# The Design and Development of a Microgrid Energy Storage System

**Capstone 2 - Final Project Report**
**Florida Polytechnic University**
**Spring 2024**

*submitted by:*
Energy Storage Optimization
Walt Ward – ME
Zachary Kaiser - ME
Noah Al-Shaer - ME
Caleb Smith – ME
James Shetter – EE
Xaviar Avalos – CS
Michael Tracey – BA
Sebastian Lopez – CS

*submitted to:*
Matt Bohm
5/1/24

## 1. Introduction

As renewable energy and electric vehicles increase in popularity, the electric grid needs to become more efficient and resilient. Lockheed Martin has developed the GridStar Flow, a microgrid solution addressing increased energy demand. A microgrid is a stand-alone electrical grid that serves as an intermediary between production sites (the grid, reserves, renewables) and efficiently dispatches energy to minimize consumer electrical costs. This energy storage system (ESS) developed by Lockheed Martin will serve as a smart battery capable of meeting resiliency, ancillary, and arbitration needs of the consumer.

The purpose of this project is to develop the smart component of this battery; namely the development of the system capable of selling and purchasing to and from the electrical grid to reduce consumer energy costs. To achieve this, the system will require a working forecast, optimizer, and battery scheduler. This is a complex problem, involving purchasing from four sources- grid, regulating reserves, supplemental reserves, and spinning reserves- as well as the option to insure reserve sources during on-peak or off-peak time periods. The ESS must account for efficiency changes on charge and discharge, unavailability in the charging sources or the battery itself, and the purchasing and sale options. The controller, which is assumed to be for a single microgrid node, will need to manage the aforementioned tasks in the 1-hour sample rate required for management.

At the start of the development process, design constraints were identified by communication with the sponsor and relevant technical research. These results were used to identify the customer needs, design criteria, and possible solution paths. Solution paths were evaluated with weighted criteria and tested with an alpha prototype and a couple proof-of-concept programs. These steps were eventually combined into a final-beta prototype.

To achieve this purpose, an alpha optimization program was developed that handled energy arbitrage without considering insuring the reserves. From this result it was determined that the chosen environment for optimization, MATLAB, was a solid choice for the project. Additionally, it was determined that the program was well within the 1-hour period required by the project. Moving to forecasting, data analysis was performed on the dataset provided by the sponsor to identify price and unavailability trends. After performing autoregressive analysis and basic statistical analysis of the data, the team developed a linear regression forecast to capture the behavior of the data. Additionally, a boosted trees ensemble classifier was developed to identify unavailable times for the grid, reserves, and battery system. To handle the uncertainties native in the forecast, a robust optimization solution was utilized. This approach generated conservative price estimates to mitigate risk in scheduling decisions. The final optimization problem- a deterministic, mixed integer process- could be solved directly in MATLAB using the optimization toolbox.

The final, combined solution meets the customer needs and the constraints given by the sponsor. Monte-Carlo results suggest that the system can perform well, making approximately $1,900 during the month-long simulation. However, though the controller works well, the team identified areas of improvement in the forecasting methodology and the possibility of using a stochastic process in the optimization script for the post 24-hour horizon.

## 2. Customer Needs, Objectives and Team Interpretation

Between information from the project sponsor and internal research on microgrids and the GridStar system, the group developed a set of customer needs and design objectives for the microgrid system. Confirmed with targeted questions, a set of assumptions were made on the battery constraints, intended purpose, and desired outcomes. The team used this information to develop a set of battery specifications which will serve as operational constraints for the optimizer of the project.

*2.1 Customer Needs*

Throughout the project, the sponsor, via the project description and bi-weekly meetings, discussed the requirements of the project. The requirements included forecast and optimization script for an energy storage system with the goal to maximize profit. Specifically, the sponsor wanted a program capable of creating an accurate single day forecast of electrical prices for the grid and three reserve sources and optimizing charging and discharging of the battery system. A more detailed customer needs list can be found in Appendix 1a.

*2.2 Design Objectives*

Derived from the customer needs and additional research, the battery's purpose and specifications were identified and used to develop design objectives. First, the battery must be able to operate under the 1-hour sample rate of the grid input to stay on pace. Second, the state of charge (SOC) of the battery must be completely accurate, even when the efficiency changes between samples (due to the SOC's effect on efficiency). Currently the forecasting accuracy as denoted by our sponsor is not the most important aspect of the project. An allowable tolerance of 50% is acceptable if the optimizer can account for these error margins (Appendix 1b).

*2.3 Team Interpretation*

Considering the project requirements, the optimizer should be capable of responding to hourly sample rates, predicting no-charge periods, adjusting to forecast changes, and variable resiliency reserves. For the optimization itself, it must be able to create a charge and discharge cycle quickly and accurately for the ESS: handling purchase and sale to all sources and taking round-trip efficiency into account. The systems need to be able to handle 24hrs of scheduled data at a time to account for the time it takes to receive real price data. Since we do not get price information until 24hrs have passed, the ability to dynamically adapt to changes has been deemed out of scope.

*2.4 Functional and Non-functional Requirements*

The beta prototype had the functional requirement to predict the best charge or discharge times based on price values imported from a dataset sent from the sponsor. Additionally, it has the functional requirement of meeting charge rate, storage limit, and efficiency constraints provided by the battery (Appendix 1c). The non-functional requirements state that the minimum forecast size must be 24 hours from the current time. This requires a forecast horizon of at least 24hrs and requires the functionality of handling wrong predictions logically.

*2.5 Limitations*

When designing the forecasting and optimization script, a few assumptions have been made by the team and sponsors. For the battery, the microgrid system will be a single node battery with a 1 MW power capacity and 10 MWh storage capacity. Efficiencies for the charge and discharge will vary based on the state of charge of the system. For the dataset, the prices will be assumed to be constant across the 1-hour sample time. For instance, if 1:00 AM shows grid price to be $8/MWh, it will remain at this price till 2:00 AM. More information on the assumptions used for the battery's design and reason behind assumptions can be found in Appendix 1c. As mentioned above the team is working with the limitation of not knowing electrical prices until after 24hrs have passed, meaning that the optimized schedule will run completely on unknown values without the ability to check until after the 24hrs have passed.

## 3. **Concept Generation and Analysis of Alternatives**

Following standard practices for concept generation and analysis, the team generated option alternatives for software, forecast method, forecast algorithms, and methods for dealing with uncertainty within optimization. These concepts were identified using research of similar product solutions and advice from experienced faculty members at Florida Polytechnic University. Once identified, these concepts were weighted using design criteria on a decision matrix and will be used in the product moving forward.

### *3.1 Literature Search*

The initial literature search attempted to identify design options commonly used for microgrid forecasting. Nine primary sources were identified and summarized in Appendix 2. These sources compared forecasting techniques, results and optimization techniques to deal with uncertainty. Two sources utilized MATLAB/Simulink (and pre-built toolboxes) to perform forecast analysis. Four sources utilized uncertainty.

### *3.2 Concept Generation*

For concept generation we mainly focused on the environment that would be able to perform both the forecasting and optimization tasks for the project. From initial research and faculty discussions the team decided to evaluate three major options: Python, MATLAB/Simulink, and GAM. Once identified, these resources were considered with a decision matrix shown in Appendix 2b with MATLAB/Simulink showing benefit over the other two options. From here, the team had to decide on the predictive and optimization functions that could be included. For forecasting, these were typically categorized as either statistical, computational, and hybrid models. From this list, the team identified two statistical methods- exponential smoothing and linear regression, one computational method – autoregressive models-, and one hybrid solution – boosted tree ensembles for investigation. However, due to time constraints, there was no additional time to compare the efficacy of the models for the given dataset.

### *3.3 Analysis of Alternatives*

The team evaluated 3 categories of integral requirements to the project: first, the simulation environment, and next, forecasting methodology and forecasting algorithms. Each category was weighted on 7 criteria referenced in Appendix 2b. Analysis of the forecasting methodology led to the conclusion that the methodologies could not be differentiated from one another. Through verification and meetings with faculty, it was found that the linear regression model was the most feasible of the solution paths and had the highest run speed, which was beneficial for the optimizer's high computation time. Furthermore, due to time constraints further validation could not be done to compare the working linear regression model with other models.

### *3.4 Flow Chart of Design Process*

When operating the team has generally following the flow in Figure 1. As seen, after the project concept was fully identified, the team evaluated ideas and concepts based on our customer needs statement. Every two weeks, progress is discussed with the sponsor and corrections are made, as necessary. This process continued until the beta prototype was completed.

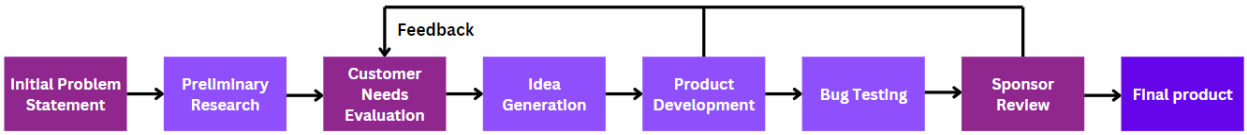


**Figure 1: Project Design Procedure**

*3.5 Functional Decomposition*

The function decomposition shows the breakdown of each component of our model (Figure 2). The basic flow starts from our 6-year hourly dataset from Lockheed Martin which contains grid pricing, reserve pricing, and whether the battery can charge. This information will flow into the predictive algorithm: generating a 24-hour forecast, where the model is retrained every hour and updated every 24 hours. The forecast results will be used to optimize the function, while accounting for uncertainty, of the GridStar system and generate a schedule for the maximization of profit with a high-low analysis for the reserves.

