporting interface. Note the fine needle attached to the sensor that penetrates the skin.

To measure blood glucose values over our test period, we use Dexcom, a continuous glucose monitoring [9][10]. As shown in Fig. 3, the Dexcom device consists of a small data conversion and transmission unit that sits above the skin, and a fine sensing needle that penetrates the skin to access interstitial fluid (IF). This sensor makes periodic IF glucose measurements every 5 minutes. IF glucose levels correlate one-to-one with blood, and time delays between the two fluid compartments are compensated for in Dexcom's signal processing. Overall, the Dexcom sensor effectively measures blood glucose levels and the generated data can be collected into a time series as shown in the right plot of Fig. 3. In this plot, the black line shows the Dexcom sensor reading; the red and yellow lines depict cut-offs for dangerously high or low glucose levels. The grey circles labeled with 'C' set the times at which the Dexcom sensor was re-calibrated via comparison with a fingerstick blood glucose test. Calibration is necessary to compensate for drifting in the Dexcom signal that could lead to underestimation or overestimation of blood glucose values. If there is a large discrepancy between the Dexcom reading and the fingerstick control measurement at the calibration

point, the sensor is re-set to match the control. In our data, our sensor did not have much drift and thus calibration points did not create significant discontinuities in the time series (apart from day 1, when the sensor is first inserted and shows particularly noisy readings as it equilibriates with its new *in vivo* surroundsings). This is important as it ensures that the blood glucose data generated by Dexcom is accurate throughout the measurement interval.

2 Predictive Models

We train and test several different models for blood glucose prediction. We develop the models incrementally to better understand how strongly each input variable impacts blood glucose levels. In the following we begin by detailing the performance metrics used to evaluate our models. We then describe each model and detail the results of training and testing each model.

2.1 Performance Metrics

- Mean Squared Error. The mean squared error (MSE) is the classic metric for quantifying the quality of prediction. Given a prediction time series $\hat{y}[n]$ and the true time series $y[n]$, both of length N, the MSE is given by:

$$MSE(\hat{y}) = \frac{1}{N}\sum_{n=1}^{N}(\hat{y}[n] - y[n])^2$$

  However, for the purposes of blood glucose prediction, the MSE can be difficult to interpret. Given our highly simplified system in which only food and activity affect glucose levels, if we are able to reliably predict when glucose rises and falls, the model should be considered quite successful. This motivates the use of another metric: the correlation coefficient.

- Correlation Coefficient. The correlation coefficient measures how correlated two random variables are, in other words how closely they rise and fall together. For our time series, $\hat{y}[n]$ and $y[n]$, the correlation coefficient is computed as follows:

$$\rho(\hat{y}) = \frac{\sum_{n=1}^{N}(\hat{y}[n] - \mu_{\hat{y}})(y[n] - \mu_y)}{\sqrt{\left[\sum_{n=1}^{N}(\hat{y}[n] - \mu_{\hat{y}})^2\right]\left[\sum_{n=1}^{N}(y[n] - \mu_y)^2\right]}}$$

$$\mu_y = \frac{1}{N}\sum_{n=1}^{N} y[n];\ \mu_{\hat{y}} = \frac{1}{N}\sum_{n=1}^{N}\hat{y}[n]$$

  The correlation coefficient is a useful performance metric for this project because it indicates how well a model is doing at predicting the rises and drops in blood glucose. Given our highly simplified system, a strong correlation between the true and forecasted blood glucose levels would represent a success. As a rough, rule-of-thumb, $\rho(\hat{y}) \geq 0.7$ would indicate that $\hat{y}$ is a good prediction.

2.2 Autoregressive Model

The autoregressive (AR) model is a simple linear model which uses current and past blood glucose measurements to predict future blood glucose levels. Given blood glucose measurements $g[n]$, the model takes the following form:

$$\hat{g}[n+k] = \sum_{i=0}^{l_g} \alpha_i g[n-i]$$

$\hat{g}[n+k]$ is our k step ahead prediction, l_g is the model order, or the number of past measurement points we use to make our prediction, and $\alpha_i, i = 0, ..., l_g$ are the model coefficients. We choose $l_g = 12$, corresponding to the use of 1 hour of previous measurements. We fit the model by solving the following ordinary least squares optimization problem.

$$\alpha^* = arg\min_{\alpha} ||G_{n+k} - G_n\alpha||_2^2$$
$$\alpha^* = (G_n^T G_n)^{-1} G_n^T G_{n+k}$$

where G_n and G_{n+k} are data matrices of glucose measurements constructed appropriately from the $g[n]$ time series. α is a vector of the model coefficients.

For $k = 1$, the model performs extremely well. The results are shown in Figure 4. This tells us that in a healthy subject, the blood glucose evolves relatively smoothly. Therefore, if the current and a few past values of the blood glucose are known, the blood glucose 5 minutes (k=1) into the future can be predicted reliably.

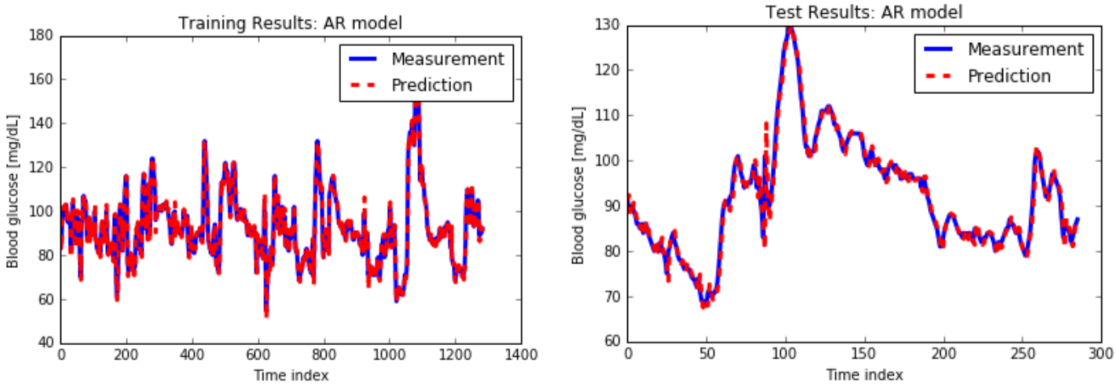


Figure 4: AR model results on training data (left) and test data (right)

As the prediction horizon (ie k) increases, the prediction quality falls steadily. As shown in Figure 5, at some point, there is a leveling off in the MSE (and correlation), indicating the time horizon at which the current blood glucose values are essentially useless for predicting the k-step ahead future blood glucose values.

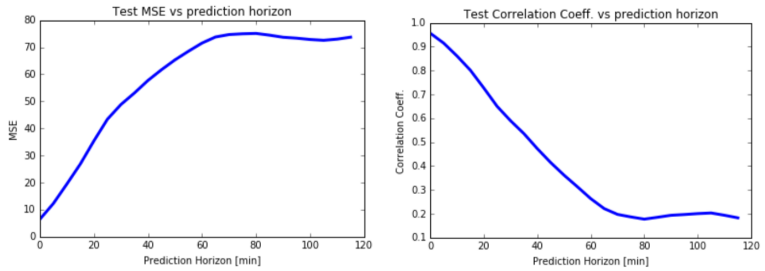


Figure 5: MSE (left) and Correlation Coefficient (right) versus the prediction horizon.

The AR model is accurate but not useful. Indeed, we could say that this model makes sense from a pure data perspective but not from a physiological perspective. It does not account for the external factors, such as food consumption and activity, that we *know* impact blood glucose. Such a model would be even less useful for diabetic subjects, whose blood glucose level is less regulated by the body and more prone to sudden, large changes due to behaviour (such as eating or running).

2.3 Affine Food Intake Based Glucose Prediction

To focus on the effect of external factors (food consumption and activity) on future glucose values, we train a model to predict one step (5 minutes) ahead glucose levels from food intake data *only*. Given food intake values $f[n]$, the model takes the following form:

$$\hat{g}[n+1] = \sum_{i=0}^{l_f} \gamma_i f[n-i] + b$$

We allow for a bias term b as the level of the glucose measurements $g[n]$ and food intake time series $f[n]$ differs. We fix $l_f = 24$, corresponding to two hours of previous food intake measurements. Again, we fit the model by solving the following OLS optimization problem.

$$\gamma^* = arg\min_{\gamma} ||G_{n+1} - F_n\gamma||_2^2$$
$$\gamma^* = (F_n^T F_n)^{-1} F_n^T G_{n+1}$$

Food intake is sporadic during the day. In fact, in our last two days of test data, the subject only ate once. During the other time periods, $f[n] = 0$, and apart from a constant b, the given model has no ability to predict the blood glucose level. Therefore, for the purposes of this model, we restrict our data set to regions during which there is food consumption. These time periods are shown in Figure 6. The blue periods correspond to training data, while the yellow correspond to test data.

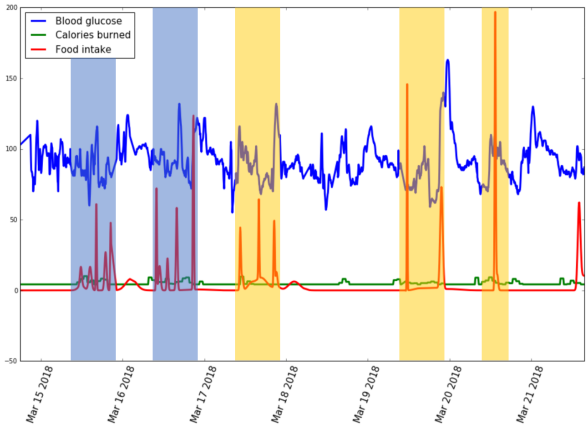


Figure 6: Data periods used for training (blue) and testing (yellow) the affine food intake glucose prediction model.

The results of the food intake model are shown in Figure 7. We can see that there is a large range in the test MSE/correlation. The correlation (which is a more intuitive metric here) ranges from -0.01 to 0.63.

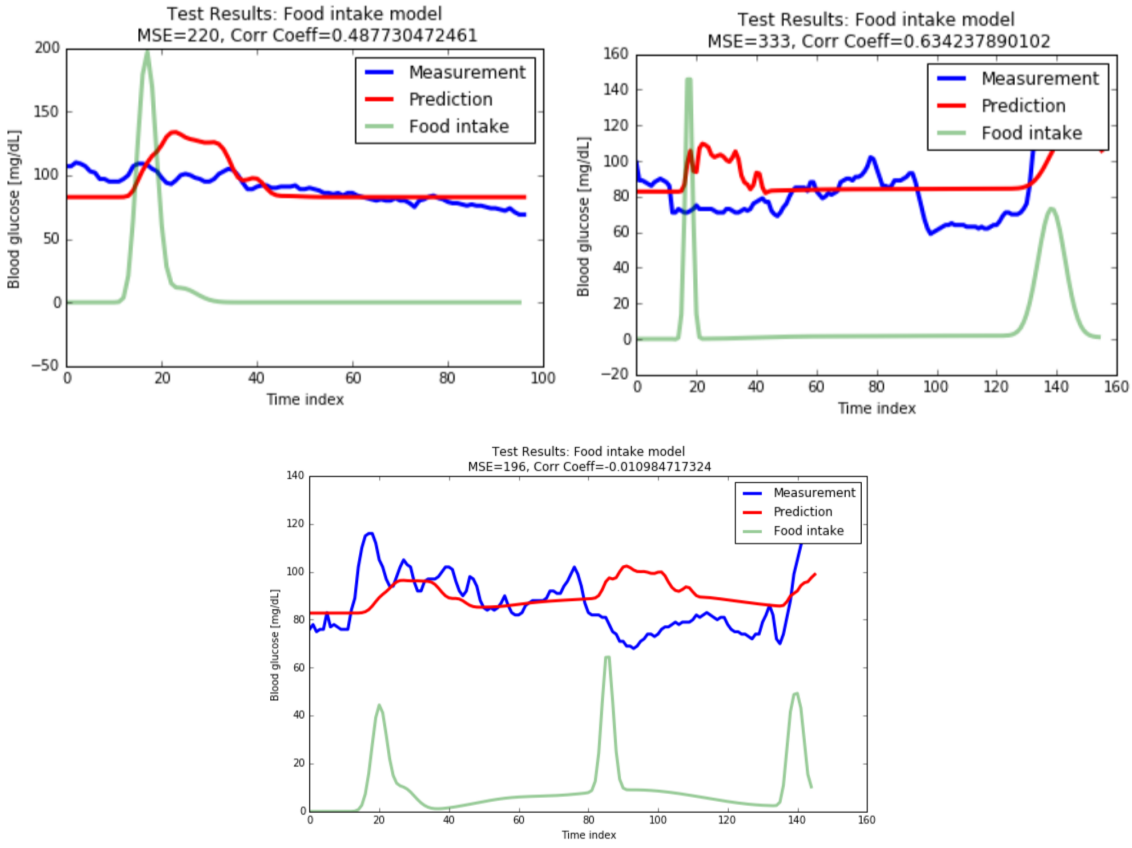


Figure 7: Test results for affine food intake prediction model

Glycemic load errors would severely deteriorate the performance of this model. This

might be the reason we see certain periods where the blood glucose predictions and measurements seem to be *anti*-correlated. If the glycemic load is inaccurate, the blood glucose rise due to food consumption could occur later than the model predicts. This is visually expressed in Figure 8.

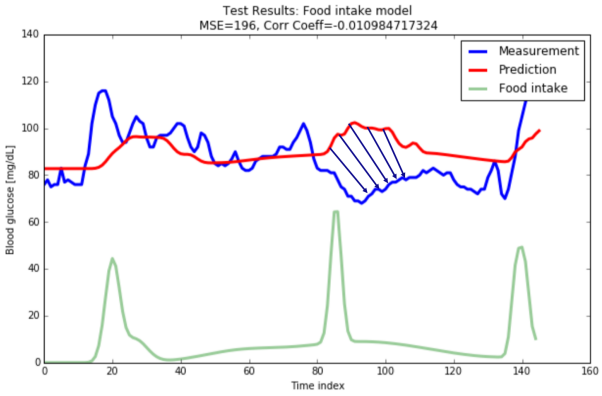


Figure 8: Potential effect of glycemic load errors on model predictions: Glucose rises are predicted to occur before they actually do, as indicated by the dark arrows.

2.4 Affine Food Intake and Activity Based Glucose Prediction

We extend the affine food intake model to incorporate activity. We again train a one step ahead glucose predictor, now using *both* food and activity time series data. Given food intake time series $f[n]$ and activity (estimated calories burned) time series $a[n]$, the model takes the following form:

$$\hat{g}[n+1] = \sum_{i=0}^{l_f} \gamma_i f[n-i] + \sum_{i=0}^{l_a} \beta_i a[n-i] + b$$

We fit the model by solving the OLS optimization, fixing $l_f = 24$, $l_a = 24$.

$$\begin{bmatrix} \gamma \\ \beta \end{bmatrix}^* = arg\min_{\begin{bmatrix} \gamma \\ \beta \end{bmatrix}} \left|\left| G_{n+1} - \begin{bmatrix} F_n & A_n \end{bmatrix} \begin{bmatrix} \gamma \\ \beta \end{bmatrix} \right|\right|_2^2$$

The results for the model are shown in Figure 9. Comparing with 8, we can see that the test performance actually deteriorates compared to the affine food intake.

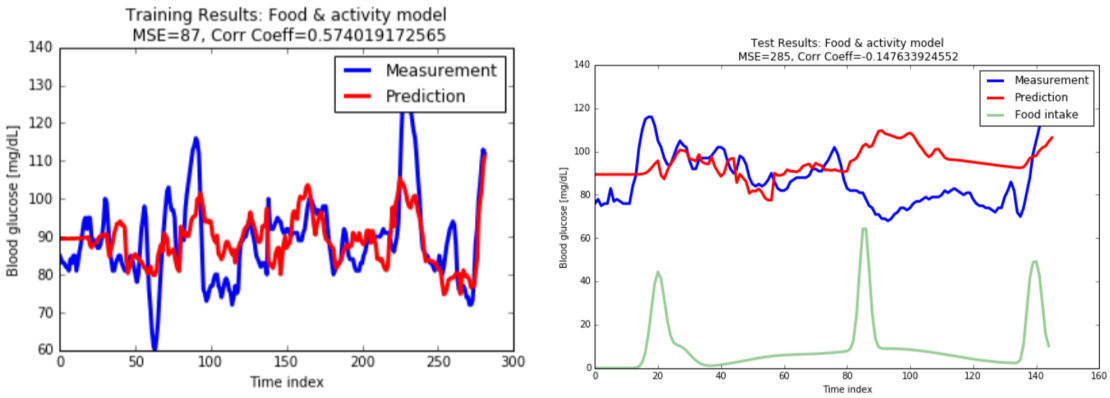


Figure 9: Training (left) and test (right) results for the affine food intake and activity based glucose prediction.

To understand this performance deterioration, we consider the fitted coefficients, shown in Figure 10. The β values, corresponding to the activity time series, do not make physical sense. The magnitude of the coefficients increases with i, and the sign of the coefficients oscillates. This is symptomatic of overfitting.

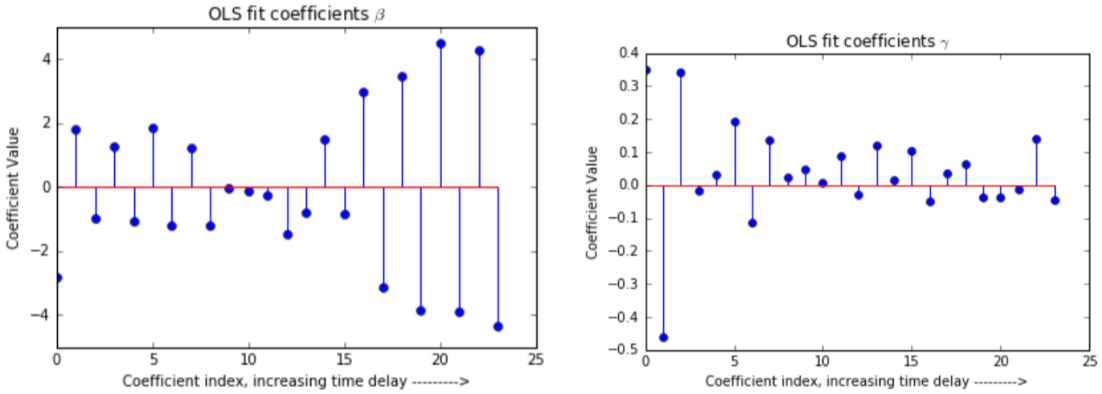


Figure 10: The OLS fitted coefficients for the affine food intake and activity based glucose prediction model. β (left) corresponds to the activity time series while γ (right) corresponds to the food time series.

To mitigate overfitting, we add regularization, specifically penalizing the γ coefficient magnitudes *only*. This is a generalized Ridge regression optimization, which can be expressed as:

$$\begin{bmatrix}\gamma\\ \beta\end{bmatrix}^* = arg\min_{\begin{bmatrix}\gamma\\ \beta\end{bmatrix}} \left|\left| G_{n+1} - \begin{bmatrix}F_n & A_n\end{bmatrix} \begin{bmatrix}\gamma\\ \beta\end{bmatrix} \right|\right|_2^2 + \begin{bmatrix}\gamma & \beta\end{bmatrix} \Sigma \begin{bmatrix}\gamma\\ \beta\end{bmatrix}$$

To penalize the γ coefficients only, we set the Σ matrix as follows:

$$\Sigma = \begin{bmatrix} \lambda & 0 & \dots & 0 & 0 \\ 0 & \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

Σ contains all zeros except in the diagonal elements corresponding to the γ coefficients. Note we penalize all γ_i equally with weight λ. The results for $\lambda = 3000$ are shown in Figures 11-12. We can see that the test MSE reduces with regularization. The β coefficients are all negative, which makes more physical sense since activity corresponds to removal of blood glucose. However, the MSE and correlation are worse than for the purely food intake based model, suggesting that either activity does not effect blood glucose in a linear and predictable way, or our training data is insufficient to establish the relationship.

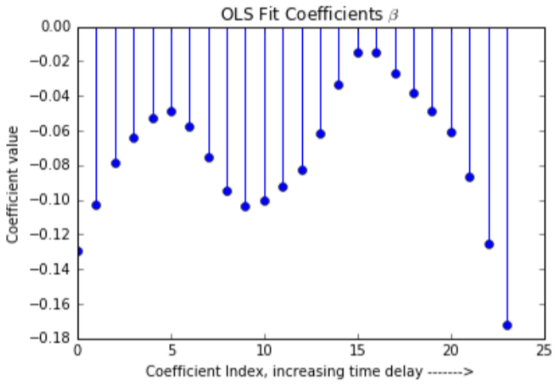


Figure 11: The regularized fitted β coefficients for the affine food intake and activity based glucose prediction model. The regularization parameter was $\lambda = 3000$

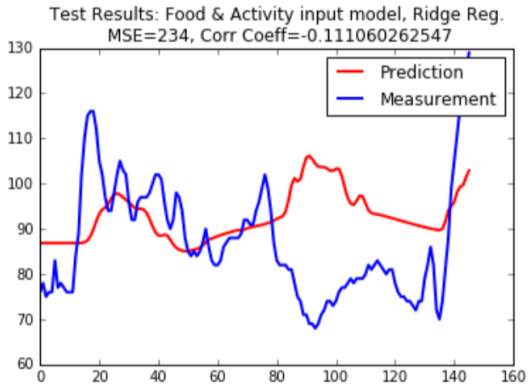


Figure 12: Test results for the affine food intake and activity based glucose prediction with regularization.

2.5 Neural Network Glucose Prediction from Food and Activity

The body's regulation of blood glucose through its internal control mechanisms is likely to be highly nonlinear. This limits the success of linear models like those we tested in the previous sections. Our final model is a neural network with $ReLu$ activations and one hidden layer with 4 neurons. A diagram of the network is shown in Figure 13. The neural network aims to predict the blood glucose from the food and activity time series.

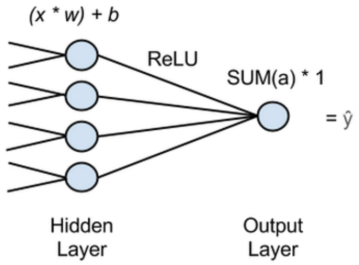


Figure 13: Diagram of the neural network model used for blood glucose prediction from food and activity data. Courtesy of deeplearning4j.org

The results of the model are shown in Figure 14. We can see that the neural network manages to achieve a better correlation than the affine food intake model on the training data. The performance on the test data is only slightly better than the affine food model. It is interesting that both the neural network and linear model achieve a similar prediction on the test data. This suggests that the poor prediction is not due to model issues, but due to changes in the relationship between blood glucose levels and activity and food intake between the training and test periods.

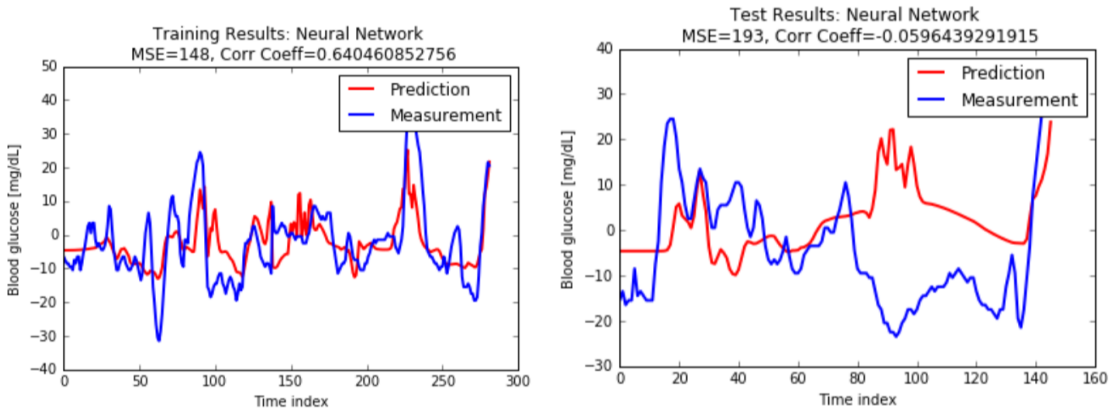


Figure 14: Training (left) and test (right) results for the Neural Network predictor.

Conclusion

In this project, we developed, trained, and tested several models for predicting blood glucose from food intake and activity data. Our success in predicting blood glucose was

limited by a paucity of training and testing data and the simplicity of our models. We made the assumption that blood glucose was predominantly affected by food intake and activity. However, in a healthy, non-diabetic subject, the body has a complex control system at work which regulates blood glucose levels. This system produces changes in blood glucose that are difficult to predict from food and activity alone.

We found that in a healthy subject, an AR model can be used to predict future blood glucose levels from current and past measurements, without the use of food intake or activity measurements. This result is useful in a control system framework where continuous glucose measurements are used to determine insulin delivery.

We also found that in our dataset, blood glucose prediction from food intake alone performed better than incorporating both food intake and activity. This suggests that food intake has a more direct impact on blood glucose, while activity has more complex effects that are not easy to learn from our limited dataset.

Finally, we found that simple linear models performed nearly as well as nonlinear neural networks on our limited dataset.

References

[1] "Diabetes statistics," *http://www.diabetes.org/diabetes-basics/statistics/*.

[2] E. D. Lehmann and T. Deutsch, "A physiological model of glucose-insulin interaction in type 1 diabetes mellitus," *Journal of biomedical engineering*, vol. 14, no. 3, pp. 235–242, 1992.

[3] M. e. a. Eren-Oruklu, "Estimation of future glucose concentrations with subject-specific recursive linear models," *Diabetes technology therapeutics*, vol. 11, no. 4, pp. 243–253, 2009.

[4] W. A. e. a. Sandham, "Neural network and neuro-fuzzy systems for improving diabetes therapy," *Engineering in Medicine and Biology Society. Proceedings of the 20th Annual International Conference of the IEEE*, vol. 3, 1998.

[5] S. V. P. Naumova, Valeriya and S. Sivananthan, "A meta-learning approach to the regularized learning—case study: blood glucose prediction," *Neural Networks*, vol. 33, pp. 181–193, 2012.

[6] "Google fit," *https://www.google.com/fit/*.

[7] "Glycemic index," *https://en.wikipedia.org/wiki/Glycemic index*.

[8] "United states department of agriculture agricultural research service usda food composition databases," *https://ndb.nal.usda.gov/ndb/search/list*.

[9] "Dexcom continuous glucose monitor," *https://www.dexcom.com/*.

[10] *https://mspoweruser.com/fitbits-ionic-smartwatch-getting-support-continuous-blood-glucose-monitoring/*.

About the authors

Mallika Bariya is a PhD student in Material Science at Berkeley. Her research is focused on developing noninvasive sensors for personalized health monitoring.

Mohini Bariya is a PhD student in Electrical Engineering at Berkeley. Her research is on using sensor data to improve situational awareness and enable integration of renwables in the distribution grid.

Optimization of Rainwater Harvesting Systems for Single-Family Residential Buildings in California

Ahmad Bin Thaneya, Fiona Greer, Minghui Zhang

I. Abstract: It is important to consider water used in buildings in California due to concerns of water scarcity, as well as a growing understanding of the water-energy nexus and its relationship with the climate. Rainwater harvesting systems (RHS) are one example of onsite water reuse systems proposed for meeting non-potable water demands in buildings. There exist several questions regarding the efficacy of RHS in meeting energy, greenhouse gas, and cost objectives. This research project will compare the use of RHS in residential buildings with two California cities, Oakland and San Diego, and simulate RHS performance in meeting established non-potable water needs through constraints of rainwater supply, transmission energy, greenhouse gas emissions, and life-cycle costs. A multiobjective optimization problem was solved in each city to determine the optimal operation of RHS in order to minimize cost and energy use, and maximize the portion of demand that is supplied by rainwater. Through dynamic programming and economic cost-benefit analysis, we found that RHS in California are not justifiable in their current cost and emissions considerations unless a RHS subsidy is provided or costs of water savings and emissions are adjusted.

II. Introduction

a. Motivation and Background:

Water is a vital resource used in buildings. It is important to consider alternative sources of water in order to meet greenhouse gas reduction goals and to sustain society's water demand. Total urban water usage in California averages around 9.1 million acre-feet a year, with 4.2 million acre-feet used in urban outdoor applications (Pacific Institute, 2014). Residential use accounts for 64% of the total urban water use, while commercial use accounts for 23% (Pacific Institute, 2014). Onsite water reuse systems, such as rainwater harvesting systems (RHS), have gained popularity in recent years as a strategy that could address concerns about water scarcity and GHG emissions associated with the treatment, transport, and heating of water. RHS, which capture, store, and use collected water from precipitation events, are a water conservation technology that have the potential to address a building's water needs, such as its toilet and urinal flushing requirements.

The impetus to study RHS is bolstered by the fact that two of the members in this group are currently working on a project for the California Air Resources Board (CARB) that studies the feasibility of net zero carbon buildings. One of the net zero carbon building strategies being analyzed in the CARB project is onsite RHSs. There are limitations in the CARB project that prevent detailed analysis of whether RHS are a cost-effective and energy saving strategy. This project aims to address whether RHS are a practical strategy.

b. Relevant Literature:

Recent interest in RHS adoption stems from the benefits that these systems can provide building owners such as: (1) limiting costs to only system installation and operation; (2) allowing for water sourcing without heavy reliance on complex distribution systems; (3) augmenting or replacing utility sourced water for non-potable water needs, including toilet flushing; and (4) reducing customer utility bills (Texas Water Development Board, 2005). Researchers and policy makers are also interested in the potential societal benefits of RHS. These benefits include: (1) mitigating rising water demand through decentralized sourcing; (2) conserving of water and energy; (3) reducing stormwater runoff and associated pollutant loads; (4) replacing depleted groundwater supplies; and (5) aiding electricity utilities by reducing summer demand peaks (EPA, 2015). The practicality of RHS adoption is contingent on

rainfall characteristics and RHS storage capacity (Texas Water Development Board, 2005). Table 1 lists the main components of a typical RHS (Texas Water Development Board, 2005).

Table 1. Typical RHS Components.

Catchment Surface
Gutters
Downspouts
Debris and Dust Removing Mechanisms
Storage Tank (also known as a Cistern)
Water Delivery System
Treatment / Disinfection Equipment

Many home appliances require 20-30 psi of pressure for proper operation, making gravity flow of water impractical. Therefore, water pumps are typically used to meet pressure requirements. The majority of energy requirements for RHS is due to water pumping (Chiu et al., 2009). Relevant factors include lift, water flow, pump efficiency, transmission efficiency, and friction loss of pipes.