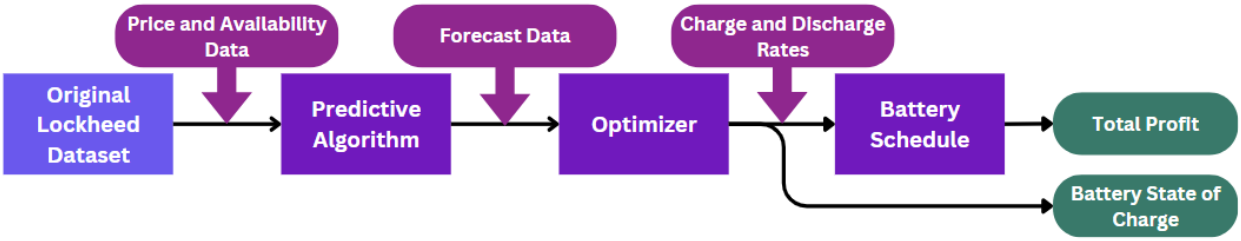


**Figure 2: Functional Model Decompositions**

## 4.0 Prototyping Process

Optimization, as reinforced by the sponsor, is the primary point of importance in the project. As such all prototypes have been centered around the development of a solid optimizer with secondary systems (forecast and check node) being developed alongside these systems. The alpha prototype was the first test of the optimization toolbox in MATLAB and handles perfect forecast situations (Appendix 3a). Developed next, the forecast (Appendix 3b) and robust optimizer (Appendix 3c) were components of the beta prototype used in the final program (Appendix 3c), which was a cumulation of all previous subsystems.

*4.1 Proof of Concepts*

During the design process, there were two primary proof-of-concept tasks completed. The first component, the Alpha Prototype, was used to demonstrate the efficacy of MATLAB in solving optimization problems. This system used a simplified version of the project constraints and assumed that sale to reserve function just like sale to the grid (Appendix 3a). The results demonstrated a logical schedule at a reasonable operating speed for optimization: though the resulting optimization refused to "hold" at any specific state of charge which was considered a bug.

With a partially functional optimizer, it was then necessary to feed it forecast information to make a "real world" prediction. Covered in detail in (Appendix 3b), the team decided to test on a linear regression model to analyze the contribution of the training data inputs in the forecast. The model used autocorrelation values to predict which lag prices were relevant in the prediction as well as average statistics to generate a prediction formula, seen below.

$$\hat{Q}_{t+n} = Q_t - (Q_{t-n} - Q_{t-24}) + \bar{Q}_{DoW} + \bar{Q}_{MoY} \quad (1)$$

The resulting linear model had reasonable accuracy and fit, but failed to predict price spikes which had a large effect on its accuracy and reliability. Seen in Figure 3, the system does generally fit the data distribution but responds to the spikes the day after they occur. Though this was designed as a test or reference point in the forecast development process, time constraints prevented the development of a more robust model for price forecasting. More details on the system design and reasoning can be found in (Appendix 3b).

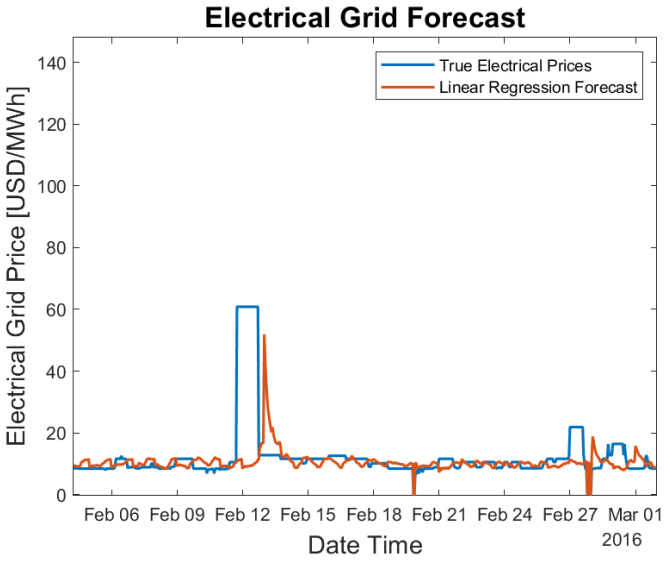


**Figure 3: Linear Regression Forecast**

*4.2 Design Details*

To convert the proof-of-concepts to finalized subsystem components, some modifications needed to be made. First, the forecast needed to be converted into a function that ran on a moving window. This was done by generating MATLAB functions that individually handled each subsystem's process in discrete steps and strung them together in a centralized loop for testing and debugging. Next, alongside this development, functions were re-formatted, condensed, and tweaked as necessary to achieve unaccounted for goals or increase the speed of the simulation. At this stage, the following major modifications were implemented:

1. A moving window component was incorporated into the forecast. This kept training data relevant to current events and helped reduce training time slightly.
2. A new optimization technique was used: robust optimization. The accounted for forecast uncertainty and removed the reliance on a perfect forecast (Appendix 3c).
3. The optimization was changed from a continuous problem to a discrete problem. This required the use of a mixed integer linear programming (MILP) solution to evaluate the optimal schedule.
4. The check node processing was modified to handle incorrect guesses. Namely if SOC left the expected region due to unavailable charge times (Appendix 3c).

As stated above, the optimization itself was changed to run using robust optimization and solved using a MILP process. This modified the objective function slightly – seen below- but it properly incorporated sale to reserves and fixed the last iteration's propensity towards action (Appendix 3c).

$$\min J = \sum_{t=1}^{24} C_{t,i}^{R} \times q_{t,i} - C_{RES} \tag{2}$$

For more details on the optimization process, please read Appendix 3c, as it covers the robust price generation, objective function, additional constraint changes, as well as the updated check node.

*4.3 Beta Prototype*

The final beta-prototype came a long way from the alpha. First and foremost, the beta included all customer requirements in the result: forecast, optimizer, and battery scheduler. This came in the form of a linear regression model, a robust optimization process, and a real-time check node for the simulation. Second, the optimization process itself changed. Uncertainty and sale to reserves incorporated previously unaccounted for components of the project. Finally, the entire system was tied together, and Monte-Carlo analysis was applied to the results (50 scenarios). Summarized in Appendix 3d, the results show that the system makes a significant amount of money in a one-month simulation period of around $1,900. There are still remaining issues from forecast inaccuracies (Figure 4), but with improvements, the beta-prototype would be very beneficial.

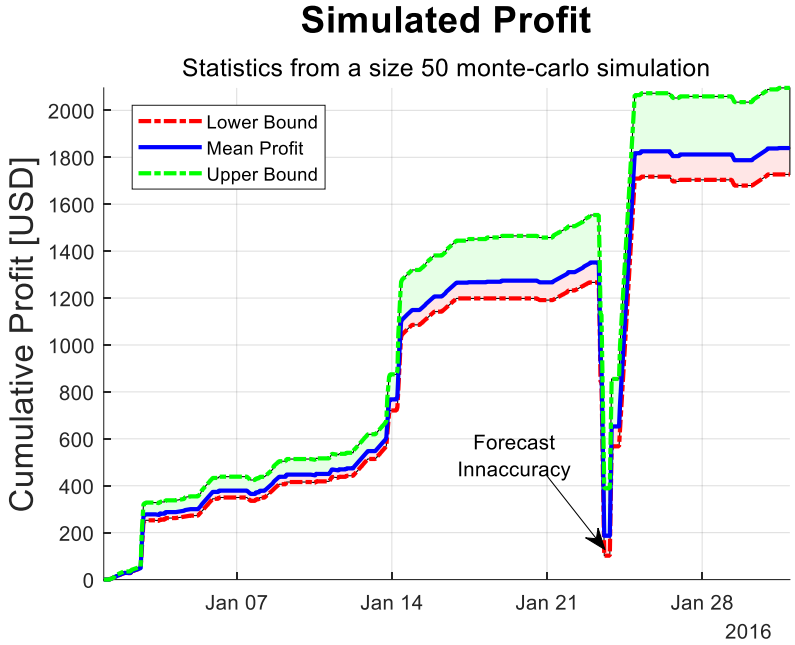


**Figure 4: Simulated Profit in One Month**

## 5. Results

5.1 *Final Product Testing*

Concluding the project with the final day of testing and the capstone showcase, the results successfully met most of the customer's needs and functionality requirements, with varying levels of performance. The final product was a cohesive controller system that combined the required subsystems given by the sponsor (forecast, optimization, check node) and was able to reduce the energy costs of the consumer. Provided with a month-long simulation with 24-hour forecast intervals, the scheduled optimizer was able to show a mean profit value of approximately $1,900 after 50 Monte Carlo simulations. This proved that even with the large errors and issues with the forecast, the optimizer was able to overcome most of the inaccuracies and make profit. Based on interactions with the sponsor during showcase and the debrief meeting afterwards, the sponsor was extremely pleased with the results and was eager to pursue further projects at Florida Polytechnic University.

Overall, the team was able to complete all of the customer needs, with some areas of future improvement which have been identified. In the following section the areas of improvement will be described to enhance future renditions of the project.

## 6. Conclusions and Future Work [All Disciplines]

By the end of the project, the team was able to complete- to a satisfactory level- all of the customer needs. Though improvements can be made to the project and the organization of the team, the final product should be beneficial to the sponsor and the extended community.

### *6.1 Conclusions and Project Summary*

Cumulatively, the customer needs were met to a reasonable degree. Though lacking, the forecast was able to capture the general trends in the data, the optimizer was able to take an imperfect forecast and generate profit over a 1-month period, and the check-node was able to handle off-cases and keep energy within the operational constraints of the system. Additionally, the team put in place a framework for the further development of the control system: hopefully, creating a significantly more efficient and accurate scheduling process.

### *6.2 Future Work*

It became clear during the development that two additional modifications would move the results from good results to great results. First, one major handicap for the optimization script was the inaccuracies in the forecast. The current model is able to predict trends to some degree but is completely unable to predict spikes in the data: instead responding 24 hours late. Improving the power of the forecast with more complete data analysis and computational forecasting methods would greatly benefit the current system ( Appendix 4a).

In addition, the current optimization targets a set charge value for every optimization. This method does function as a form of "preparing" the battery for the next day but does not truly capture the next 24 hours properly. As such, the team recommends incorporating a two-stage or multi-stage stochastic optimization process for times after the 24-hour horizon (Appendix 4b).

### *6.3 Reflection Report*

Starting from scratch, the design approach would be significantly different. First, instead of focusing solely on industry research, a mixed research and interview approach would be necessary. The goal of this change would be to take advantage of experience-derived knowledge of experienced individuals: possibly giving greater context to the problem. This would have given the team more time to evaluate solutions instead of re-evaluating the problem multiple times.

Further, more inter-team communication and honesty might have improved the overall state of the project. For example, member performance should have been evaluated weekly, with constructive critiques given in real time to help improve individual productivity. Additionally, in the case that members refuse to meet work expectations, stricter action could have been carried out earlier in the process.