Researchers have found that RHS cannot economically compete with local utilities in supplying potable water; RHS might be competitive for non-potable uses (Texas Water Development Board, 2005). A perceived lack of economic benefit is often cited as an impediment for widespread RHS adoption, as upfront capital costs of RHS installation and annual maintenance fees make it difficult to compete against low-cost and highly subsidized municipal water sources. However, many studies neglect full life-cycle costing within their benefit-cost analyses. There is a growing need to incorporate social benefits of water conservation and stormwater capture in addition to financial costing (EPA, 2015). A full life-cycle costing analysis should include: RHS capital cost, maintenance costs, water conservation benefit, stormwater management benefit, possible energy conservation and environmental benefits, and the displacement of the need for water infrastructure development to meet growing water demand (EPA, 2015) (Farreny et al., 2011) (Sample and Liu, 2014).

Economic feasibility analyses of RHS incorporate several performance criteria in order to gauge adoption thresholds. Chiu et al. (2009) employed a benefit-cost (B/C) ratio to study the efficacy of RHS adoption in Taiwan. The study found that when water and energy-saving benefits are combined, the B/C ratio exceeds the threshold for adoption. If water and energy-saving benefits are analyzed separately, the B/C ratios are lower than one, signifying an economically unattractive system. Hajani and Rahman (2014) came to a similar conclusion using a reliability and life-cycle cost analysis for implementing RHS in rural areas of Australia. Farreny et al. (2011) used a life-cycle costing framework to conduct a net present value (NPV) study on implementing RHS infrastructure within dense Mediterranean neighborhoods. They found that capital costs, operational costs, and maintenance expenses outweighed the financial benefit of not having to use utility-sourced water. This study did not place any value on the benefits associated with water conservation, stormwater capture, and displacement of the need for growing water supply infrastructure. Their study found that RHS adoption is not cost-efficient, which is a common trend within literature under current water pricing (Rahman et al., 2010).

Jenkins and Pearson (1979) studied RHS feasibility in California. They concluded that rainwater systems in California are feasible. RHS could only partially supply domestic demand however, as the high cost of the large tanks required to sustain demand during the dry season would make it too costly a system. However, the authors did not define the difference between the costs, energy, and carbon emissions of RHS compared to utilities. This is a gap in the literature that this study hopes to fill.

The majority of the literature assesses the performance of RHS using a behavioral model. The behavioral model is a discrete-time mass balance equation of the water storage in the RHS tanks under inputs of rainfall, demand, and one of two operating conditions. The operating conditions, yield after spillage (YAS) and yield before spillage (YBS), are two approximations of the timing between water demand and supply. In reality, these may occur continuously and at the same time over the course of a

day. In YAS, the demand is assumed to occur before any rain replenishes the tank on that day; the opposite is true of YBS (Jenkins and Pearson, 1979).

The most common method for assessing the optimal system design is through sensitivity analysis. Most studies attempt to determine optimum sizing of RHS using a variety of system performance indices, such as water-saving efficiency (WSE) and overflow ratio (OR) (Palla et al, 2011), (Mun and Han, 2012). WSE is the ratio between volume of rainwater supplied and water demand for a selected time interval. OR is the ratio between the volume of rainwater exceeding the system capacity and the inflow to the RHS for a selected time interval. The water-saving efficiency will appear in our multi-objective optimization function. A notable pattern in RHS literature is a marked absence of optimization of RHS operation with respect to cost, energy use and carbon emissions. Moreover, when performing benefit cost analyses, researchers and policymakers have also failed to include the non-monetary benefits of RHS, such as conservation and stormwater management benefits. This gap in the literature presents an opportunity for us to apply our new optimization knowledge to a novel research question.

c. Focus of this Study:

The main objective of this study to analyze the ability of RHS to meet economic targets, energy efficiency goals, and greenhouse gas emission reduction requirements, while being a reliable source for meeting non-potable water demand. Ultimately, the goal is to assess the conflicting trade-off between water conservation and emission profiles of RHS.

III. Technical Description

Data Sourcing and Data Prepping:

Rainfall data:

Daily rainfall data for each city were obtained from the National Oceanic and Atmospheric Administration's Climate Data Online portal. If multiple rain gauge data were available in the same city, the gauge closest to the city center was chosen to obtain the best representation of the precipitation experienced by most buildings in the city. Before we are able to optimize RHS operation, we must determine an appropriate storage tank size. We followed a popular sizing technique, which ensures a storage capacity sufficient to store water collected during rainy times to last through dry spells (Texas Water Development Board, 2005). Given the wide range of dry spell lengths, we took the maximum dry spell from each calendar year between 1970 and 2017, and took the average of these maximum annual dry spells to size the tank for each city and building type. The average dry spell lengths and standard deviations are as follows: 93 +/- 36 days for Oakland and 98 +/- 41 for San Diego. This information helped us design a tank size that has the capacity to supply demand during an average water year. The size of the tank is equal to the product of daily demand and the length of the dry period.

Building Parameters Description:

Building type characteristics, including catchment area and non-potable daily water demand, are provided in Table 2. The data source for the catchment areas comes from a 2012 Department of Energy report from which all building characteristics (e.g. square footage, catchment area) are derived (Arup, 2012). The non-potable water demand, which is comprised of toilet and urinal water demand, is based on current research being conducted for the California Air Resources Board (CARB).

Table 2. Building Type Characteristics.

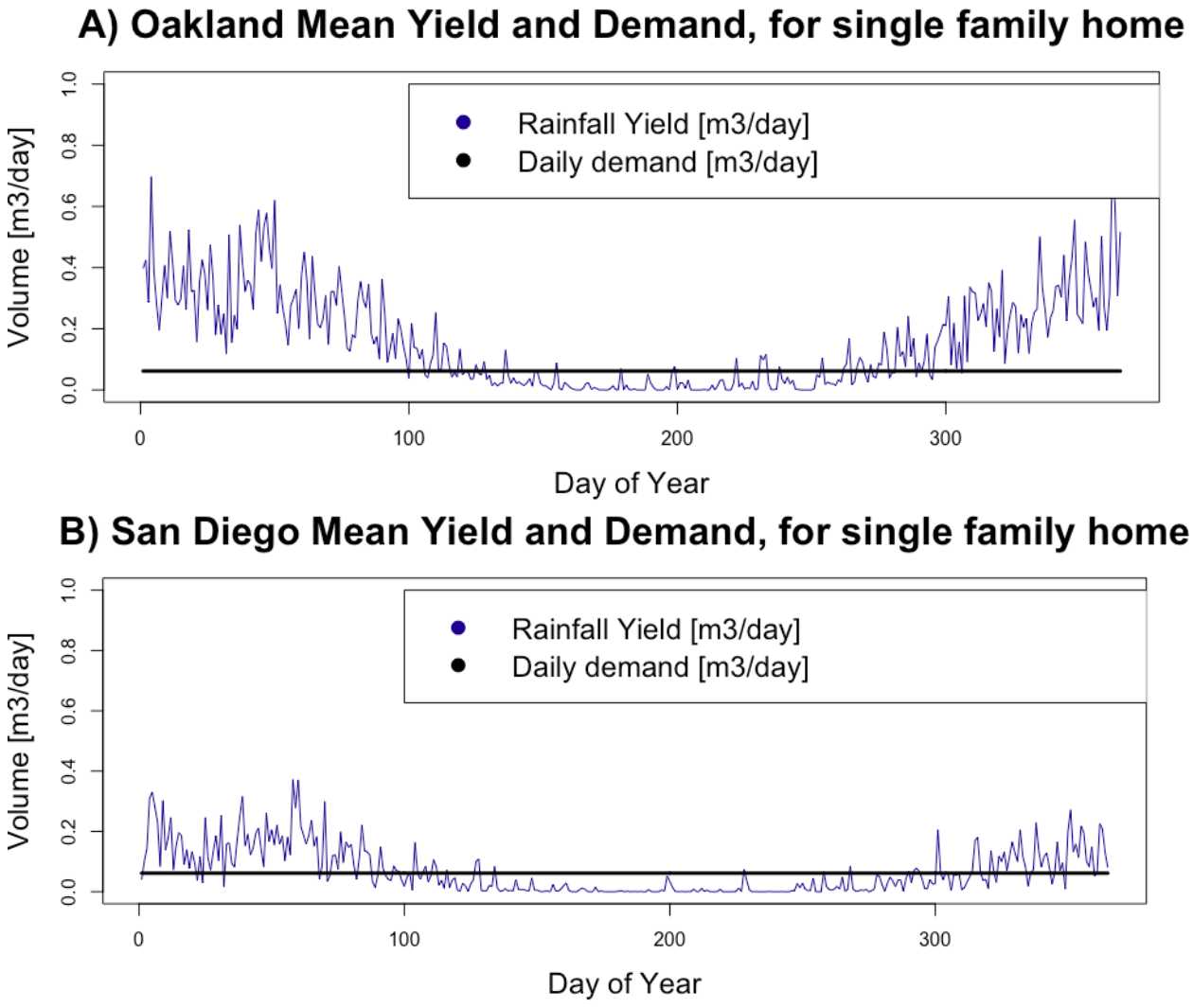
Building Type	Catchment Area [m^2]	Non-potable Water Demand (m^3/day)
Single Family	97	0.0616

The daily potential harvestable volume, or daily RHS yield, is calculated using Equation 1. Sathe et al. assume a collection efficiency of eighty percent (Sathe et al., 2012):

$$RHS\ Yield\ (m^3) = Daily\ Precipitation\ (m) \times Catchment\ Area\ (m^2) \quad (1)$$

Seasonality and variability of rainfall yield:

Figure 1 shows mean daily yield, demand and standard deviation for a representative city, Oakland, from 1970 to 2017. These graphs show a dry season in the summer months, and large standard deviations in daily yield indicate a need to capture the full range of rainfall dynamics. In Oakland, the home's demand is lower than rainwater yield for most of the year; in San Diego, the opposite is true.



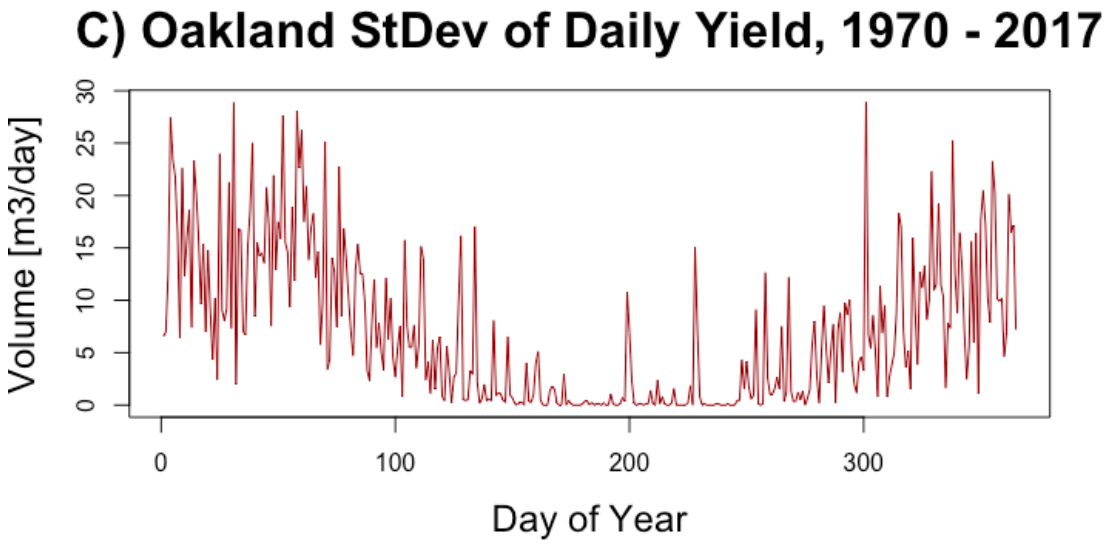


Figure 1. Mean daily yield and demand for (a) Oakland single family homes and (b) San Diego single family homes, and (c) standard deviation of daily (maximum) yield for Oakland, CA from 1970 to 2017.

Energy Data Sourcing:

Energy consumption data for RHS pumps were obtained through surveying literature on both theoretical and empirical studies. This data was then compared to commercially available RHS water pumps. The heterogeneity of building parameters in this study was taken into account when selecting energy values (Cheng, 2002) (Viera et al., 2014) (Ward et al., 2011).

The electricity utilities of interest are presented in Table 3. Power mixes from each utility are analyzed in order to determine the life cycle carbon emission factors used in estimating carbon emissions from water use.

The energy intensities associated with municipal water use vary by hydrologic region in California (Stokes et al. 2017). The variances in energy intensities stem from two critical factors: (1) differing treatment processes and (2) differing transportation distances. Energy intensity related to wastewater treatment largely depends on the number and type of unit process that each treatment plant employs. In terms of energy intensity related to transportation distances, the fact that water is not distributed evenly across the state means that in order to meet regional population demands, water is diverted from the northern part of the state to the southern regions. The energy required to pump water over the Grapevine increases the energy intensity of the water supply in the Southern California regions. Energy intensities for the hydrologic regions of interest are presented in Table 3. They are adapted from the 2020 regional intensities from (Stokes et al., 2017).

Table 3. Energy and Carbon Intensity Data.

City	Hydrologic Region	Energy Intensity (kWh/m^3)	Electricity Utility	Life-Cycle Carbon Intensity (g CO_2e/kWh)
Oakland	San Francisco Bay	0.69	Pacific Gas & Electric (PG&E)	145
San Diego	South Coast	2.1	San Diego Gas & Electric (SDG&E)	313

RHS Cost Sourcing:

The cost components for using water from a RHS and from a centralized, municipal source were obtained from several sources. Federal and statewide technical reports provided estimates for RHS capital

and maintenance costs (EPA, 2015) (Texas Water Development Board, 2005) (Cabell Brand Center, 2009). RHS operational costs depend upon daily energy consumption for pump operation and vary by each study city's electricity provider. Operational costs are the only costs considered when using municipally-provided water. Water utility costs also vary for each study city's water utility provider.

The valuation of non-monetary benefits of RHS is crucial for a holistic, comparative analysis. Estimating water conservation benefits is difficult because the price of water is volatile and subject to local circumstances. For the sake of this analysis, water conservation pricing figures and projections of growth from the EPA will be adopted (EPA, 2015). Similarly, valuation of stormwater runoff capture will follow a method developed by Sample and Liu (2014), where the value of runoff capture will be based on local stormwater utility charges. There currently does not exist any methodology or precedence for costing the benefits of the displacement of water infrastructure development. Therefore, it will not be included in this analysis.

Normalized costs and benefits of RHS are calculated in Equation 2:

$$C_R = (C_c + C_m + \lambda E_R) - (B_w + B_{sc}) \qquad (2)$$

Where:
C_R = *total cost of RHS ($/m^3)*
C_c = *capital costs of RHS (normalized to annual basis) ($/m^3)*
C_m = *annual maintenance cost of RHS ($/m^3)*
λ = *cost of electricity consumption ($/kWh)*
E_R=*energy consumed for RHS pumping (kWh/m3)*
B_w = *benefit associated with water conservation ($/m^3)*
B_{sc} = *benefit associated with stormwater runoff capture ($/m^3)*

Optimization Program:

Scope of Analysis:

The optimization program was solved on a daily time step. The daily time step was chosen because the highest resolution of rainfall and demand data are at the daily level. Since demand and rainfall occur simultaneously, the relatively coarse time step of 24 hours presents a challenge. We chose to model the RHS using Yield After Storage (YAS); all demand for the day is assumed to occur before any rain replenishes the tank on that day. This gives the most conservative estimate of RHS efficacy.

Based on RHS literature, a study period of at least 30 years is required to capture the full range of rainfall dynamics (Mun and Han, 2012). To ensure that our simulation is representative of all water years, we used almost 50 years of rainfall data spanning from 1970 to 2017 to study optimal RHS operation. Due to lack of information about historical demand and utility costs, we simulated a hypothetical scenario in which these factors remain at today's levels throughout the study period.

Math Model:

The RHS was modeled following a behavioral model described by (Palla et al, 2011), which is summarized in Figure 1. The primary constraint is a mass balance around the volume of water stored in the tank (V), with an influx due to rainfall collection (Q) and outflux of demand satisfied by rainfall (Y) and overflow (O) when volume stored exceeds storage capacity (S). The total demand for water (D) that is not satisfied by rainfall (Y) is made up by municipal water supply (M). The state variable, fluxes, and parameters in this model are listed in Table 4.

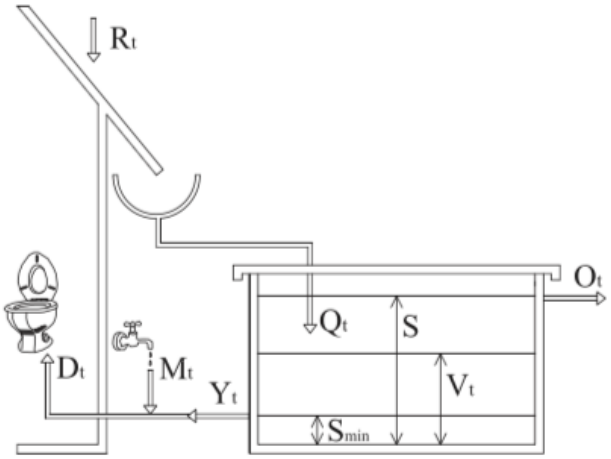


Figure 2. Rainwater harvesting mathematical model, to be used at a daily timestep. Adapted from: Palla, A; Gnecco, I; and Lanza, L.G. 2011. Non-dimensional design parameters and performance assessment of rainwater harvesting systems. Journal of Hydrology. 401, 65-76. Accessed on 2/6/2018 at 11:00 P.M.

Table 4. Variables, inputs, and parameters of the RHS model.

Symbol	Description and equation
S	RHS tank capacity
Y	Rainwater supply (i.e. the amount of demand that is satisfied by rainwater storage). According to the YAS operating rule, at any time t, $Y(t) = \min(D(t), V(t-1))$
M	Municipal water use (i.e. the amount of demand that is satisfied by the utility)
V	The stored volume in RHS tank. According to the YAS operating rule, at any time t, $V(t) = \min(V(t-1)+Q_t-Y_t, S-Y_t)$
D	Total daily water demand, which is assumed constant. At any time t, D = Y(t) + M(t)
O	Overflow. At any time step, O(t) = max(0, (V(t-1) + Q(t) - S)
R	Raw rainfall depth.
ϕ	Runoff coefficient, or the ratio between rainfall collected to the total rainfall. Typically set at 0.8 (Palla et al, 2011)
A	Rainfall collection area.
Q	Rain inflow to the storage tank. At any time t, Q(t) = **ϕ***R(t)*A
Smin	Minimum technical capacity of the tank. Typically set at 0.5 cubic meters (Palla et al, 2011)

Objective function, optimization variables and constraints:

The optimization variables are the demand provided by rainwater (Y(t)) and the demand provided by the utility (M(t)). These optimization variables were solved daily over for each city and building type; simulations initialize with a tank storage of $V_0 = 0.5*S$. Constraints are listed in Table 5. The variables were optimized for cost, energy use, and water use efficiency at each time step over an average rainfall year according to the following linear multi-objective function:

$$\text{minimize } f(t) = \sum_{t=1}^{T} (\alpha[(C_R(t))Y(t) + (C_M(t))M(t)] + \beta[(E_R)Y(t) + (E_M)M(t)] - \gamma[WSE(t)]) \qquad (3)$$

Where:
T = *total number of timesteps taken (days)*
α = *weighting factor of importance of cost*
β = *weighting factor of importance of energy*
γ = *weighting factor of importance of water use efficiency*
$C_R(t)$ = *cost of RHS* ($/m^3$)
$C_M(t)$ = *cost of municipal water* ($/m^3$)
E_R = *energy for pumping RHS* (kWh/m^3)
E_M = *energy for using municipal water* (kWh/m^3)
WSE = *water saving efficiency* = $Y(t)/D$, *where D is constant total daily demand*
$Y(t)$ = *available rainwater supply at time t* (m^3)
$M(t)$ = *municipal water supply at time t* (m^3)

Table 5. Constraints and their physical meaning for our optimization program.

Constraint	**Physical Meaning**
$D_t = Y_t + M_t$ (5)	Total demand (D) each day must always be satisfied either through RHS (Y) or the utility (M)
$V_t = Q_t - V_{t-1} - Y_t - O_t$ (6)	Mass balance of the rainwater stored in the tank
$Q_t = \phi * R_t * A$ (7)	Inflow of rainwater to the tank is proportional to rainfall and a loss factor.
$Y_t = \min [D_t, V_{t-1}]$ (8)	The yield of the rainwater storage will either partially or fully meet water demand.
$O_t = \max [0, (V_{t-1} + Q_t) - S]$ (9)	Allows for water overflow discharge when max storage capacity is reached.
$V_t \geq S_{min}$ (10)	The storage in the tank must be above minimum technical capacity.

Dynamic Programming:

The optimization problem was solved using dynamic programming with the yield (Y) from the rainwater harvesting tank as the control and storage (V) in the tank as the state variable.

The value function, in words, is as follows:

Let $V_k(Y_k, v_k)$ denote the minimum cost from time step k to terminal time step N, where the control, the yield from the tank, is Y_k and the resource level, the storage in the tank, in step k is v_k.

The principle of optimality equation is:

$$V_k(x_k) = \min\{ (\alpha[(C_R(k))Y(k) + (C_M(k))M(k)] + \beta[(E_R)Y(k) + (E_M)M(k)] - \gamma[WSE(t)]) + V_{k+1}(Y_{k+1}, v_{k+1}) \} \quad (4)$$

Because there is no terminal cost at the end of the year, the boundary condition of the value function is:

$$V_N(Y_N, v_N) = 0 \quad (5)$$

Due to computation time concerns, optimal yield from the RHS was only calculated over an average water year. This average water year was calculated for each city and represents the daily average rainfall that the city received from 1970 to 2017. Using an average water year allowed us to capture the seasonal patterns of rainfall over the past several decades without needing to optimize each year separately. For each optimization, the tank was initialized with a storage equal to half the tank capacity.

Comparative Scenario Analysis:

Two scenarios are analyzed in order to the assess the feasibility of implementing a RHS. The first scenario, Scenario 1, utilizes an optimized combination of rainwater and municipally-sourced water to meet non-potable demand. Scenario 2, which is the baseline condition, assumes that all non-potable demand is met by each city's water municipality. Energy usage, GHG emissions, and costs are compared for each scenario. Figure 3 provides a graphical representation of the two scenarios. Table 6 summarizes the main parameters used to conduct the analysis.

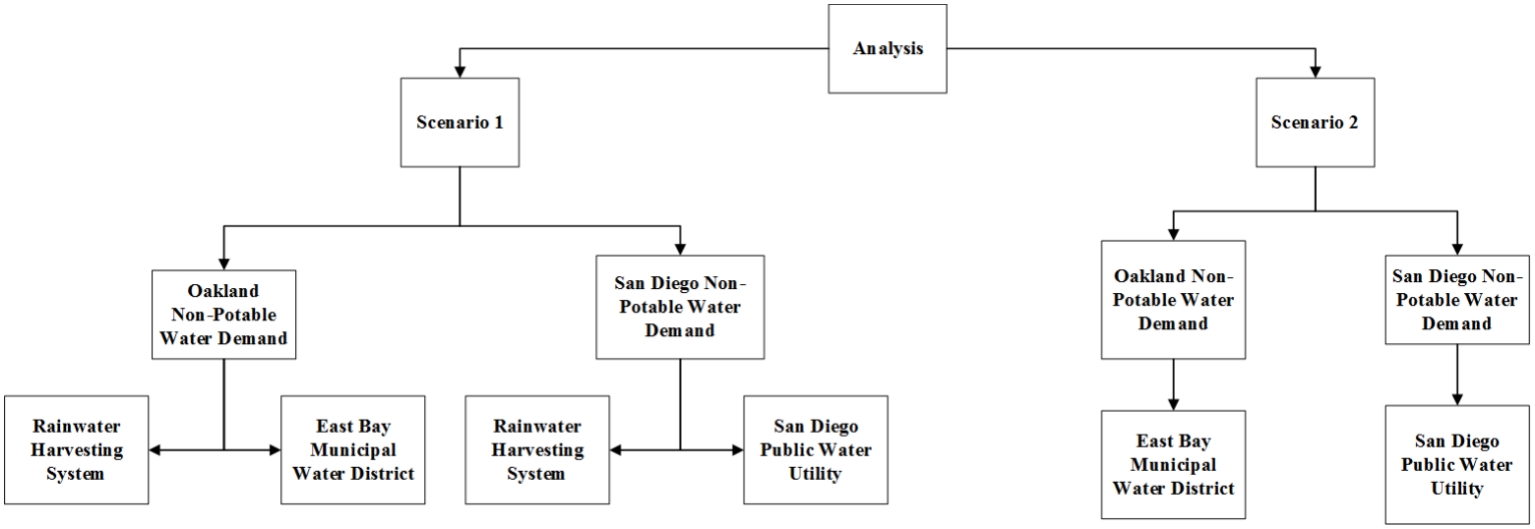


Figure 3. Flow chart describing the comparative analysis of the two scenarios conducted.

Table 6. Summary of main parameters used in the optimization program.

	Oakland	**San Diego**
Water Demand		
Single-Family Non-Potable Water Demand [m^3/day]	0.0616	0.0616
Municipal / Utility Data		
Municipal Water Intensity [kWh/m^3]	0.69	2.1
Utility Carbon Intensities [gCO_2/kWh]	145	313
Municipal Water Costs [USD/m^3]	*1.22*	*1.71*
Utility Energy Costs [USD/kWh]	*0.309*	*0.251*
RHS Data		
RHS Operational Costs [USD/m^3]	1.14	0.813
RHS Operational Energy Consumption [kWh/m^3]	5.6	5.6

IV. Discussion

Figures 4 and 5 display the results of the dynamic programs conducted for RHS use in a single-family residential home in Oakland and San Diego, respectively. The results reflect the team's a priori

expectations that the utilization of harvested rainwater follows the expected pattern of yearly rainfall. Reliance on RHS tends to be higher in the winter and fall than it is during spring and summer.

For the Oakland case (Figure 4), the non-potable water demand is met through RHS in a much higher capacity than it is in the San Diego case (Figure 5). The observed trend is related to historical rainfall patterns and the differing energy/GHG intensities of the two regions. Firstly, San Diego is extremely more arid than Oakland. The average yearly rainfall is much higher in Oakland as clearly indicated in Figures 4 and 5. Given the expected value of monthly rainfall data, the program attempts to ration water sourcing from RHS in both Oakland and San Diego. Additionally, in equation (3), one of the multi-objective criterion in the objective function is to reduce the energy intensity of the RHS system, and in turn the embodied GHG intensity of the sourced water. Since the RHS has the same energy intensity in both cities, the minimization objective is mostly contingent on the energy and GHG intensities of the water supplies and power mixes in both cities. The energy intensity of San Diego's municipal water supply is almost three times that of Oakland's. Additionally, the San Diego power mix GHG intensity is roughly twice that of Oakland's.

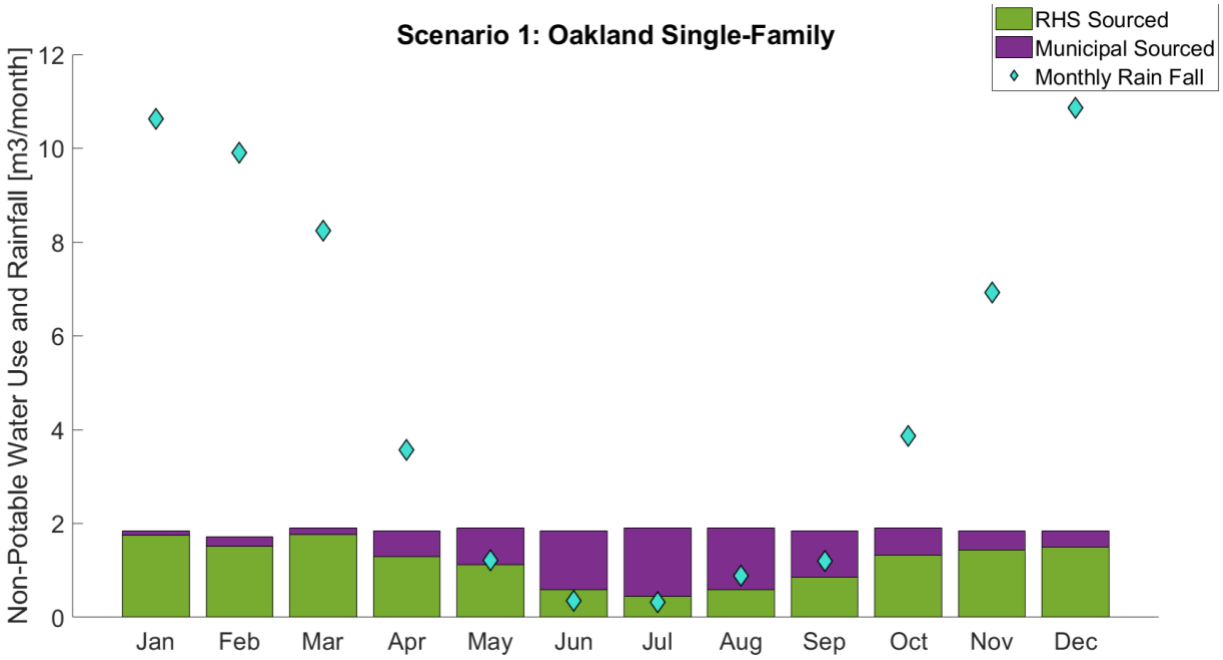


Figure 4. Dynamic program simulation results of RHS use in Oakland for an average rainfall year.

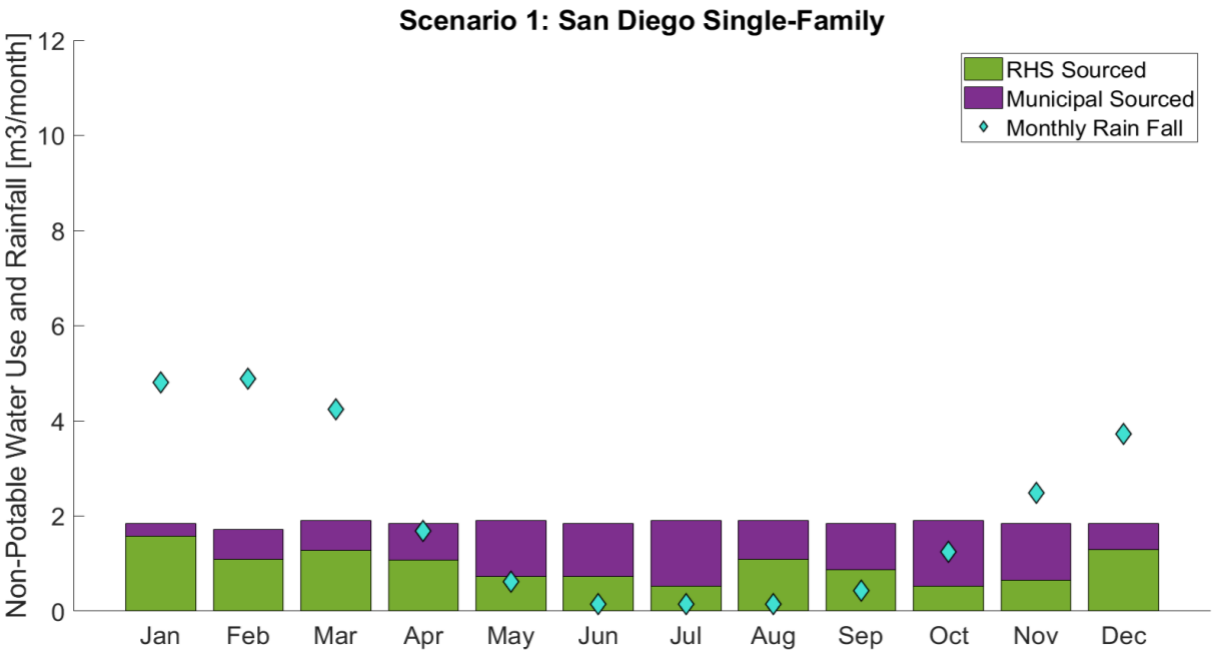


Figure 5. Dynamic program simulation results of RHS use in San Diego for an average rainfall year.