However, there were some successful portions of the project as well. Once the scope was pinned down, the back-and-forth meetings with the sponsor and knowledgeable experts such as Dr. Dewey greatly helped the team gather context to a complete solution path: including aspects of the solution which could not be completed this semester. Additionally, having three short meetings a week (in class or directly after) had the benefit of keeping members on the same page and allowing for frequent brainstorming sessions. This same concept applies to the sponsor meetings, held every two weeks, which allowed the team to evaluate design choices, ask questions, and extend the scope of the problem.

*6.4 Broader Impacts.*
This product will have a great positive impact on the community. If the GridStar is installed with a smart algorithm, the system will serve to reduce electrical costs in its impact region. Additionally, if this concept is used on a larger scale, the electrical grid will have less supply issues which might decrease the stress on energy productions: eventually balancing out the grid system.

Moreover, this system has environmental benefits compared to other ESS alternatives. The fluids it uses to store energy are non-toxic and, in the case of a spill, will cause no harm to the environment. This makes it especially useful in regions with harsh natural conditions, such as earthquake or hurricane prone areas.

## 7. Individual Contributions

Based on the contributions to the ideation, research, and development of the project. The following contribution rates have been assigned and agreed upon by the capstone team (Table 1). More weight is considered in the later portion of the semester and contributions to writing, presentations, and showcase are weighted.

**Table 1: Team Contribution**

| Team Member | | Contribution |
|---|---|---|
| 1. | Walt Ward | 50.00% |
| 2. | Noah Al-Shaer | 16.50% |
| 3. | Zachary Kaiser | 15.50% |
| 4. | Caleb Smith | 7.25% |
| 5. | James Shetter | 5.75% |
| 6. | Xaviar Avalos | 3.00% |
| 7. | Sebastian Lopez | 1.50% |
| 8. | Michael Tracey | 0.50% |

## References

[1] H. Zareipour, C. A. Canizares and K. Bhattacharya, "Economic Impact of Electricity Market Price Forecasting Errors: A Demand-Side Analysis," in IEEE Transactions on Power Systems, vol. 25, no. 1, pp. 254-262, Feb. 2010, doi: 10.1109/TPWRS.2009.2030380.

[2] X. Luo, X. Zhu, and E. G. Lim, "A hybrid model for short term real-time electricity price forecasting in smart grid - big data analytics," BioMed Central, https://bdataanalytics.biomedcentral.com/articles/10.1186/s41044-018-0036-x (accessed Dec. 7, 2023).

[3] F. Otieno, N. Williams, and P. McSharry, "Forecasting Energy Demand For Microgrids Over Multiple Horizons," 2018 IEEE PES/IAS PowerAfrica. (accessed Dec. 9, 2023)

[4] lIbrahim A. Bello a et al., "A linear regression data compression algorithm for an islanded DC Microgrid," Sustainable Energy, Grids and Networks, https://www.sciencedirect.com/science/article/pii/S2352467722001540 (accessed Dec. 10, 2023).

[5] B. P. Ganthia et al., "Artificial neural network optimized load forecasting of Smartgrid using MATLAB," Control Systems and Optimization Letters, https://ejournal.csol.or.id/index.php/csol/article/view/14/16 (accessed Dec. 9, 2023).

[6] H. -T. Zhang, F. -Y. Xu and L. Zhou, "Artificial neural network for load forecasting in smart grid," 2010 International Conference on Machine Learning and Cybernetics, Qingdao, China, 2010, pp. 3200-3205 (accessed Dec. 9, 2023).

[7] Alharbi, H., & Bhattacharya, K. (2018). Stochastic optimal planning of battery energy storage systems for isolated microgrids. *IEEE Transactions on Sustainable Energy*, *9*(1), 211–227. https://doi.org/10.1109/TSTE.2017.2724514

[8] Vergara, P. P., López, J. C., Rider, M. J., Shaker, H. R., da Silva, L. C. P., & Jørgensen, B. N. (2020). A stochastic programming model for the optimal operation of unbalanced three-phase islanded microgrids. *International Journal of Electrical Power and Energy Systems*, *115*. https://doi.org/10.1016/j.ijepes.2019.105446

[9] Ahmed, H. (n.d.). *Formulation of Two-Stage Stochastic Programming with Fixed Recourse*. http://biarjournal.com/index.php/bioex

[10] Giraldo, J. S., Castrillon, J. A., Lopez, J. C., Rider, M. J., & Castro, C. A. (2019). Microgrids Energy Management Using Robust Convex Programming. *IEEE Transactions on Smart Grid*, *10*(4), 4520–4530. https://doi.org/10.1109/TSG.2018.2863049

## Appendix 1 - Customer Needs, Requirements and Objectives

The customer needs, requirements, and objectives of the project have slowly developed based on sponsor meetings, and continued scope research from the team. However, the overarching goal has not changed: maximize customer profit but reducing energy purchased during expensive (on-peak) hours and selling excess power to the grid and reserves. To achieve this, the team was tasked with the development of a Microgrid controller that will forecast grid and reserve prices, optimize charge and discharge times, and schedule the energy storage system (Figure 5).

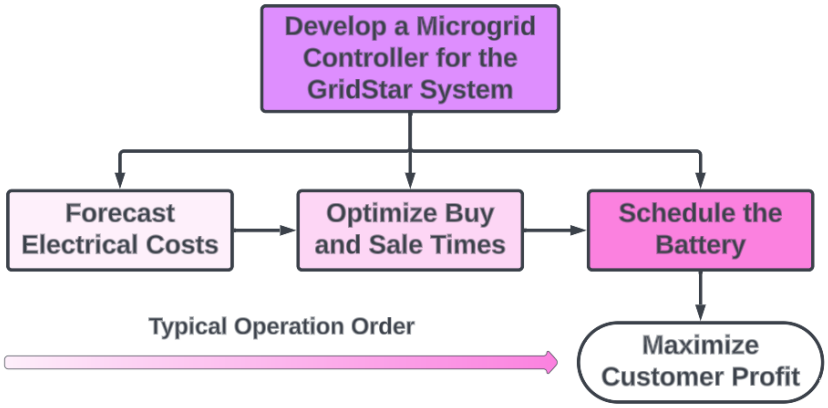


**Figure 5: Basic Project Operation Order**

### *Appendix 1a – Customer Needs*

The customer needs can be separated into three subsystems: forecast, optimization, and battery. These requirements are further separated into one of three categories. Primary (P) tasks are tasks that are a requirement of the project for it to be considered complete. Secondary (S) tasks that the sponsor has expressed interest in meeting if time or resources allow. Finally, Latent (L) needs are needs that the sponsor did not explicitly state to the team but have been identified as points of interest.

Forecast:

For the forecast, the customer requirements are the following:

- [P] The forecast shall be able to predict single-day prices for grid and reserve pricing.
  - Though the forecast might want to predict more than 24 hours ahead for more context to the optimizer, 24 hours is the value set by the sponsor as a minimum.
- [P] The forecast shall be able to react to dynamic changes in the price.
  - There are times when energy spikes during the week. The forecast should be able to predict this based on trends in the data. In the future, additional factors may be used to predict this, such as high temperatures or bad weather (secondary requirement).
- [P] The forecast shall be able to predict when charging is not possible for the battery.
  - There are times during the day where, for various reasons, the battery is unable to charge from any source- despite the demand levels of the grid. The forecast must be able to predict this, so the optimizer does not discharge all reserves right before one of these times.
- [S] The forecast should be able to incorporate temperature and weather data into the forecast.
  - Temperature and weather can have an impact on grid demand, and as a result, grid pricing. Knowing the current temperature and weather conditions as well as using online forecasts, can improve the accuracy of the forecast system.

- [S] The forecast should be able to adapt to location changes.
    - Currently, the system is being developed under the assumption that it will be used in the northwest United States. However, different regions in this area will operate slightly differently. A forecast that can adapt to these differences is preferable.

Optimization:
For the optimization, the customer requirements are the following:

- [P] The optimization shall be able to react at the hourly sample rate.
    - The grid pricing is sampled every 1 hour. The system must be able to react at this pace. Otherwise, the optimizer will fall behind.
- [P] The optimization shall reduce energy costs for consumers.
    - The optimization will manage the purchase and sale of electricity. It will save money by either holding cheap energy for use during expensive hours or selling the electricity back to the grid during peak times.
- [S] The optimization should minimize stagnant energy.
    - Holding energy can be beneficial when there will be a large price spike in the near future. However, outside of these situations, the system should attempt to be in constant use to maximize the total profit of the system.
- [S] Resiliency should be included in the optimization.
    - There should be a setting for consumers to set energy that should not be touched by the system for sale or use. It will hold onto the energy until specified by the user.

Battery:

- [S] The battery subsystem should be able to display live information from the forecast, the state of charge, current charge rate, and other relevant information.
    - Data visualization provides a good demonstration of how the product is currently functioning. This can be used for troubleshooting or presentations.
- [L] The battery system might need to specifically conform to grid regulations.
    - The grid requires certain voltage and frequency levels when purchasing and selling to and from the system. The battery system may have systems in place to manage this already, but it has not yet been confirmed.

***Appendix 1b – Design Requirements***

Some functional requirements and design objectives have been provided by the sponsor. Though specific error percentages were not provided by the sponsor, the team used literature reviews as a reference point for the required system accuracy.

First, the forecast should be able to predict reserve and grid prices 24 hours in advance. The accuracy of the forecast should be within reasonable tolerance but high accuracy in the forecast is not the primary directive of the project. Since the sponsor wanted the team to focus into optimization and this component had greater influence on the model's performance, the forecast only needs to generally match the dataset trends. Though the price forecast did not need to be incredibly accurate, this cannot be said for the unavailability prediction as this had a significantly larger effect on the optimization.

The optimizer's requirements were more of a challenge due to the complexity of the system. First, the system must be able to handle purchasing and sale to and from the grid and three reserves sources: regulating, supplemental, and spinning reserves. This must include the unique requirements for reserve sales where it functions more like an insurance period. This involved handling efficiency

differences dependent upon the SOC of the battery and insurance fulfillment chances (chance that insuring to a reserve would result in energy being pulled). Energy cannot be charged or discharged during the insurance period.

The final optimizer must be capable of meeting the customer requirement of reducing consumer energy costs. This involves, by extension, the ability to handle uncertainty in the forecast and mitigate risks in the scheduling process. The final output from this system must be checked to determine its viability before carrying out an action.

***Appendix 1c – Design Limitations and Assumptions***

When performing research on the GridStar Flow system, the battery specifications, applications, and limitations were identified. The battery can have a range of specifications, however the team made, and received approval for, a set of assumptions for the battery to operate at and how grid information is to be handled.

Battery Specifications:

The battery itself can have multiple storage and power capacities, however, the team will be designing under the assumption that the battery is a single node microgrid system. Below are the assumptions made for the system and the reasoning behind each.

- The battery is a single node system, meaning it cannot charge and discharge at the same time.
  - This specification was provided by the sponsor for the ESS.
- The battery has a storage capacity of 10 MWh and power output capacity of 1 MW.
  - These specifications were pulled from an article on Lockheed's website on a battery system installed at a Colorado army facility.
- The battery has a depth-of-discharge (DoD) of 100% at a design life of 20 years.
  - The informational page on Lockheed's website for the GridStar system is capable of 100% DoD- or amount the battery can discharge of its total storage- and is designed for 20-years. The sponsor confirmed that the battery is designed to handle this level of discharge and any harm it may receive is already considered in the 20-year life.
- The battery has differing operating efficiencies based on its state of charge.
  - This specification was provided by the sponsor.
  - The battery will have a 70% efficiency from 0 – 20% charge.
  - The battery will have an 80% efficiency from 20% - 80% charge.
  - The battery will have a 60% efficiency from 80% -100% charge.
- The battery is aware of its state of charge at any given time.
  - This should be a normal statistic gathered by the ESS and is necessary to make effective decisions without the precise knowledge of availability.

Dataset Information:

The dataset itself has some idealized components by nature. This is partially due to the sample rate, but additional assumptions are made regarding how reserves react to purchase and sale. Below are the assumptions made for the system and the reasoning behind each.

- The prices between the sample times (hour sample rate) will remain the same.
  - This assumption is common among microgrid systems due to the processing requirements of forecasting the information. The sponsor has set this as a rule for the current optimization.
- Reserve will be able to react instantly and in full capacity.

    - Reserves have a ramp-up time before they can sell power to the consumer. However, the sponsor specified that instant reaction times will be assumed. A framework is in place to handle the ramp-up time if this assumption is reversed.
- Reserves have a set change of pulling power from the consumer.
    - Lockheed has specified the percentage change that selling power to a reserve will pull power.
    - Regulating: 30%. Spinning: 10%. Supplemental: 10%.
    - Handling random change is difficult for an optimization, so it must be able to handle outputting maximum and minimum input capacities.
- Sale to reserves can be scheduled at the initiation of the sale hour.
    - When selling to reserves for the on-peak (hours 8-23) or off-peak times (1-7, 24) the decision can be made as early as the start of the reserve period.
    - This should be done in case energy storage quantities do not match expectations at the start of the insurance period.

Additional Constraints:
The project itself has some additional limiting factors due to the nature of the dataset provided by Lockheed Martin. The data itself only contained time and date, price information, and whether the battery could charge. It in no way specified current demand, how much energy was drawn by the consumer, voltage, frequency, temperature, weather, or any other identifying information that could be used to train a forecast to meet the secondary and latent requirements.

## Appendix 2 - Concepts and Analysis of Alternatives

### *Appendix 2a – Existing Products*

Article 1**:**
In research by Xing Luo [2], the researchers designed a short-term, real-time electricity price forecast for a smart grid system. The algorithm generated a Real Time Price (RTP) forecast for the system using least squares analysis, grey prediction model, and an artificial neural network. Cumulatively, the forecast creates a hybrid system where the least squares fitting model handles linear data, the grey prediction model handles non-linear data, and the neural network is used for error analysis and optimization. The results were analyzed using mean absolute percent error (MAPE) and had an approximate error of 3.5% which "significantly outperforms existing models" [2].

Article 2**:**
In research by Fred Otieno [3], the researchers developed an electricity demand forecast for microgrids in Sub-Saharan Africa. Here the researchers develop specifically to optimize the 40+ microgrid nodes set up by PowerGen. The research covers very-short term (a few hours or less), short term (hours to weeks), up to long term (months or years ahead). Data was collected from 7 of these sites from years 2014 to 2017 and found that exponential smoothing methods outperformed autoregressive models at predicting demand and seasonal changes up to four months ahead.

Article 3**:**
In research by Ibrahim A. Bello [4], researchers apply linear regression model to compress as a technique for data compression in an islanded microgrid system. The goal is to compress the storage size of voltage and current time-series data in the microgrid to improve data compression. The research found that it was possible to achieve near lossless compression rates of up to 50-to-1.

Article 4:
In research by Bibhu Prasad Ganthias and others [5], researchers developed load prediction for a smart grid in India. The team developed the program using the neural network toolbox in MATLAB/Simulink to optimize the prediction model. The researchers normalized the input and output vectors to improve the performance of back propagation (ANN training technique). The three-layer ANN used was shown to perform well, with an average error of approximately 12%, which higher variance across months.

Article 5:
In research by Hao-Tian Zhang [6], researchers developed a forecast algorithm using an artificial neural network to predict load demand. The system utilized the Neural Network Toolbox in MATLAB as the primary programming tool. The mean squared error of ANNs of sizes 5, 8, 10, 18, 30 were evaluated to estimate the quality of the simulation. Three models of ANN were evaluated with these node variations: TrainOSS, TrainBR, and TrainLM and found similar errors between the three options and neuron size.

Article 6:
In research by Hisham Alharbi and Kankar Bhaattacharya [7], The researchers developed an optimal planning framework for an optimal battery energy storage system (BESS.) In this paper a new representation of the energy diagram of the BESS was used to determine optimal power and energy size and installation information to use as constraints. The model used for the problem was designed to minimize the installation and operational cost and had constraints based on the microgrid used. Furthermore, the paper discussed potential stochastic cases with the stages of the problem being based on the energy used.

Article 7:
In research by Pedro P. Vergara and others [8], a stochastic mixed-integer nonlinear programming model is used for the optimal operation of islanded microgrids with stochastic demands. This paper uses a set of stochastic scenarios to represent load consumption and power generation. For the load consumption, the forecast errors are modeled using a probability distribution using a normal pdf. The optimization model of the paper was made to minimize operational costs for all scenarios provided. The results of the stochastic model were compared with a deterministic model and were found to be cheaper on average by 100 dollars, with an increase in computation times required.

Article 8:
In research by Hashnayne Ahmed [9], the researcher developed a mathematical model using recourse for a two-stage stochastic programming problem. A two-stage model is one where there is a decision made after a choice is already made in the first stage. The mathematical model discusses the stages of the problem in relation to fixed recourse, where fixed recourse implies that the second stage of the problem is deterministic. This paper formulated basic stochastic programming models and classified the variables into fixed constraint variables or discrete random variables.

Article 9:
In research by Juan S. Giraldo and others [10] an energy management was presented for single-phase or balanced three-phase microgrids using robust optimization. Robust optimization was used to deal with uncertainty while still minimizing the objective function. A probability distribution was used to characterize the expected value and the added robust parameter. The robust equivalents of the model has an optimist, average or pessimist scenario for the load generation in comparison to the renewable energy sources (RES.) This paper found that the robust formulation was more flexible for the system, as the robustness parameter could be modified.

***Appendix 2b – Concept Analysis***

To analyze any solution paths, a detailed set of requirements needed to be developed and weighted for analysis. The weighted criteria, seen in Table 2 below, were selected to try and weigh design options by the customer needs list.

**Table 2: Weighted Evaluation Criteria**

| Criteria | Weight | Reason |
|---|---|---|
| Optimize/Forecast Daily Prices | 25% | Charge and discharge times must be accurate to maximize profit. |
| Predict Unavailability | 20% | Charging is not always possible. These times must be identified. |
| Adapt to Dynamic Changes | 20% | The system could erroneously purchase at peak times. |
| Feasibility | 15% | The team must be able to develop a working product on time. |
| Ease of Modification | 10% | Existing libraries might need modification to meet requirements. |
| Speed | 5% | System must operate quickly and under the sample rate. |
| Readability | 5% | The user must be able to read and understand the code. |

Short Criteria Descriptions:

As seen in the table, the first and most heavily weighted criteria is the ability to either forecast or optimize off daily electrical prices. This is the most integral component of the project since programs that are unable to complete this step, are unable to function at all. Next, the system should be able to adapt to times where charging is impossible or where there are large spikes or dips in price. This will generally allow the battery to capitalize on jumps in price for sale and maintain a minimum energy value. Feasibility and ease of modification weigh the ability of the team to either modify existing programs, libraries, and plugins to meet design requirements and the ability to finish the design on schedule. Speed and readability or secondary weights to system performance. Currently, speed is only a slight issue due to the combining of the subsystems, but still performs well within time tolerances.

Environment Evaluation:
Before analysis could be done on forecasting methodology and algorithm, it was necessary to determine which environments would be able to handle both optimization and forecasting. To accomplish this, a decision matrix was created (Figure 6) to measure three top choices with weighted design criteria. Here, MATLAB/Simulink performed noticeably better than the two other alternatives: having a built-in optimization program and multiple forecasting add-ins available. It also had the benefit of being easier to read. Since MATLAB can run external Python scripts, any functionality available in Python which is not in MATLAB already can be run at a slight cost in speed.

| Scale | Evaluation Criteria | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 - 7 | Predict and Optimize Daily Prices | Forecast Unavailable Times | Forecast Dynamic Prices | Feasibility | Ease to Modify | Speed | Readability | |
| Solutions | 25% | 20% | 20% | 15% | 10% | 5% | 5% | Score |
| MATLAB/ SIMULINK | 6 | 5 | 5 | 7 | 5 | 6 | 6 | [5.65] |
| Python | 5 | 5 | 4 | 3 | 7 | 3 | 5 | 4.60 |
| GAM | 4 | 2 | 2 | 1 | 4 | 2 | 5 | 2.70 |

**Figure 6: Program Environment Evaluation**

Forecast Methodology Evaluation:
With a programming environment chosen, the team wanted to evaluate forecasting methodologies for forecasting the price data. Commonly, processing styles are separated into three categories: statistical, computation, and hybrid algorithms. As seen in Figure 7 below, statistical methods can handle consistent-seasonal data, but struggle with dynamic changes and Boolean measures (such as unavailable times). However, statistical scripts are much faster than computational methods. The inverse can be seen for computational methods where speed and ease of use are traded for more accurate forecasts. Hybrid, the mixture of the two, generally can forecast daily prices with more accuracy, but is harder to develop.

| Scale | Evaluation Criteria | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 - 7 | Forecast Daily Prices | Forecast Unavailable Times | Forecast Dynamic Prices | Feasibility | Ease to Modify | Speed | Readability | |
| Solutions | 25% | 20% | 20% | 15% | 10% | 5% | 5% | Score |
| Statistical | 5 | 3 | 4 | 6 | 5 | 7 | 6 | 4.70 |
| Computational | 5 | 6 | 6 | 4 | 3 | 4 | 4 | 4.95 |
| Hybrid | 6 | 6 | 6 | 3 | 3 | 3 | 4 | 4.95 |

**Figure 7: Forecasting Method Evaluation**

From the decision matrices, the team decided to pursue MATLAB/Simulink as the primary programming environment.