As can be seen in Figures 6 and 7, the results of the dynamic programming simulations indicate that energy usage and GHG emissions are greater for Scenario 1 than for Scenario 2. This means that the combination of using RHS water and municipally-sourced water to supply toilet demand is more energy intensive than using only municipally-sourced water. The use of a RHS would be an appropriate GHG mitigation strategy if it produced fewer emissions than municipally-sourced water. According to this report's system boundaries, a RHS is not an appropriate GHG mitigation strategy. This report only considered using RHS to supply indoor non-potable demand (i.e., toilet flushing water). RHS might be a more effective strategy for supplying irrigation demand, as there would be limited pumping requirements, and thus fewer GHG emissions.

Although a RHS might not serve as an effective GHG mitigation strategy, it is an effective water conservation strategy. Costs play an integral role when analyzing the water conservation benefits of RHS. The dynamic programming simulations provide cost differences between the two scenarios. For Oakland, the cost of using a RHS to meet non-potable demand is roughly commensurate with the cost of using only municipally-sourced water. In San Diego, utilizing a RHS to partially supply demand costs less than completely relying on municipally-sourced water. The cost components of the RHS only included operating expenses and monetized benefits associated with water conservation and stormwater management. If RHS capital costs were included, Scenario 1 would always be more expensive than Scenario 2 as the planning horizon is one year. An interesting policy component to consider is the use of subsidies to cover the capital costs of RHS. Some cities cover the cost of installing basic rainwater capturing systems. For example, San Diego provides rebates of up to $400 per property for residents who use rain barrels (City of San Diego 2018). At the state level, a proposed ballot measure, Proposition 72, would reward homeowners for building rainwater harvesting systems by preventing them from having to pay increased property taxes (Rogers 2018). As the cost of water increases in the future, water-stressed regions, such as parts of California, might further consider incentivizing the use of RHS.

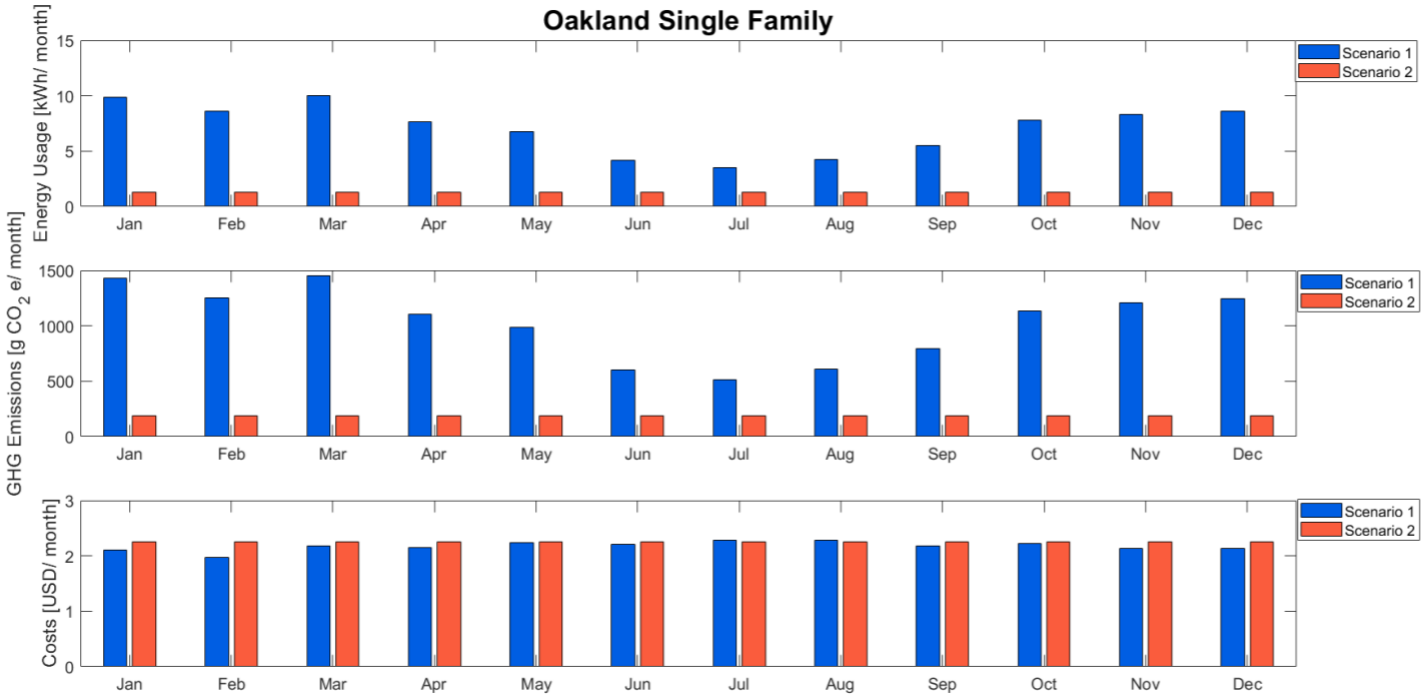


Figure 6. Energy intensity, GHG emissions, and cost comparison between the two modelled scenarios in Oakland.

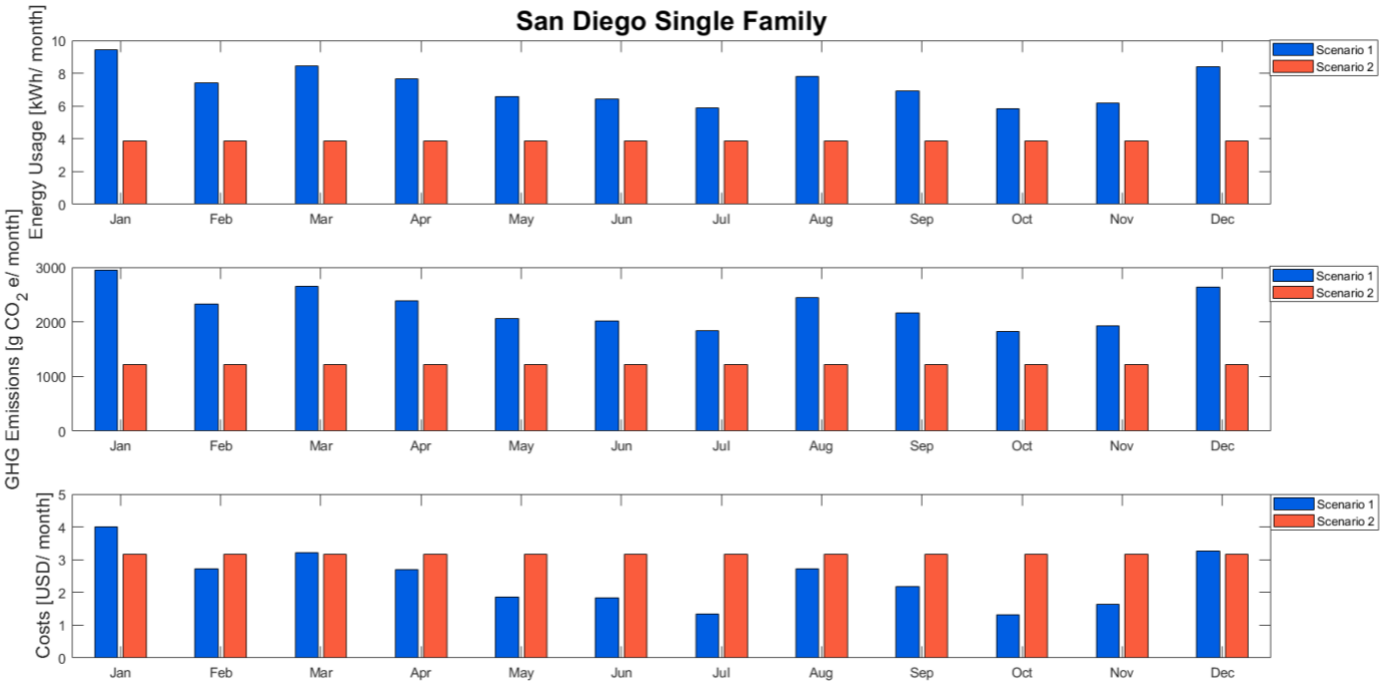


Figure 7. Energy intensity, GHG emissions, and cost comparison between the two modelled scenarios in San Diego.

One of the more interesting revelations of the obtained results is the apparent tradeoff between addressing GHG emissions reductions and water conservation goals. On the one hand, RHS perform well in addressing water scarcity and resource depletion issues by making use of water that would have otherwise gone to waste. Conversely, RHS are highly energy intensive in their operation. Having high

energy use requirements translates into a higher Global Warming Potential due to excess GHG emissions. Optimization and control mechanisms are uniquely positioned to assess the tradeoff between the two aforementioned environmental impact categories. Attempting to quantify this tradeoff through other methods would be more difficult, and won't offer the required resolution for an optimal analysis. One can leverage dynamic programming tools to perform analyses on systems that evolve with time, such as RHS, and are subject to conflicting constraints to obtain optimal system profiles.

Additional analyses can be performed on the water-sourcing profiles that resulted from the dynamic programming simulation. By translating both the water conserved and the emitted GHGs into monetary values, the trade-offs between the two criteria can be quantitatively analyzed. Total water savings through RHS are expressed in [m^3], while GHG emissions are expressed in gCO_2e. Water savings can be expressed in net savings by considering the monetary benefits of water conservation and stormwater capture ($\$/m^3$). GHG emissions can also be expressed monetarily by considering California's Carbon Cap-and-Trade program (Global Warming Solutions Act of 2006 (AB32)). The Cap-and-Trade program provides the social cost of carbon in $\$/gCO_2e$. Table 7 provides the transformation parameters of both values.

Table 7. Scalar parameters used to transform water savings and GHG emissions into costs.

Parameter	Scalar Value
Water Conservation Benefit	$\$1.81/m^3$
GHG Emission Costs	$\$0.0000151/gCO_2e$ (Climate Policy Initiative, 2018)

Only the Oakland case will be considered for this additional analysis. Figure 4 contains the total amount of water savings in Oakland. The water sourced from RHS, which is displayed in green, is considered "water-saved", and can be directly translated into water conservation benefits. Figure 6 contains the total GHG emissions from using municipally-sourced and harvested rainwater to meet non-potable demand in Oakland. The GHG emissions resulting from the RHS alone will be taken into account for the analysis. The carbon costs are found by multiplying the net RHS emissions by the scalar emission value found in Table 7. Table 8 shows the comparison between the water conservation benefits of RHS and their respective GHG emissions trade-off for Oakland. Two cases are considered: one in which capital and annual maintenance costs are included, and another in which capital and maintenance costs are excluded.

Table 8. Water Conservation / GHG Emissions Trade-off analysis (a negative value signifies net costs, whereas a positive value represents net benefits).

	Including Capital + Maintenance Costs	Excluding Capital + Maintenance Costs
Water Conservation Benefits [$/y]	-$1,300	$26
GHG Emission Costs [$/y]	- $0.18	-$0.18

Table 8 reaffirms the idea that under current costing schemes, RHS water conservation benefits do not justify their relative GHG emissions. The annual capital and maintenance costs dwarf the overall benefits by 50 times. In order for the water conservation benefits from RHS to justify their GHG emissions, a full annual subsidy of costs would be needed ($1300). Another way in which water conservation benefits of RHS can be justified is either through increasing the monetary benefit of water conservation to $\$92/m^3$, or through increasing the costs of GHG emissions to $\$0.086/gCO_2e$. Both values would lead to the trade-off costs and benefits to breakeven. The efficacy of changing costing schemes to such high values is quite unreasonable. While it is true that water is severely discounted and the costs of

GHG emissions are highly undervalued, it seems unlikely that any political or administrative body would approve of the needed valuations, especially considering the scale at which water is consumed and GHGs are emitted.

In terms of the overarching analysis of RHS, future recommendations for research should look into possibly setting energy efficiency targets for RHS, and how those targets could affect the optimal water sourcing scheme. It would also be interesting to see how RHS perform against other water conserving technologies such as efficient household water appliances (e.g., toilets, dishwashers, cloth washers). Specifically, the analysis should consider opportunity costs by looking into whether the needed subsidies for RHS should be used for other water appliances that may conserve more water. Lastly, in terms of the dynamic programming aspect of the analysis, stochastic rainfall that is characterized statistically rather than deterministically should be used in the analysis for increased robustness.

V. Summary

This report presents an analytical framework for assessing the feasibility of implementing rainwater harvesting systems (RHS) in California to meet non-potable demand in single-family buildings. Analyzing the feasibility of RHS is especially relevant and important for California because of the state's history of water-scarcity and the state's energy intensiveness of its water supplies. The team chose to look at Oakland and San Diego for the study, as these two cities represent best and worst case scenarios in terms of rainfall and energy intensities of their respective water supplies.

The team developed two scenarios to study: (1) Using a combination of harvested rainwater and municipally-sourced water to meet toilet flushing water, or non-potable, demand in a single-family home and (2) Utilizing only municipally-sourced water to meet non-potable demand. The team created an optimization program with the objective of minimizing the costs and energy usages associated with implementing a RHS and of maximizing the water saving efficiency of the RHS. Using dynamic programming, the team solved the optimization problem for a representative average year of rainfall. For both the Oakland and San Diego case studies, energy and greenhouse gas emissions are greater for the scenario that utilizes harvested rainwater than for the scenario that only utilizes municipally-sourced water. When including net benefits of stormwater management and water conservation, RHS in both cities cost less than municipally-sourced water. However, when capital costs are included, RHS are not cost-effective.

From the results of the dynamic program, it is clear that RHS are currently not an affective GHG mitigation strategy in California. The water conservation benefits of RHS are clear. Homeowners could be more likely to implement RHS if local governments or policy-makers provided subsidies to cover the capital costs of the systems. One way in which the RHS water conservation and GHG emission trade off can be quantitatively assessed is by translating both values into costs. The translation can be performed using current costing schemes for water conservation benefits issued by the EPA, and the current cost of carbon through California's Cap-and-Trade program. The results reaffirm the notion that the water conservation benefits of RHS do not justify their GHG emissions. As water availability becomes more uncertain in the future, RHS might become a more acceptable strategy for conserving water.

VI. References

ARUP. 2012. The Technical Feasibility of Zero Net Energy Buildings in California. Report ZNE/219664, Pacific Gas and Electric.

Cabell Brand Center. 2009. Virginia Rainwater Harvesting Manual. Second Edition. Accessed on 3/18/2018.