***Appendix 2c – Project Flow***

The project flow was a standard iterative design process where feedback was used to evaluate designs at multiple stages of the project (Figure 1). Starting with the initial problem statement and scope research, the team used this period to develop customer needs and design objectives for the project.

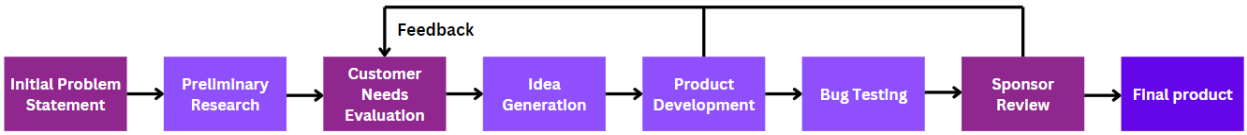


**Figure 1: Project Design Procedure**

After customer needs are identified, the team proceeds to idea generation and product development. Results of product development are compared to the customer needs to determine if they meet, or have the potential to meet, the requirements of the project. If changes are required, the process in question returns to the idea generation phase, elsewise it continues to bug testing and sponsor review. Sponsor review is used to determine the quality of the product, what additional features are desired, and have additional questions answered. Once the sponsor is satisfied with the project, it will be refined and prepared for presentation.

## Appendix 3 - Prototyping Process

### *Appendix 3a – Alpha Prototype Optimization Methodology*

Once an environment was decided upon, different optimization techniques native to the MATLAB/Simulink environment were identified. It was discovered that no native optimization functions existed in Simulink for optimization, but Simulink could run external MATLAB functions with optimization script included, which was the solution route chosen. To optimize price data, the group used the "optimproblem" function in the optimization toolbox. This toolbox can manually or automatically choose a processing methodology based on the input data type and set of variables and constraints. This function attempts to either minimize or maximize an objective function, or mathematical objective, by varying the design variables. In this case, the variables were state of charge and charge rate, with constraints involving the physical limitations of the battery system. Once the constraints and objective are defined the script runs and attempted to find peaks in the objective function and find the highest point.

Since the goal of the project is to maximize profit for the consumer, the objective function needs to relate the design variables (charge rate and SOC) to the maximization goal (Profit). To accomplish this, the following function is used where the following definitions are true:

| | |
|---|---|
| $G$ | Profit [USD] |
| $G_D$ | Discharge Cost [USD] |
| $G_C$ | Charge Cost [USD] |
| $q_D$ | Discharge Rate [MW] |
| $q_C$ | Charge Rate [MW] |
| $P_{Grid}$ | Grid Price in [USD/MWh] |
| $P_{Reg}$ | Regulating Reserve Price [USD/MWh] |
| $P_{Sp}$ | Spinning Reserve Price [USD/MWh] |
| $P_{Su}$ | Supplemental Reserve Price [USD/MWh] |
| $P_{Min}$ | Minimum Price for Purchasing [USD/MWh] |
| SOC | Battery State of Charge [MWh] |
| $\eta$ | Battery Efficiency [-] |

Objective Function:

Profit is a function of money made minus money spent on energy.

$$G = G_D - G_C \tag{2}$$

The amount of money made from discharge is the sale cost times to charge rate over a time interval $\Delta t$. Only a portion of the energy makes it out of the battery denoted by $\eta$.

$$G_D = \eta P_{Grid} q_D \Delta t \tag{3}$$

Similarly discharge profit, the charge cost uses the same method, but the consumer pays for the energy pulled, whether it is received or not so efficiency is not a factor.

$$G_C = P_{\mathrm{Min}} q_C \Delta \mathrm{t} \tag{4}$$

Combining equations (1-3)

$$G = \eta P_{Grid} q_d \Delta t - P_{Min} q_C \Delta t \tag{5}$$

Variable Ranges:

The battery has a maximum storage capacity of 10 MWh and cannot have negative energy.

$$0 \leq SOC \leq 10\ [MWh]$$

The battery can pull or push a maximum of 1 MW.

$$0 \leq q_{C,D} \leq 1[MW]$$

The efficiency of the battery's charge and discharge is dependent on the amount of energy current stored in the system.

$$\eta = \begin{cases} 0.6 & 10 \geq SOC \geq 8 \\ 0.8 & 8 > SOC \geq 2 \\ 0.7 & 2 > SOC \geq 0 \end{cases}$$

Constraints:

The state of charge of the battery at any given point is equal to the last state of charge plus the net energy entering the system. Note efficiency is only added to the charge and not discharge.

$$SOC(t) = SOC(t-1) + \eta q_C \Delta t - q_D \Delta t \qquad (6)$$

The battery charge and discharge at the same time.

$$0 = q_C q_D \qquad (7)$$

Other Functions:

Since it is necessary to determine what the battery is pulling from at any given point, the translation matrix T – consisting of a ones and zeros- can be multiplied by the minimum charge rate to get the energy pulled from each source.

$$[P_{Grid} \quad P_{Reg} \quad P_{Sp} \quad P_{Su}] = [T].* P_{Min} \qquad (8)$$

The "optimbroblem" function struggles with variables depending upon what they are affecting and would often result in a solution which broke the constraints. As such, the efficiency is assumed at the start of the simulation to be 80% which coincides with the largest operating range. After a simulation is done with this result, the State of Charge at every point is used to determine what the efficiency should have been. This process of simulation is iterated for a set number of times or until the efficiencies converges.

Additionally, to tell the program that prices at zero are unavailable, a matrix is made with zeros and one to denote whether it can charge or discharge. This can be modified in the future to incorporate restricted charge times where the battery itself is unavailable instead of the grid or reserves.

Simulink Data Processing Methodology:
Outside of the optimization script, the Simulink environment had additional components to maximize optimization speed and accuracy. Three processing steps were added. First, the environment needed to set an input interval to the optimization function. Without this step the optimizer would attempt to maximize profit from a single datapoint. Second, this interval was given an additional "context" for every optimization. Without the context, the optimizer will try to empty the battery at the end of the simulation, no matter the price. Finally, a post processer is added to determine the exact efficiencies

and state of charge values based on the charge rates. This is to avoid the battery slowly drifting from the expected values.

Iteration Interval and Context:
To best input data into the simulation, the dataset will have to be grouped into sections from the forecast information. The data is first grouped and sent as a batch to the iteration size desired using a "buffer" block in Simulink. This batched is then sent to another set of functions which replaces a portion of the last simulation with the new data. Every optimization, the newer data gets moved forward until its output is used in the scheduler then discarded. The exact data size flow can be seen in Figure 8 below:

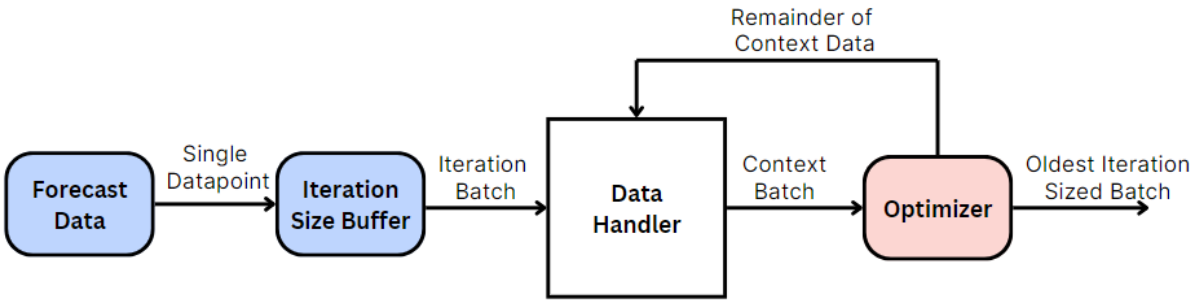


**Figure 8: Batch Processing Methodology**

For instance, if a 100 datapoint dataset is processed with 5-point iterations and a context of 20, the data would be handled as seen in Figure 9. Here, the first 4 There are two side effects to this methodology. First, there is a ramp up time in the script where there are still zero-value datapoints from before the context buffer fills. This is eliminated by switching off the data input till after the data is cycled through the initialization values. Next, there is an additional bias at the final datapoint since there will be progressively less valuable context as the dataset runs out. The hope is that with constant data input the latter issue will not become a problem, but in case this is not viable an assumed "trend" will need to be pre-generated to eliminate the programs desire to empty the battery system.

The current model does not loop the data back from the optimizer, but it operates similarly. Future renditions of the optimizer should use this method as it should operate more consistently and allows for more flexibility with the data handling.

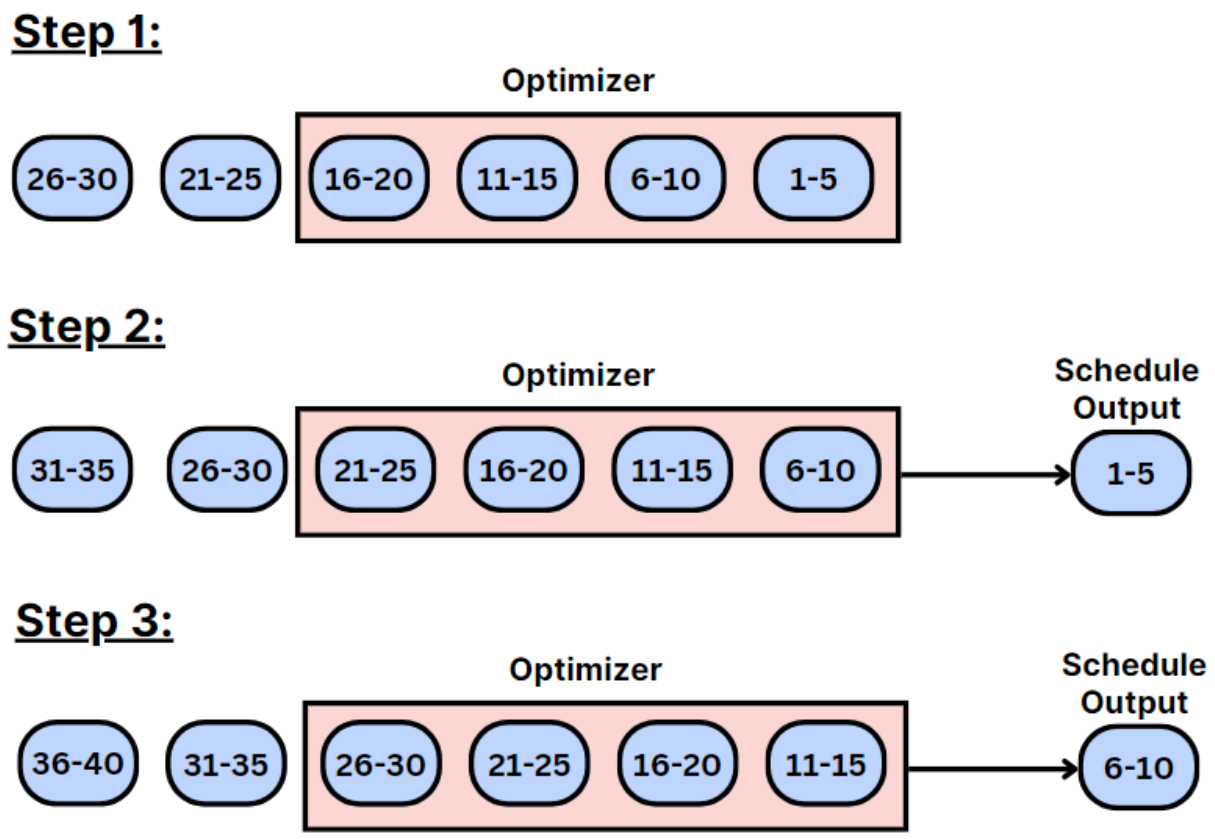


**Figure 9: Batching Methodology Example**

Post Processor:
When the optimizer finds an optimal schedule to make profit, it must manage the battery and determine what percentage of the storage is used at any given time. This is to prevent the battery from attempting to overcharge or attempting discharge when empty. However, in cases where the state of charge jumps over an efficiency point between time intervals, inaccuracy is added. Over time this can completely skew the optimization results. Normally, the real state of charge can be used to correct the values in real time, but since that information is not available, a placeholder option will be used.

The post processor inputs the initial battery charge from the previous optimization iteration and use the output charge and discharge rates to determine the actual final charge and profit of the schedule. The final charge output from this system will then by used as the initial charge for the next optimization iteration. Currently, the correction works by identifying locations where the efficiency changes across a datapoint and subtracting the extra output from the final point.

First, the system determines the percentage of the new data over (or under) the point of efficiency change.

$$SOC_p = \frac{SOC(t) - (8 \; or \; 2)}{SOC(t) - SOC(t-1)} \qquad (9)$$

Next, the updated final energy can be found by subtracting the estimated final by the percentage of the datapoint over the data times the efficiency difference of the charge rate.

$$SOC(t)_{new} = SOC(t)_{old} - SOC_p[eta(t-1) - eta(t)]q_C(t-1)\Delta t \quad (10)$$

The difference in state of charge between simulations can be seen in Figure 10 where an obvious charge difference becomes apparent as soon as 70 hours into the simulation. This different tends to expand over time and might cause significant error if not corrected early.

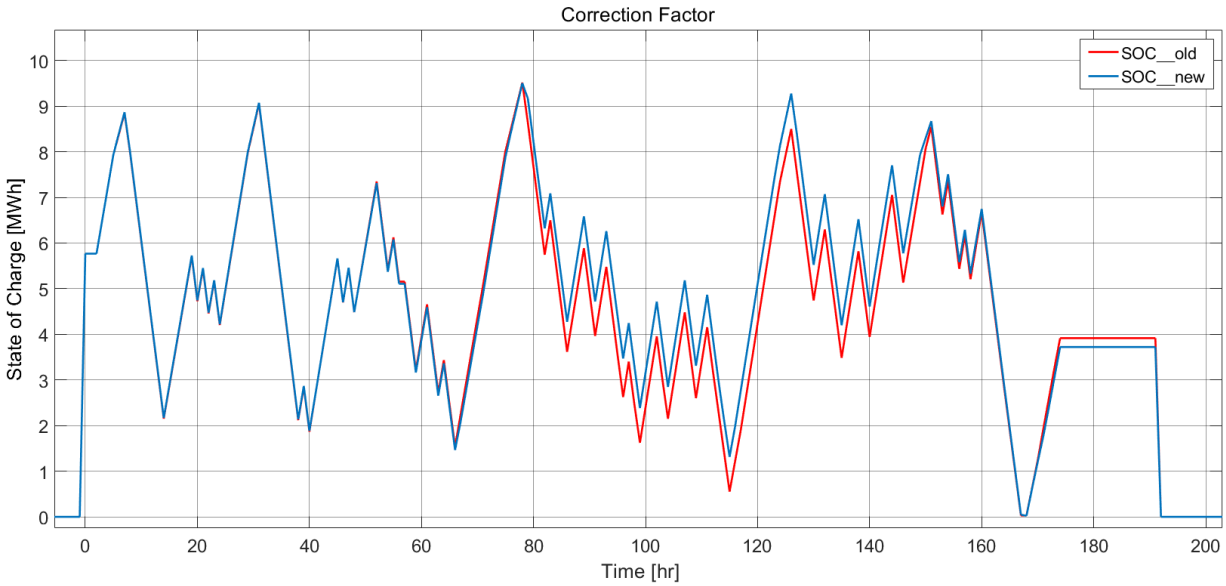


**Figure 10: Energy Drift from Efficiency Errors**

Alpha Experimental Procedure and Results:
With a functioning optimization script, the team wanted to test the processing speed and efficacy of different iteration and context sizes. The goal of this is to determine what type of processing duration is necessary for high accuracy while minimizing the time taken. The lower the processing time, the more locations where optimization can be implemented between sample points. The processing speed and profit was compared to the MATLAB simulation for reference (186 batch with 186 context). The time was taken using the default MATLAB functions "tic" and "toc" which display the real time between each run. For Simulink "tic; sim('AlphaName'); toc" was used to run and display operation time. The results are summarized in **Error! Reference source not found.** below.

**Table 1: Simulink Optimization Results**

| # | Iteration Size | Context Size | Profit | Total Time | Time/Iteration | Time/Batch |
|---|---|---|---|---|---|---|
| Mat | 168 | 168 | $405.80 | 13.82 seconds | 13.82 seconds | 13.82 seconds |
| Sim | 168 | 168 | $408.30 | 13.20 seconds | 13.20 seconds | 13.20 seconds |
| 1 | 24 | 76 | $ 419.90 | 26.37 seconds | 3.77 seconds | 11.30 seconds |
| 2 | 24 | 48 | $401.40 | 20.63 seconds | 2.95 seconds | 5.89 seconds |
| 3 | 24 | 24 | $363.10 | 12.42 seconds | 1.77 seconds | 1.77 seconds |
| 4 | 8 | 48 | $419.90 | 68.04 seconds | 3.24 seconds | 19.44 seconds |
| 5 | 8 | 24 | $370.00 | 37.44 seconds | 1.78 seconds | 5.35 seconds |

Here, it is noted that the two highest profit options (simulation #1 and #4) both had higher processing times than other options. Additionally, results with less context (Simulation #3 and #5) had unacceptable results compared to other options. For future iterations, the minimum context size will be 48 hours.

The simulation was done on a 186 datapoint set from Lockheed's dataset which would simulate a full week's data. The dataset, seen in Figure 11 below, attempted to include as many aspects of optimization as possible: including a large spike, zero values, and varying reserve prices.

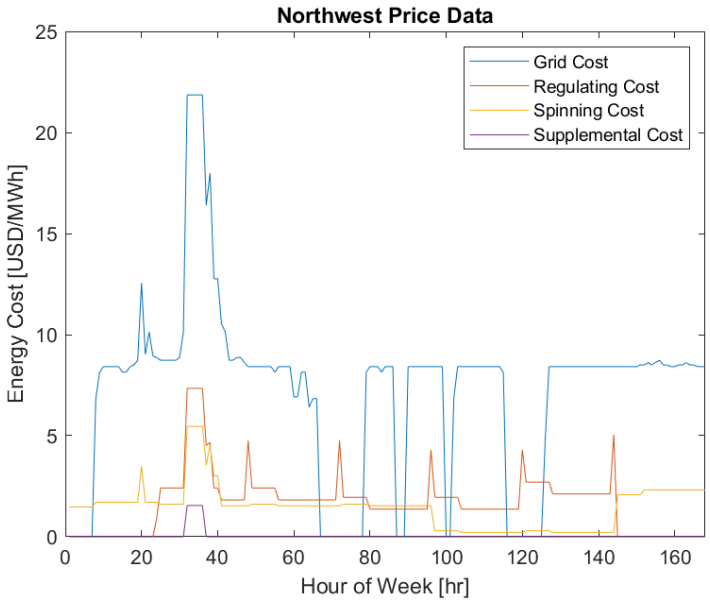


**Figure 11: Week of Price Data for Analysis**

The alpha proved relatively successful across all iteration and context sizes. In Figure 12 below, the state of charge of the battery is compared to the prices of reserve and the grid.