Cheng, C.L. 2002. Study of the inter-relation between water use and energy conservation for a building. Energy Build. 34, 261–266.

Chiu, Yie-Ru; Liaw, Chao-Hsien, and Chen, Liang-Ching. 2009. Optimizing rainwater harvesting systems as an innovative approach to saving energy in hilly communities. Renewable Energy. 34, 492-498.

City of San Diego. 2018. "Rainwater Harvesting Rebates". https://www.sandiego.gov/water/conservation/rebates/rainbarrel. Accessed on May 2, 2018.

Climate Policy Initiative. 2018. "California Carbon Dashboard". http://calcarbondash.org/. Accessed on May 3, 2018

Farreny, R; Gabarrell, X; and Rieradevall, J. 2011. Cost-efficiency of rainwater harvesting strategies in dense mediterranean neighbourhoods. Resources, Conservation and Recycling. 55, 686-694.

Fewkes, A. 2000. Modelling the performance of rainwater collection systems: towards a generalized approach. Urban Water. 1, 323-333.

Hajani, E. and Rahman, A. 2014. A. Reliability and cost analysis of a rainwater harvesting systems in peri-urban regions of Greater Sydney, Australia. Water, 6, 945–960.

Jenkins, D.; and Pearson F. 1978. Feasibility of Rainwater Collection Systems in California. Technical Report. NTIS 197906. California Water Resources Center.

Mun, J.S. and Han, M.Y. 2012. Design and operational Parameters of a rooftop rainwater harvesting system: Definition, sensitivity and verification. Journal of Environmental Management. 93, 147-153.

National Oceanic and Atmospheric Administration, Climate Data Center. https://www.ncdc.noaa.gov/ Accessed on 16 Mar 2018.

Palla, A; Gnecco, I; and Lanza, L.G. 2011. Non-dimensional design parameters and performance assessment of rainwater harvesting systems. Journal of Hydrology. 401, 65-76.

Pacific Institute. 2014. Water Reuse Potential in California. Issue Brief IB:14-05-E.

Rahman, A., Dbias, J., and Imteaz, M. 2010. Sustainability of rainwater harvesting systems in multistory

residential buildings. Am. J. Eng. App. Sci.: 3:889-98.

Rodgers, Paul. 2018. “New state ballot measure would reward people who build rainwater collection systems”. *Bay Area News Group*.

Roebuck, R.M.; Oltean-Dumbrava, C.; Tait, S. 2011. Whole life cost performance of domestic rainwater harvesting systems in the United Kingdom. Water Environ. J., 25, 355–365

Sathe, Amul, Michael Noreika, and Miriam Morris. 2012. *On-Site Water Generation: An Analysis of Options and Case Study*. California Sustainability Alliance. http://sustainca.org/sites/default/files/On-Site_Water_Generation-Final_Report.pdf.

Sample, D.J. and Liu, J. 2014. Optimizing rainwater harvesting systems for the dual purposes of water supply and runoff capture. J. Clean. Prod., 75, 174–194.

Stokes-Draut, J. et. al. 2017. Evaluating the Electricity Intensity of Evolving Water Supply Mixes: The Case of California’s Water Network. Environmental Research Letters.

Swamee, P. and Sharma, A. 2008. Design of water supply pipe networks. New York: John Wiley.

Tam, V., Tam, L., and Zeng, S. 2010. Cost effectiveness and tradeoff on the use of rainwater tank: an empirical study in Australian residential decision making. Resour. Conserv. Recycle.; 54:178-86.

Texas Water Development Board. 2005. The Texas Manual on Rainwater Harvesting. Third Edition. Accessed on 3/18/2018.

U.S. Environmental Protection Agency. 2013. Rainwater Harvesting: Conservation, Credit, Codes, and Cost Literature Review and Case Studies. Report EPA-841-R-13-002. Accessed on 3/18/2018.

Vieira, A.S. Beal, C.D. Ghisi, E. Stewart, R.A. 2014. Energy intensity of rainwater harvesting systems: A review. Renewable and Sustainable Energy Reviews, Volume 34, Pages 225-242, ISSN 1364-0321.

Ward, S., Butler, D., and Memon. F. 2012. Benchmarking energy consumption and CO2 emissions from rainwater-harvesting systems: An improved method by proxy. Water and Environment Journal 26(2)

Wildfire Modeling: A Case Study on the 2013 Rim Fire

Wes Adrianson, Daniel Herron, Emily Peterson, and Borna Poursheikhani

Abstract

Motivated by recent wildfire events in California, this project seeks to model the spatiotemporal evolution of past and future fires. Using semi-empirical fire equations and ecological inputs such as vegetation dryness, wind speed, and terrain data we aim to predict the rate of wildfire spread using a dynamical model to adapt to real-time changing conditions. We hope that this tool could one day be used to advise evacuation plans, inform bulldozer lines/pre-burn regions, and guide wildfire mitigation efforts. Our model will build upon previous research efforts in this field by linking the dynamical analysis of Mandel with the spatial construction presented by Ntaimo, using Richard Rothermal's mathematical wildfire spread model as our empirical foundation. We apply our model to the Rim Fire in California, modeling the spread of the leading edge of the fire and using a least squares regression to improve our daily spread predictions. We hope that this implementation of machine learning will fill in the knowledge gaps that currently exist in first principles understanding of wildfire spread.

Introduction

Motivation and Background

As average global temperatures rise, we have seen exacerbated extreme weather events. Higher temperatures during the spring and summer along with snow melting earlier in the spring have led to hot, dry conditions. These arid circumstances lead to increased probability, duration, and intensity of wildfires. Data from the National Interagency Fire Center shows that wildfire-burned areas in the United States have been increasing since the 1980s. As climate change continues, wildfire frequency and intensity is only expected to increase. In order to protect the lives and properties of people in high risk areas, there is a need to prevent these fires or lessen their damage whenever possible.

Recently, California's Napa and Sonoma Counties experienced a devastating wildfire killing over 40 people and destroying 8400 structures. The suspected origin of the fire was high winds causing power lines to throw sparks that ignited nearby vegetation. Ideally, we can learn from this misfortune and prevent similar occurrences in the future. There are many factors affecting the ease with which wildfires spread. However, by taking into account wind speed, temperature, dryness of vegetation, and proximity to power lines, our goal is to model the severity of a wildfire starting at various locations. The hope is that firefighters may us this tool to prevent, and, if necessary, minimize the damage from wildfires.

Battling large scale wildfires includes pre-burning vegetation in the fire's path in a controlled fashion, or bulldozing and removing the trees to stop or stall the progression of the fire. Our goal is to create a tool that firefighters and wildfire experts will be able to use to inform their evacuation plans and bulldozer lines given a predicted rate of spread. Figure 1 shows a bulldoze fire prevention plan-- a potential real world application of our modeling tool.

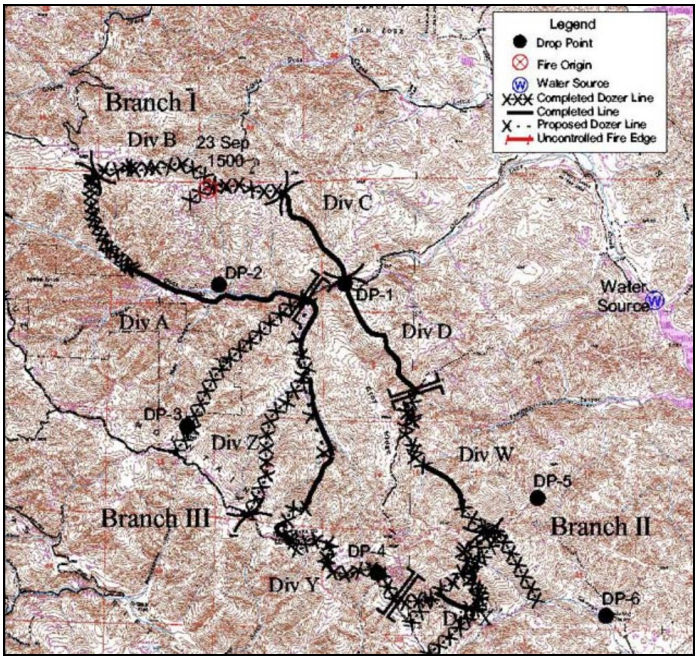


Figure 1 - Croy Fire Bulldozer Map. Retrieved from http://www.fire.ca.gov/cdf/incidents

Literature Review

Mandel et al built a real-time coupled atmospheric-wildland fire modeling system that uses machine learning to incorporate real data and strikes a balance between model complexity and fast execution by relying primarily on just two partial-differential-equations in 2D. The equations are based on balance equations for energy and fuel. Data in the model includes atmospheric conditions, fire, fuel, terrain, and more. This method uses a first principles model combined with an ensemble Kalman filter technique with regularization (Mandel et al, 2008). However, the first principles used to formulate this model are not common in fire spread modeling.

In 1972, Richard C. Rothermel published a widely cited research paper for the USDA Forest Service titled, " A Mathematical Model for Predicting Fire Spread in Wildland Fuels". This paper details the development of a mathematical model for predicting the rate of fire spread and intensity. The only necessary inputs of the model are the physical and chemical makeup of the fuel and the expected environmental conditions. The model is used to assess fire spread and intensity in the National Fire Danger Rating System. The derived mathematical model for quasi-steady state rate of spread (ft/min) is:

$$R = \frac{I_R \xi (1 + \phi_W + \phi_S)}{\rho_b \varepsilon Q_{ig}}$$

where I_R represents the reaction intensity (btu ft^{-2} min^{-1}), ξ is the propagation flux ratio, ϕ_W and ϕ_S are wind and slope parameters, ρ_b is the fuel particle density, ε is the fuel's effective heating number, and Q_{ig} is the heat of preignition (btu lb^{-1}) (Rothermel, 1972).

In another influential paper, "DEVS-FIRE" (Ntaimo, et al) constructed a cellular space model with spatial fuels data, terrain data, and temporal weather data to predict wildfires across time and space. The model uses the DEVS, a modeling and simulation framework based on generic dynamical systems concepts, an emerging tool for modeling complex adaptive systems like wildfires. Our approach is similar to the DEVS-FIRE model in that it combines Rothermal's first principles model with a systems technique.

Two of the most widely accepted fire behavior predictive models are FARSITE and BehavePlus. FARSITE relies on Huygen's principle of wave propagation, where fire growth is simulated as a 2D elliptical wave using spatial GIS data. The firefront is projected over a finite time step, in which local raster information on fuels, topography, and weather predicts a speed and direction of fire spread at each 2.5m x 2.5m cell using the Rothermel model. Aggregating these points around the fire perimeter creates the model for 2D fire growth. FARSITE is currently used by both CALFIRE and the U.S. Forest Service for training and operations during large wildfire events (Gollner et. al). The DEVS-FIRE model was designed to be integrated with a stochastic optimization model for the deployment of firefighting resources quickly and at low cost (Ntaimo, et al). Researchers noted that different GIS data or applications of different resolutions resulted in significantly different fire spread shapes. Results weren't compared to the spread of real fires, clearly an area of future development in the space.

Two of the major inputs across all simulation methodologies is "class of fuel" and wind. There are 13 classes of fuel as spelled out by the 13 Anderson Fire Behavior Fuel Models. These fuel classes consolidate twig size and moisture content into a "fuel class" which is directly linked to its flammability.

The wind used by Petrasova et al. was broken up into two main components—midflame velocity and direction, both of which can be spatially variable and must be accounted for in an accurate fire spread simulation. It is noted by all sources that variability in wind leads to high uncertainty in all fire simulation models. Similarly, the data required to initialize and parametrize these models such as fuels, topography, weather, etc., are also subject to large uncertainties and limited resolution (Gollner et al).

Figure 2: 13 Classes of Fuel (Source: Petrasova, Anna et al.)

Fuel class	Description
Grass and grass-dominated	
1	Short grass (1 foot)
2	Timber (grass and understory)
3	Tall grass (2.5 feet)
Chaparral and shrub fields	
4	Chaparral (6 feet)
5	Brush (2 feet)
6	Dormant brush, hardwood slash
7	Southern rough
Timber litter	
8	Closed timber litter
9	Hardwood litter
10	Timber (litter and understory)
Slash	
11	Light logging slash
12	Medium logging slash
13	Heavy logging slash

The original 13 Fire Behavior Fuel Models work well for predicting fire spread during peak season when conditions are dry. However, these models do not work well for certain situations such as prescribed fires, wildland fire use, simulating fuel treatment effects, and crown fires (Scott et. al). Scott et. al developed a new fuel model that incorporated live herbaceous components as dynamic. This means that the load shifts between live and dead depending on the moisture content. We elected to use the Standard Fire Behavior Fuel Model for our model. The Fuel Model Parameters are included in Appendix II.

There is currently very little technology being implemented on the ground to help firefighters make dynamical decisions in real time. There is no shortage, however, in preventative firebreak simulations. Many fire simulations and spread prediction models use Tangible Landscape and the GRASS GIS wildfire toolset to model impacts of different orientation firebreaks on the spread of a fire (Petrasova et al.). Of course these models require knowing the source of the fire, which is unrealistic in a wildfire scenario. Thus we hope to create a more dynamical system that can augment firefighting decision making in real time. It is no secret that this is a paramount issue and yet, despite the development of a myriad of fire models, “their use has been relatively limited operationally” (Gollner et al). This is in large part due to the lack of fundamental understanding of fire physics as they relate to different regimes (Gollner et al). Recently, some work has been done by Gollner et al. regarding the bridging of simulation gaps by using data-driven modeling which essentially supplements low resolution simulations with real-time observations of wildland fire dynamics.

Focus of This Study

The primary goal of this study is to integrate foundational fire spread theory with the methods we have learned throughout this course to develop a unique approach to modeling wildfires. We will focus on

training a Rothermel-based ordinary least squares model on the 2013 Yosemite Rim Fire in order to dynamically model and forecast one-dimensional fire spread in the future.

Technical Description

The basis of our model is derived from the fire spread fundamentals provided in Rothermel (1972). In Rothermel's analysis, the conservation of energy principle is applied to a unit volume of fuel to show that the quasi-steady state rate of spread can be defined by the ratio between the heat flux received from the fire and the heat required for ignition. The resulting rate equation is shown below,

$$R = \frac{I_R \xi (1 + \phi_W + \phi_S)}{\rho_b \varepsilon Q_{ig}}$$

where I_R represents the reaction intensity (btu ft^{-2} min^{-1}), ξ is the propagation flux ratio, ϕ_W and ϕ_S are wind and slope parameters, ρ_b is the fuel particle density, ε is the fuel's effective heating number, and Q_{ig} is the heat of pre-ignition (btu lb^{-1}). It should be clear, then, that the rate of spread is very closely related to the fuel characteristics (surface-area-to-volume ratio, moisture content, etc.) and environmental conditions (wind and ground slope). These relationships were defined through empirical analysis, and the results can be found in Appendix I.

Although the Rothermel Model provides a semi-empirical formulation of fire spread, it is very limited in its scope of application -- a lack of first-principle understanding of fire physics has proven to be a huge barrier in the field of wildfire modeling/prediction. The industry standard fire modeling techniques include overload and under supply specific variables within the Rothermel to generate a "realistic" prediction. This approach is highly subjective and leaves no room for reproducibility in the absence of a fire expert. We hope that by applying machine learning, more specifically a least squares regression, we will be able to bypass the need for first principles and provide an actionable, dynamic tool for wildfire responders to use.

Discretization and Modeling

Several key parameters for the fire spread equations, such as fuel loading, fuel depth, and slope, vary across space in the landscape. Satellite Lidar data was processed to provide fairly effective estimates of these parameters, limited only by the level of pixel detail. We utilized ArcGIS to render these spatial landscape data rasters before ultimately processing the model in Matlab.

Due to the pixelated structure of our data, we discretized the Rothermel fire spread model in space. To demonstrate proof of concept, our initial model was 1-dimensional. In other words, it calculated the spread of a fire along a line with parallel wind flow.

The first-generation 1-dimensional model utilized a discrete structure where each "pixel" in our 1-dimensional landscape is defined as 40 feet across. In this example, the landscape is completely uniform, with typical parameter values for grassland fires, a constant 45-degree slope, and constant wind. Each discrete unit of time is 1 minute. A "status" variable keeps track of whether a pixel is unburned, burning, or burned.

$$s \in 0,\ 1,\ 2 \quad (0 = \text{unburned},\ 1 = \text{burning},\ 2 = \text{burned})$$

A “fuel load burned” variable keeps track of the percentage of each pixel burned during each time step.

$$F \in [0,\ 1]$$

The for-loops calculate a “rate of spread” at each time step for any pixels that are defined as currently burning (s=1).

$$R \in [0,\ \infty]$$

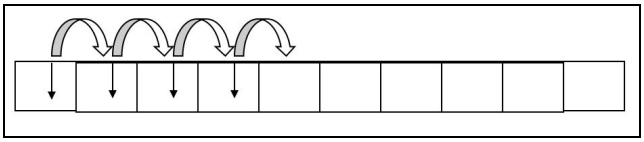


Figure 4: Illustration of 1-Dimensional pixel model

Our initial model includes 50 pixels (200 feet total) and a timeframe of 120 minutes. The plot below shows the calculated and discrete rates of spread across the 1-dimensional landscape.

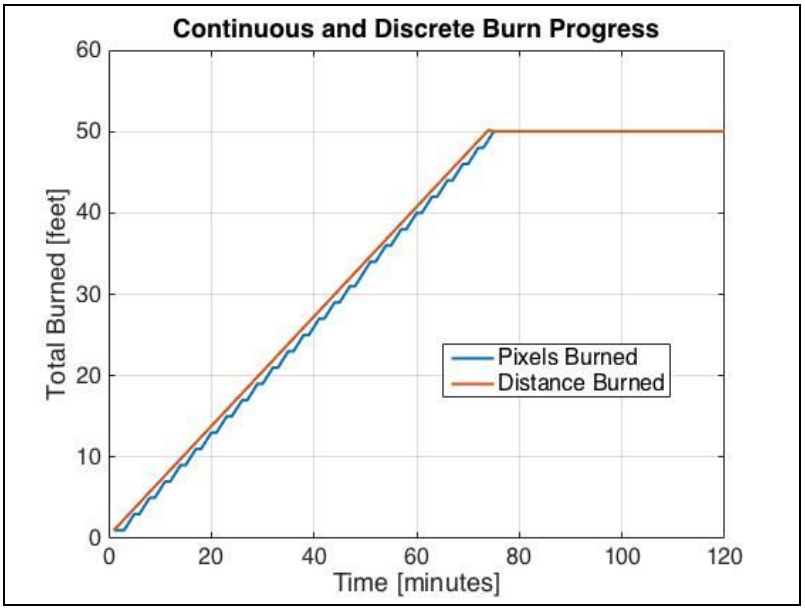


Figure 5: Calculated and discrete rates of spread

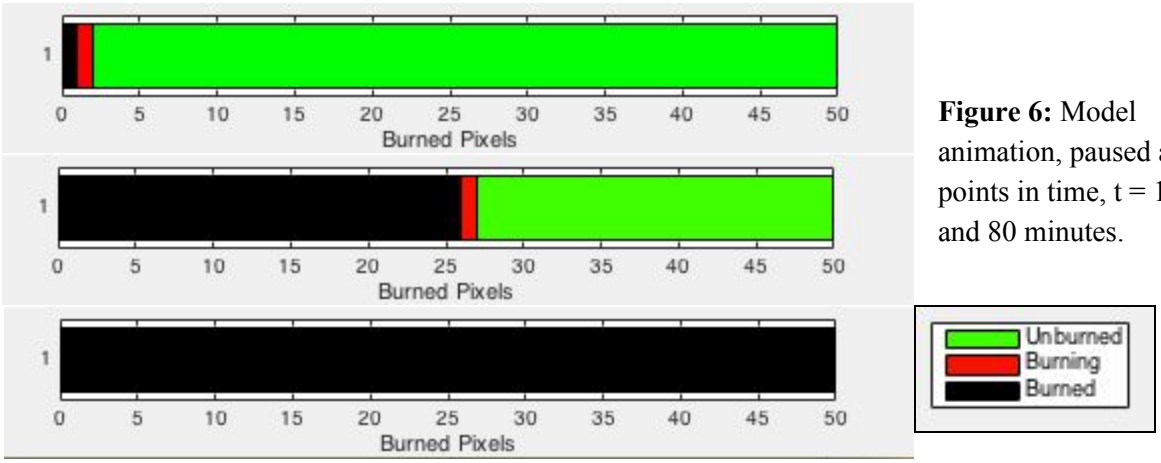


Figure 6: Model animation, paused at three points in time, t = 1, 40, and 80 minutes.

The animation shows the spread of the fire (red) across the 1-dimensional landscape.

The second generation of our 1-D model applies the same methodology but uses actual GIS Data to populate the mathematical model and conduct daily adjustments to the model parameters via historical burn data. Spatial, elevation and fuel data was obtained for every 37.7 square meter pixel of the Rim Fire. Each pixel was assigned a Fuel Model Code that corresponds with the Fuel Model Parameters in Appendix II. This provided the fuel data inputs for the Rothermel equation for each pixel. Additionally, we were able to calculate the slope based on USGS Digital elevation model data. The World Weather Online API was used to collect the wind velocity and direction for every hour of the 23 day fire.

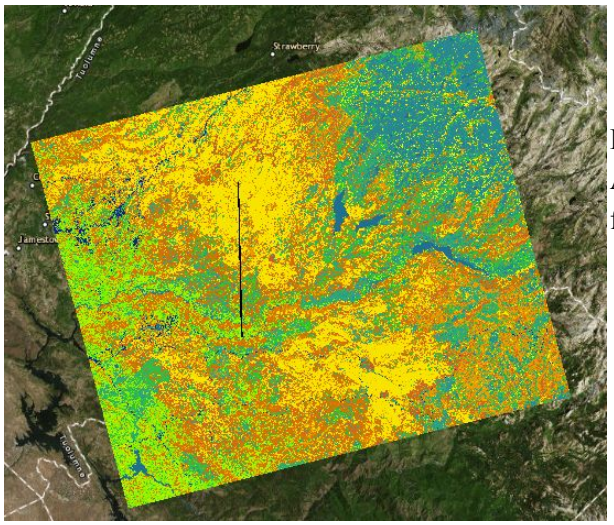


Figure 7: LIDAR Vegetation Type Fuel Loading Analysis. Areas of yellow and orange represent high fuel loading.

We took the vector components of the wind and slope data such that all vectors included in the model are parallel to the 1-D pixel trajectory, creating a chain of discretized fire-spread rates unique to each pixel. The fire model is then run with these variable rates to observe the spread of the leading edge of the fire in 1-D.

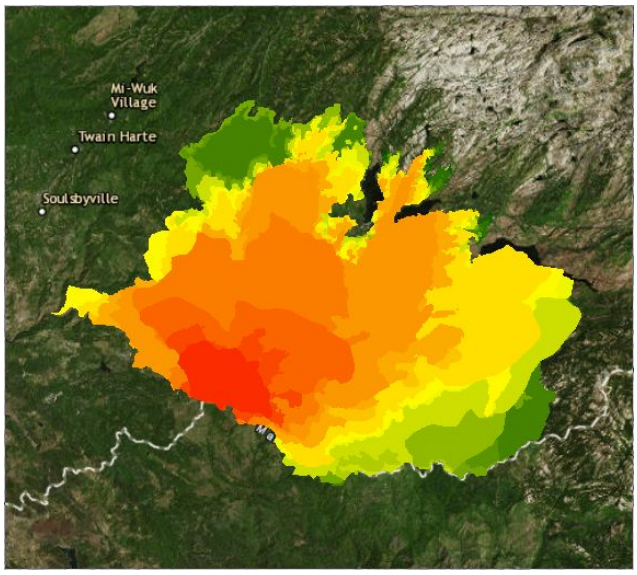


Figure 7: Rim fire burn record from 8.17.13 to 9.3.13 (shapefile provided courtesy of Berkeley Stephens Lab on Wildland Fire Science).

The third generation of our 1-D model adopts machine learning in the form of a least squares regression, pulling data from a real fire progression and using the error between actual fire spread and our model to train our model and improve future predictions. Here, we first had to discretize the Rothermel equation and multiply by the burn time across a given pixel to calculate a distance burned.

$$f(u_i) = \sum_{i=1}^{N} \frac{I_{R,i}\xi_i}{\rho_{b,i}\varepsilon_i Q_{ig,i}}\left(1 + \phi_{w,i} + \phi_{s,i}\right)\Delta t_i$$

Next, we distributed the first term throughout the rest of the equation and include our optimization variables, $\boldsymbol{\theta}$. From left to right, each term in this equation can be viewed as the fuel, wind, and slope influences on the total fire spread in a given pixel.

$$f(\theta, u_i) = \sum_{days} \sum_{i \in day} \left[\theta_1 \frac{I_{R,i}\xi_i}{\rho_{b,i}\varepsilon_i Q_{ig,i}} + \theta_2 \frac{I_{R,i}\xi_i}{\rho_{b,i}\varepsilon_i Q_{ig,i}}\phi_{w,i} + \theta_3 \frac{I_{R,i}\xi_i}{\rho_{b,i}\varepsilon_i Q_{ig,i}}\phi_{s,i}\right]\Delta t_i$$

Our least squares regression is composed of a quadratic optimization that seeks to minimize the error between the actual fire spread and our model predictions, using $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, and $\boldsymbol{\theta}_3$ from the above equation. We were then able to formulate our optimization program as follows, with X representing fuel, wind, and slope input data for each pixel and Y representing the actual fire spread for each day.

$$min_\theta \|\theta X - Y\|_2^2 \qquad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \qquad X = \begin{bmatrix} \frac{I_{R,1}\xi_1}{\rho_{b,1}\varepsilon_1 Q_{ig,1}} & \frac{I_{R,1}\xi_1\phi_{w,1}}{\rho_{b,1}\varepsilon_1 Q_{ig,1}} & \frac{I_{R,1}\xi_1\phi_{s,1}}{\rho_{b,1}\varepsilon_1 Q_{ig,1}} \\ \frac{I_{R,2}\xi_2}{\rho_{b,2}\varepsilon_2 Q_{ig,2}} & \frac{I_{R,2}\xi_2\phi_{w,2}}{\rho_{b,2}\varepsilon_2 Q_{ig,2}} & \frac{I_{R,2}\xi_2\phi_{s,2}}{\rho_{b,2}\varepsilon_2 Q_{ig,2}} \\ \vdots & \vdots & \vdots \\ \frac{I_{R,N}\xi_N}{\rho_{b,N}\varepsilon_N Q_{ig,N}} & \frac{I_{R,N}\xi_N\phi_{w,N}}{\rho_{b,N}\varepsilon_N Q_{ig,N}} & \frac{I_{R,N}\xi_N\phi_{s,N}}{\rho_{b,N}\varepsilon_N Q_{ig,N}} \end{bmatrix} \qquad Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{bmatrix}$$

For the purpose of this study, we simulated our model's interaction with daily data by utilizing historical fire progression maps. In a real world application, this tool would take in real time fire spread data (as frequently as possible) to train the model quickly and predict the spread over the next few days.

Given the inherent unpredictability of wildfire spread, we hope that our machine learning augmentation will fill in the gaps between Roethermal's idealized model and reality. We hope to create a foundation for dynamical fire modeling and prevention to help mitigate the increasing devastation caused by these anthropogenically intensified natural disasters.

Discussion

The base Rothermel model successfully tracks the fire progress across 730 discrete units of space (37.7m pixels) and 23040 discrete units of time (minutes). A variable titled 'status' keeps track of which pixels have burned, are burning, and are not yet burned at each time step. The code calculates a unique value of R for each pixel that is determined to be currently on-fire at each time step, referencing unique hourly wind data and pixel-specific slope and vegetation fuel loading data. A general location parameter was adjusted to tune the Rothermel model so that it would burn roughly the full distance of the fire in the appropriate number of days. The results of the Rothermel forecast model are plotted with the orange line in Figure 8. The actual daily burn record of the fire is plotted with the stepwise yellow line.

The second model incorporates learning via the ordinary least squares optimization described in the previous section. The Rothermel model calculates a unique burn rate for each minute. At each new day, the code conducts a CVX optimization to adjust the three theta-parameters, which primarily reflect relative importance of fuel loading, slope, and wind in the model. The previous day's Rothermel model forecast is compared to the actual historical record of the fire. The CVX includes bounds of -15 and 15 to prevent extreme theta definitions.

The Rothermal model with learning demonstrates a successful adjustment of theta parameters after the rapid fire-burn in Day 4. Real burn data for some days, including Day 5, is missing. The forecast speeds up again after Day 6, where the optimization again responds to the deficit between the model and the real burn. After Day 8, the parameters are made smaller again by the least squares optimization, responding to the model slightly overshooting the actual burn. All 3 theta parameters, representing relative importance of fuel, wind, and slope in the model, were significant throughout. The theta with the largest magnitude was wind and the second largest magnitude was slope. This indicates that variation in wind and slope were generally more significant in predicting the real fire burn rate. The results of the Rothermel model with learning are plotted with the blue line in Figure 8. The RMSE was 864.5, an improvement compared to the base model RMSE of 966.7.

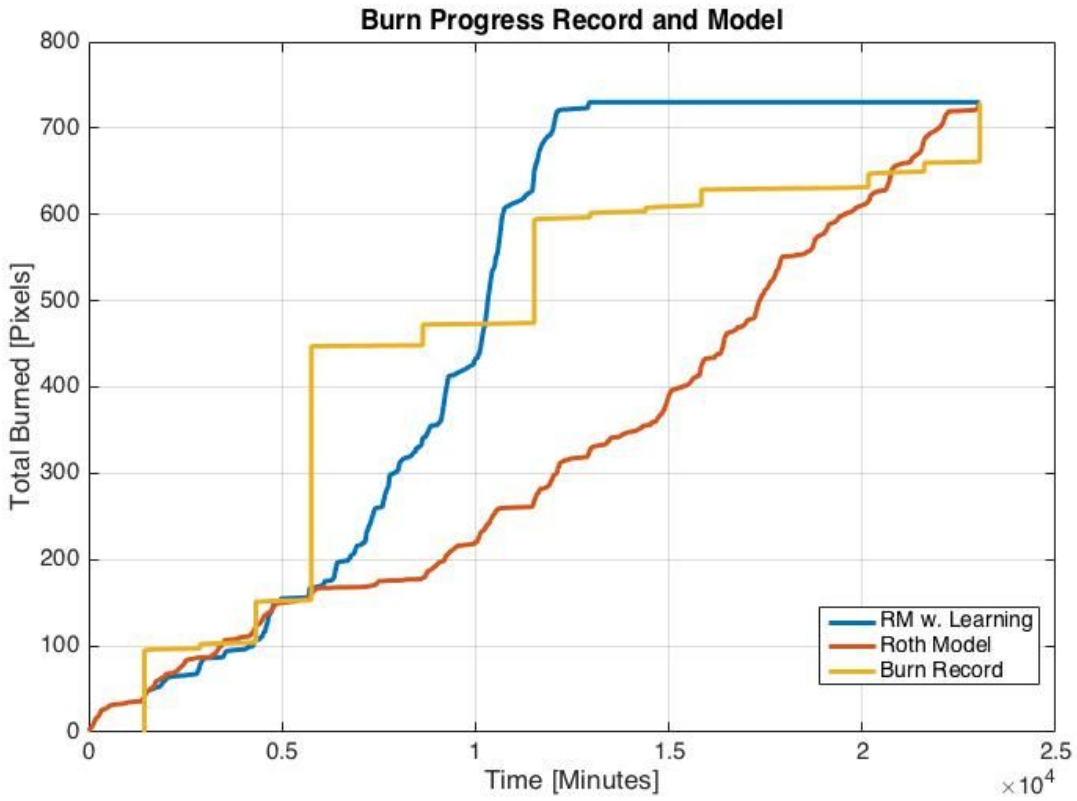


Figure 8: Plots comparing the historical burn record with the base Roth. Model and the Roth. Model with learning.