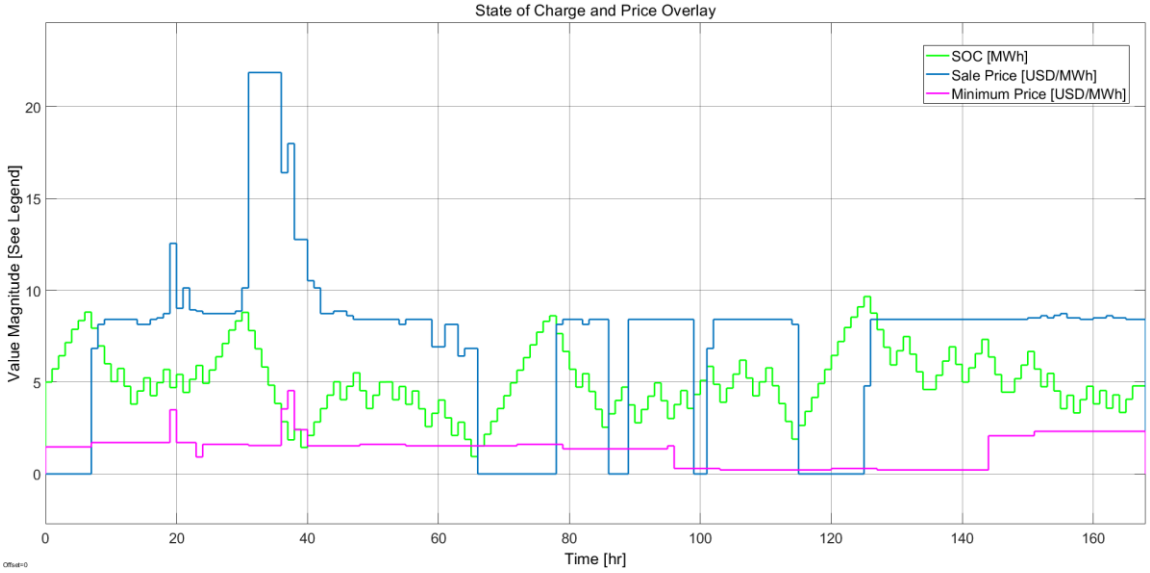


**Figure 12: Optimized SOC VS Electrical Prices**

As seen in the figure, the SOC of the system rises when the minimum price is low: especially at ideal times such as hour 0, 65, and 115. Additionally, at peak prices such as hours 30-40 there was a huge amount of discharge which is expected. However, there is room for improvement. The battery seems to try to both charge and discharge at times where there is beneficial, but not consistently. It instead jumps up and down in charge, not allowing for the battery to operate at a constant charge rate for any reason during this time. This might either prove useful or harmful depending on the forecast's accuracy but is currently viewed as a bug and requires fixing.

***Appendix 3b – Forecasting Methodology***

To generate the scheduling data required for the optimizer, a price and availability forecast was necessary to predict prices for the next 24 hours. The solution first split the process into two distinct sections: price forecasting and unavailability forecasting – both using different methodologies and algorithms. However, before either of these models were developed, initial data analysis was performed.

Data Analysis:

To begin the analysis, the team decided to look at broad statistics first to understand the problem and look for expected trends or result forms. First trends were observed for average price per hour, average price per day of week, and average price per month, seen below. It is worth noting that "processed" data was the filtering of data spikes to see if there was a significant contribution to the data form.

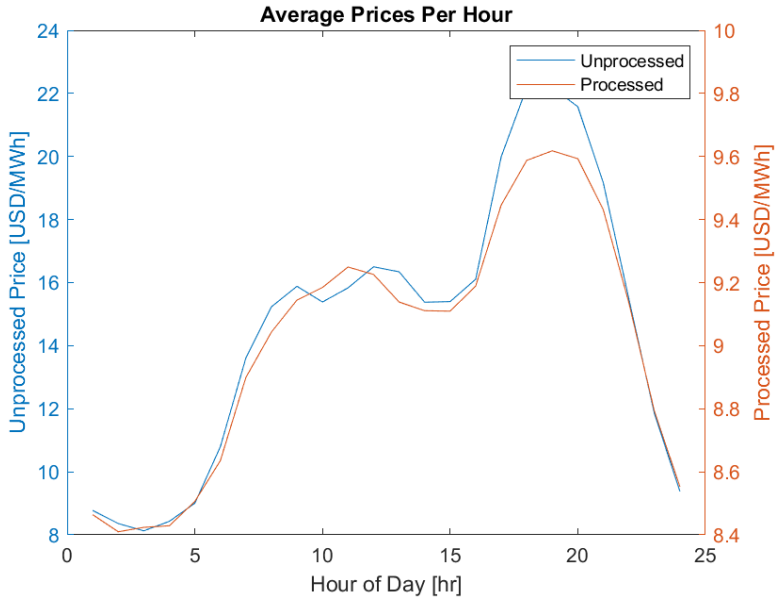


**Figure 13: Average Price Per Hour of Day**

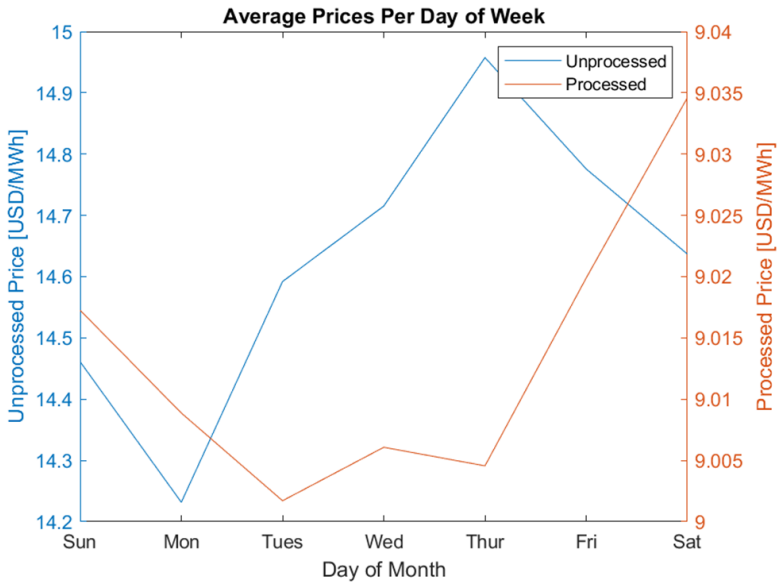


**Figure 14: Average Price Per Day of Week**

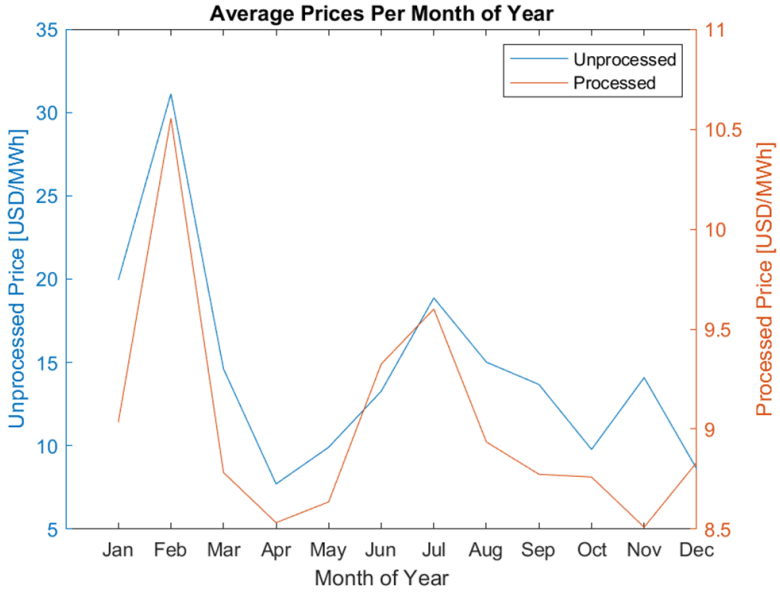


**Figure 15: Average Price Per Month of Year**

From the results, it was clear that price spikes seemed to occur in the on-peak hours of the day, as expected, as well as during Winter and Summer months during the year. These trends did match expected prices due to AC usage and other time and weather specific factors to the price. However, though the processed price matches the trend – for the most part – in the hourly and monthly datasets, the day of week spiked data deviated from the processed data. The team concluded from this that, though the average price of energy increased on the weekends, the number of spike prices (due to sudden unavailability and high demand) occur during the week more commonly. This would make sense based on physical factors and should explain the data.

Next, to measure the correlation to the price data and itself, the team measured the autocorrelation using the Sample Autocorrelation Function (ACF) and Sample Partial Autocorrelation Function (PACF). These processing techniques measure how much the lags cumulatively and individually

affect the current price respectively. The ACF function (Figure 16) demonstrates that there is a steep drop in contribution after the first couple lag prices, however, after the data levels out, additional correlation appears every 24-ish hours. This implies that the data has a 24-hour seasonality which would make sense based on the above statistics. On the other hand, the PACF (Figure 17) demonstrated that only the first lag, and possibly the second lag had any direct effect on the current price. Some additional input does seem to appear around lag 25, but there was unlikely to be much direct contribution due to this input.