	Rothermal Model	Rothermal Model with Learning
RMSE	966.7	864.7

Summary

In the last year, many areas throughout California have been devastated by wildfires. As the impacts of climate change become more apparent each day, it is clear that these events will continue to pose a serious threat to the state. The main purpose of this project was to better understand the spatiotemporal evolution of these fires by creating a dynamical model.

Using the foundational fire spread theory presented in Rothermel (1972) along with vegetation, wind, and slope data, we were able to model the spread of the 2013 Yosemite Rim Fire. However, the Rothermel model has several limitations when it comes to extreme fire events such as this. To account for these shortcomings and to present a new approach to solving the problem of fire spread modeling we applied a

linear least squares optimization on parameters representing three of the most significant inputs: fuel, wind, and slope. The goal here was to outperform the rate of spread estimates presented in the Rothermel model, and our success can be confirmed by the RMSE values of 966.7 and 864.7, of the base and least squares model, respectively. This is the first known demonstration of machine learning in wildfire modeling.

While our model does provide some exciting new results, we believe there still exists significant room for further research in the fire modeling space. Most importantly, in order to more accurately apply machine learning algorithms to this problem, it is imperative that more detailed fire spread data be recorded and made publicly available during the evolution of wildfires. When a model can only "learn" at the end of each day, it is difficult to make accurate predictions about the hourly behavior of a fire. Additionally, the model should incorporate data from many different fires in many different regions to best forecast a variety of possible conditions. Of course, we would also like to expand the model to fire spread in 2-dimensions in future iterations.

Fire spread models are used to inform containment and evacuation efforts anytime a wildfire breaks out. Improving these models will help insure citizens' immediate safety and direct the efforts of firefighters trying to mitigate the spread. In this regard, we hope that our report outlines a useful machine learning approach that can be used to improve future models of wildfire spread.

References

Altinas, I., et al. "Towards Data-Driven Operational Wildfire Spread Modeling."

Coen, Janice L., and Wilfrid Schroeder. "Use of spatially refined satellite remote sensing fire detection data to initialize and evaluate coupled weather‐wildfire growth model simulations." *Geophysical Research Letters* 40.20 (2013): 5536-5541.

Gollner, M., et al. "Towards Data-Driven Operational Wildfire Spread Modelling." *Report of the NSF-Funded Wildfire Workshop*. 2015.

Mandel, Jan, et al. "A wildland fire model with data assimilation." *Mathematics and Computers in Simulation* 79.3 (2008): 584-606.

Ntaimo, Lewis, Xiaolin Hu, and Yi Sun. "DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment." *Simulation* 84.4 (2008): 137-155.

Petrasova, Anna, et al. "Wildfire Spread Simulation." *Tangible Modeling with Open Source GIS*. Springer, Cham, 2015. 105-113.

Rothermel, Richard C. "A mathematical model for predicting fire spread in wildland fuels." (1972).

Scott, Joe H., and Robert E. Burgan. "Standard fire behavior fuel models: a comprehensive set for use with Rothermel's surface fire spread model." (2005).

Appendix I: Rothermel Parameters

Summary of Basic Fire Spread Equations

$$R = \frac{I_R \xi (1 + \phi_w + \phi_s)}{\rho_b \varepsilon Q_{ig}}$$ Rate of spread, ft./min. (52)

$$I_R = \Gamma' w_n h \eta_M \eta_s$$ Reaction intensity, B.t.u./ft.^{2} min. (27)

where:

$$\Gamma' = \Gamma'_{max} (\beta/\beta_{op})^A \exp[A(1-\beta/\beta_{op})]$$ Optimum reaction velocity, min.$^{-1}$ (38)

$$\Gamma'_{max} = \sigma^{1.5}(495 + .0594\sigma^{1.5})^{-1}$$ Maximum reaction velocity, min.$^{-1}$ (36)

$$\beta_{op} = 3.348\sigma^{-.8189}$$ Optimum packing ratio (37)

$$A = 1/(4.774\sigma^{.1} - 7.27)$$ (39)

$$\eta_M = 1 - 2.59 \frac{M_f}{M_x} + 5.11 \left(\frac{M_f}{M_x}\right)^2 - 3.52 \left(\frac{M_f}{M_x}\right)^3$$ Moisture damping coefficient (29)

$$\eta_s = 0.174\ S_e^{-.19}$$ Mineral damping coefficient (30)

$$\xi = (192 + 0.2595\sigma)^{-1} \exp[(0.792 + 0.681\sigma^{.5})(\beta + 0.1)]$$ Propagating flux ratio (42)

$$\phi_w = C U^B \left(\frac{\beta}{\beta_{op}}\right)^{-E}$$ Wind coefficient (47)

$$C = 7.47 \exp(-0.133\sigma^{.55})$$ (48)

$$B = 0.02526\sigma^{.54}$$ (49)

$$E = 0.715 \exp(-3.59 \times 10^{-4}\sigma)$$ (50)

$$W_n = \frac{w_o}{1 + S_T}$$ Net fuel loading, lb./ft^2 (24)

$$\phi_s = 5.275\ \beta^{-.3}(\tan \phi)^2$$ Slope factor (51)

$$\rho_b = w_o/\delta$$ Ovendry bulk density, lb./ft.^{3} (40)

$$\varepsilon = \exp(-138/\sigma)$$ Effective heating number (14)

$$Q_{ig} = 250 + 1{,}116\ M_f$$ Heat of preignition, $\frac{\text{B.t.u.}}{\text{lb.}}$ (12)

$$\beta = \frac{\rho_b}{\rho_p}$$ Packing ratio (31)

Input Parameters for Basic Equations

w_o, ovendry fuel loading, lb./ft.^{2}

δ, fuel depth, ft.

σ, fuel particle surface-area-to-volume ratio, 1/ft.

h, fuel particle low heat content, B.t.u./lb.

ρ_p, ovendry particle density, lb./ft.^{3}

M_f, fuel particle moisture content, $\frac{\text{lb. moisture}}{\text{lb. ovendry wood}}$

S_T, fuel particle total mineral content, $\frac{\text{lb. minerals}}{\text{lb. ovendry wood}}$

S_e, fuel particle effective mineral content, $\frac{\text{lb. silica-free minerals}}{\text{lb. ovendry wood}}$

U, wind velocity at midflame height, ft./min.

$\tan \phi$, slope, vertical rise/horizontal distance

M_x, moisture content of extinction. This term needs experimental determination. We are presently using 0.30, the fiber saturation point of many dead fuels. For aerial fuels (β <.02) with low wind velocity (<5 m.p.h.) $M_x \simeq 0.15$.

Appendix II: Standard Fire Behavior Fuel Model Parameters

Table 7—Fuel model parameters.

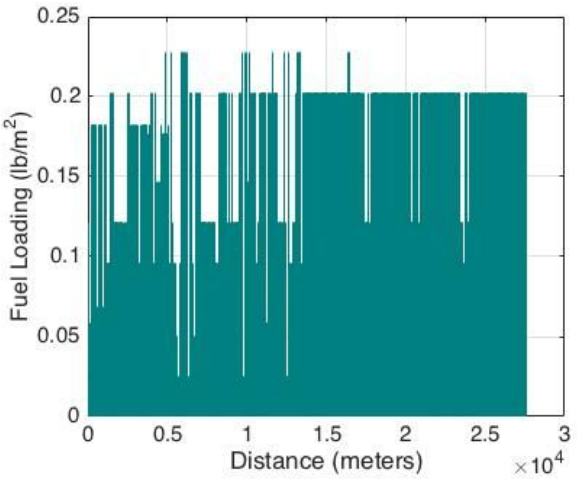
Fuel model code	Fuel load (t/ac)					Fuel model type[a]	SAV ratio (1/ft)[b]			Fuel bed depth (ft)	Dead fuel extinction moisture (percent)	Heat content BTU/lb)[c]
	1-hr	10-hr	100-hr	Live herb	Live woody		Dead 1-hr	Live herb	Live woody			
GR1	0.10	0.00	0.00	0.30	0.00	dynamic	2200	2000	9999	0.4	15	8000
GR2	0.10	0.00	0.00	1.00	0.00	dynamic	2000	1800	9999	1.0	15	8000
GR3	0.10	0.40	0.00	1.50	0.00	dynamic	1500	1300	9999	2.0	30	8000
GR4	0.25	0.00	0.00	1.90	0.00	dynamic	2000	1800	9999	2.0	15	8000
GR5	0.40	0.00	0.00	2.50	0.00	dynamic	1800	1600	9999	1.5	40	8000
GR6	0.10	0.00	0.00	3.40	0.00	dynamic	2200	2000	9999	1.5	40	9000
GR7	1.00	0.00	0.00	5.40	0.00	dynamic	2000	1800	9999	3.0	15	8000
GR8	0.50	1.00	0.00	7.30	0.00	dynamic	1500	1300	9999	4.0	30	8000
GR9	1.00	1.00	0.00	9.00	0.00	dynamic	1800	1600	9999	5.0	40	8000
GS1	0.20	0.00	0.00	0.50	0.65	dynamic	2000	1800	1800	0.9	15	8000
GS2	0.50	0.50	0.00	0.60	1.00	dynamic	2000	1800	1800	1.5	15	8000
GS3	0.30	0.25	0.00	1.45	1.25	dynamic	1800	1600	1600	1.8	40	8000
GS4	1.90	0.30	0.10	3.40	7.10	dynamic	1800	1600	1600	2.1	40	8000
SH1	0.25	0.25	0.00	0.15	1.30	dynamic	2000	1800	1600	1.0	15	8000
SH2	1.35	2.40	0.75	0.00	3.85	N/A	2000	9999	1600	1.0	15	8000
SH3	0.45	3.00	0.00	0.00	6.20	N/A	1600	9999	1400	2.4	40	8000
SH4	0.85	1.15	0.20	0.00	2.55	N/A	2000	1800	1600	3.0	30	8000
SH5	3.60	2.10	0.00	0.00	2.90	N/A	750	9999	1600	6.0	15	8000
SH6	2.90	1.45	0.00	0.00	1.40	N/A	750	9999	1600	2.0	30	8000
SH7	3.50	5.30	2.20	0.00	3.40	N/A	750	9999	1600	6.0	15	8000
SH8	2.05	3.40	0.85	0.00	4.35	N/A	750	9999	1600	3.0	40	8000
SH9	4.50	2.45	0.00	1.55	7.00	dynamic	750	1800	1500	4.4	40	8000
TU1	0.20	0.90	1.50	0.20	0.90	dynamic	2000	1800	1600	0.6	20	8000
TU2	0.95	1.80	1.25	0.00	0.20	N/A	2000	9999	1600	1.0	30	8000
TU3	1.10	0.15	0.25	0.65	1.10	dynamic	1800	1600	1400	1.3	30	8000
TU4	4.50	0.00	0.00	0.00	2.00	N/A	2300	9999	2000	0.5	12	8000
TU5	4.00	4.00	3.00	0.00	3.00	N/A	1500	9999	750	1.0	25	8000
TL1	1.00	2.20	3.60	0.00	0.00	N/A	2000	9999	9999	0.2	30	8000
TL2	1.40	2.30	2.20	0.00	0.00	N/A	2000	9999	9999	0.2	25	8000
TL3	0.50	2.20	2.80	0.00	0.00	N/A	2000	9999	9999	0.3	20	8000
TL4	0.50	1.50	4.20	0.00	0.00	N/A	2000	9999	9999	0.4	25	8000
TL5	1.15	2.50	4.40	0.00	0.00	N/A	2000	9999	1600	0.6	25	8000
TL6	2.40	1.20	1.20	0.00	0.00	N/A	2000	9999	9999	0.3	25	8000
TL7	0.30	1.40	8.10	0.00	0.00	N/A	2000	9999	9999	0.4	25	8000
TL8	5.80	1.40	1.10	0.00	0.00	N/A	1800	9999	9999	0.3	35	8000
TL9	6.65	3.30	4.15	0.00	0.00	N/A	1800	9999	1600	0.6	35	8000
SB1	1.50	3.00	11.00	0.00	0.00	N/A	2000	9999	9999	1.0	25	8000
SB2	4.50	4.25	4.00	0.00	0.00	N/A	2000	9999	9999	1.0	25	8000
SB3	5.50	2.75	3.00	0.00	0.00	N/A	2000	9999	9999	1.2	25	8000
SB4	5.25	3.50	5.25	0.00	0.00	N/A	2000	9999	9999	2.7	25	8000

[a] Fuel model type does not apply to fuel models without live herbaceous load.
[b] The value 9999 was assigned in cases where there is no load in a particular fuel class or category
[c] The same heat content value was applied to both live and dead fuel categories.

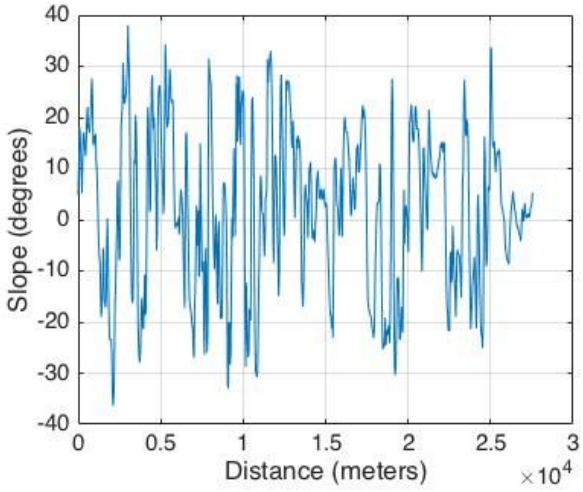
Source: Scott, Joe H., and Robert E. Burgan. "Standard fire behavior fuel models: a comprehensive set for use with Rothermel's surface fire spread model." (2005).

Appendix III: Fire Model Input Data

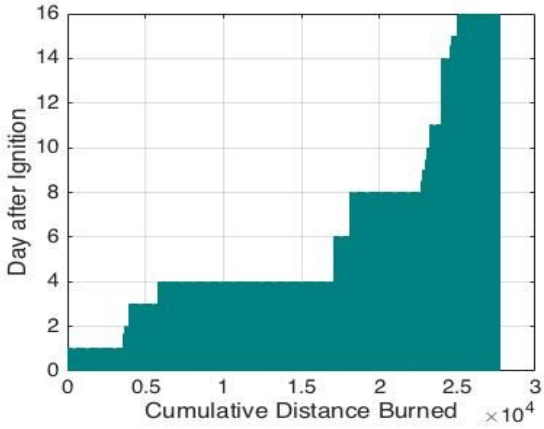
Fuel Loading



Slope



Actual Burn Record



Wind (Northern projection)