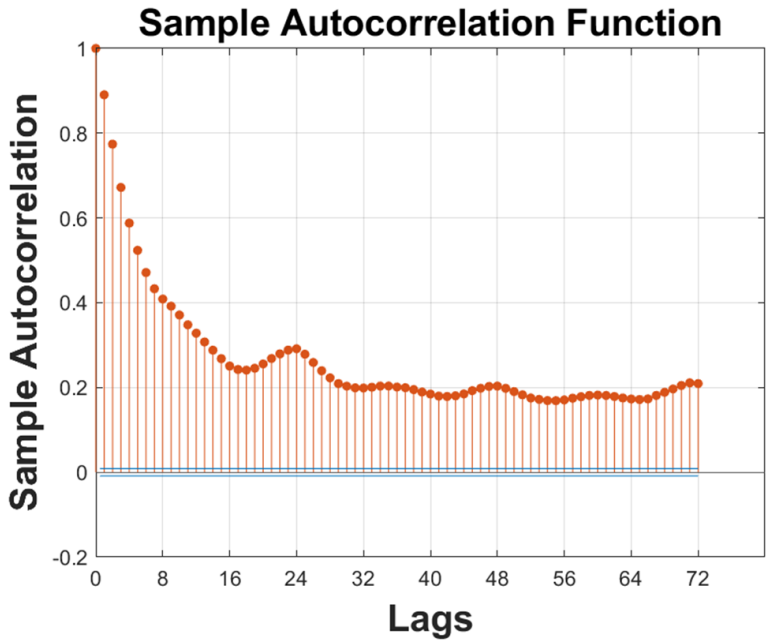


**Figure 16: Grid Price ACF Plot**

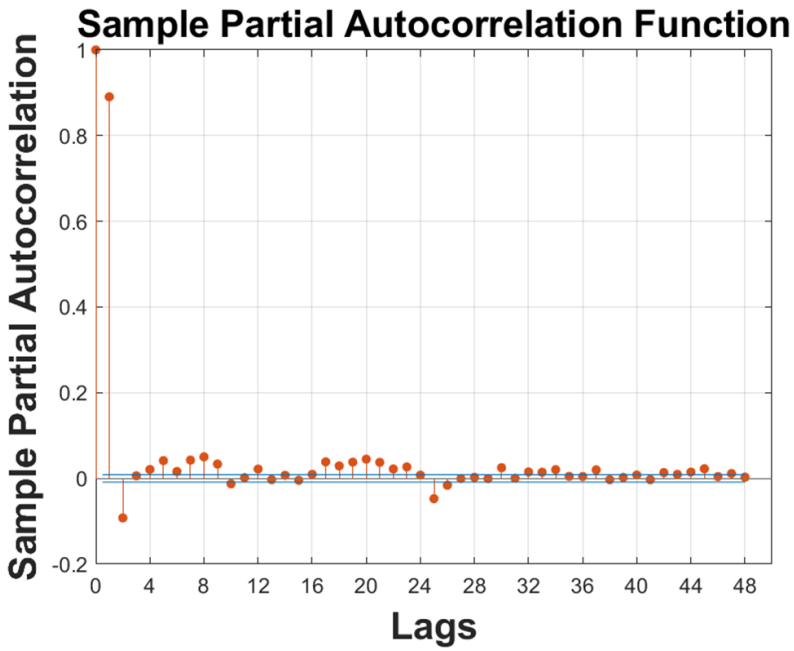


**Figure 17: Grid Price PACF Plot**

Linear Regression Forecasting:
With this analysis complete, a simple forecast was developed to attempt and find this trend from the data. Linear regression was chosen for this process since it allowed for the development of a model solely based contributions from each chosen factor. Namely, the contribution from the inputs could be easily measured for future development (where computational methods could achieve a greater fit).
The regression forecast was developed and included running average statistics, the first lag value from the PCF, and a 24-hour seasonal difference. The regression, done in log scale, can be mathematically represented below where Q is the log price.

$$\hat{Q}_{t+n} = Q_t - (Q_{t-n} - Q_{t-24}) + \bar{Q}_{DoW} + \bar{Q}_{MoY} \quad (1)$$

In the above formula, t is the current time, n is the number of horizons of the forecast, $\bar{Q}_{DoW}$ is the average hourly price of the weekday, and $\bar{Q}_{MoY}$ is the average hourly price of that month in the year. The goal is to keep the form of the recent statistics ($\bar{Q}_{DoW}$ and $\bar{Q}_{MoY}$) while reacting to the recent price jumps in relation to the previous recorded price. The statistics are evaluated on a moving context window of 1 year.

Linear Regression Program:
To generate an accurate forecast, an individual linear regression forecast "object" was generated for each hour ahead on the horizon. Since the system was developed to forecast 24 hours ahead, 24 linear regression models were created in the form of the above formula. These were individually trained on the same data in a moving window form of 365 days. The model objects were re-created every time the forecast was run, and average statistics were updated to input into the training data. The goal was to have the system be dynamic enough to react to rapid changes in trend, while also matching the cumulative trend of the data. Once trained, the last 25 known prices were used to predict the next 24 hours.

Once the forecast was generated in log scale, it was converted back to correctly scaled values and compared to actual prices after the schedule period ended. Cumulatively the model did not perform as expected: having high error (Figure 18) and reacting to price spikes late, if at all (Figure 19). Though this caused massive inaccuracies, the other portions of the data fit well, and the result was usable enough for the optimizer to run.

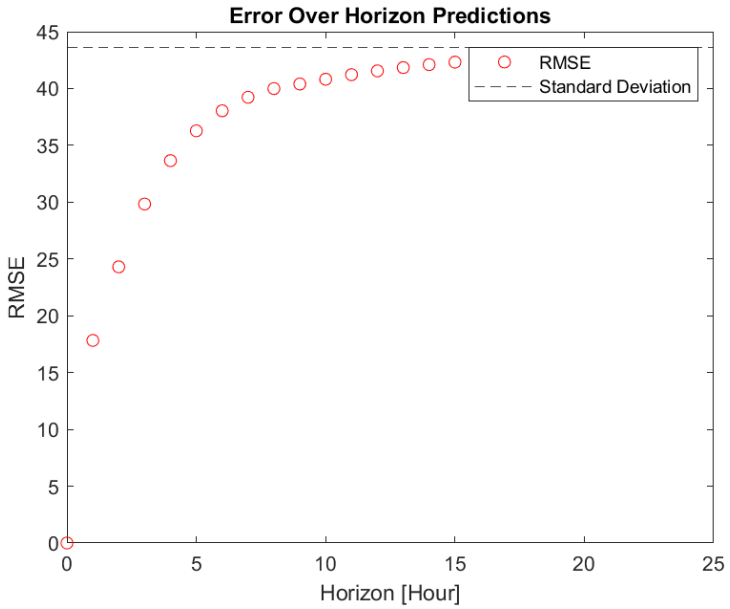


**Figure 18: Root Mean Square Error of Forecast Horizons**

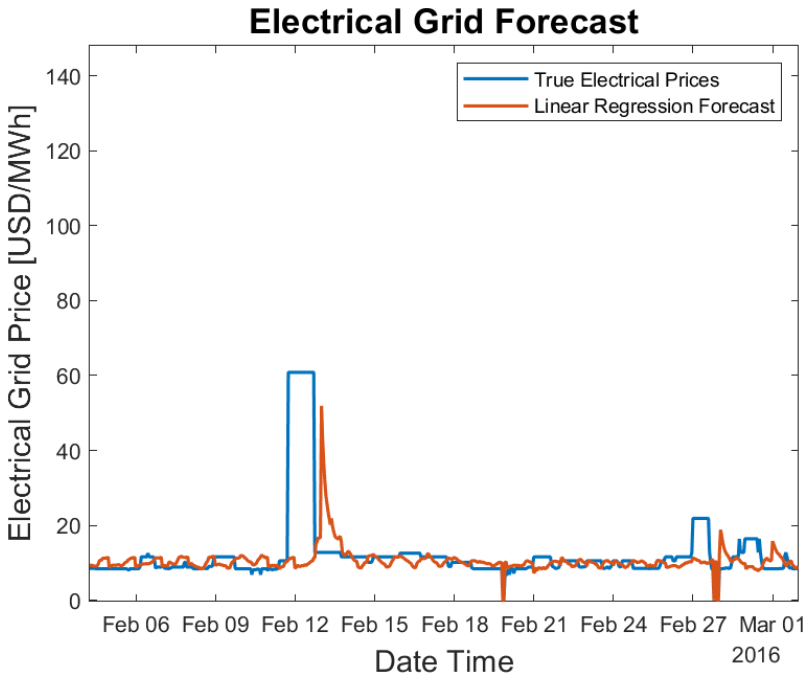


**Figure 19: Linear Regression Forecast Sample from the Grid**

Classification Methodology:
The classifier used a different approach to find the unavailability times. Since the unavailability for each source and the battery was predictable based on the time and date, the hour, day of week, and day of month statistics were input into a random forest classifier to find the pattern of availability. The built-in MATLAB random forest classifier was used to generate the prediction and the same training period was used as the linear regression model.

The results from this method had high accuracy for all sources, seen in (Table 3) below:

**Table 3: Classifier Accuracy Results [Sample Region]**

| # | Battery | Grid | Regulating | Supplemental | Spinning |
|---|---|---|---|---|---|
| **Accuracy** | 98.69% | 94.85% | 98.26% | 98.09% | 95.51% |
| **True Guesses:** | 15, 575 | 14,969 | 15,507 | 15,479 | 15,073 |
| **False Positives:** | 125 | 660 | 83 | 26 | 473 |
| **False Negatives:** | 82 | 153 | 192 | 227 | 236 |

The results suggest a strong and consistent relationship between time of day and unavailability times. In future work, it might be useful to use data from this analysis in the prediction of electricity prices as it might contribute to a stronger fit without much risk of overfitting.

***Appendix 3c – Beta Prototype Optimization Methodology***

During the alpha prototype, sale to reserves functioned just like sale to grid. However, in reality, when selling to a reserve you "ensure" an hour of energy for the on- or off-peak period (hours 8-23) and (hours 1-7,24) respectively: energy that may or may not be purchased. Additionally, the optimizer input was assumed to have no error and it was not robust or have stochastic capabilities. As such, it needed to be completely re-developed to handle these additions.

Assumptions:

Before generating the optimization script, the team had to determine the assumptions that were used in the project. These assumptions determined the difficulty, accuracy, and features of the system and allowed for the beta prototype to finish on schedule. The cumulative list of assumptions can be found in (Table 4).

First, regarding the entire system, we assume the battery specifications to operate within a 0 – 10 MWh SOC and have a maximum energy output of 1 MWh. The battery should have an expected depth of discharge (DoD) of 100% and discharging the entire battery should have no effect on the lifetime of the system. Additionally, the battery has varying charge/discharge efficiencies depending on the state of charge. Between 0 and 2 MWh, the efficiency is 70%; between 2 and 8 MWh, the efficiency is 80%; and between 8 and 10 MWh, the efficiency is 60%. The charge and discharge efficiencies match these regions: influencing profit and charging costs.

The program itself was assumed to operate on one battery which could only perform one action at a time (buy, sell, sell to reserve, or do nothing). The output of the battery was assumed to be properly moderated to the voltage and frequency required by grid regulations to simplify the optimization. Sales to reserves were assumed to have a certain chance of discharging energy during the guarantee period: which would increase the profit for that specific day. This discharge position was assumed to occur at the most expensive available grid price during the insurance region.

**Table 4: List of Assumptions and Justifications**

| Assumption | Justification |
|---|---|
| Battery operates between 0 and 10 MWh | Arbitrarily chosen based on specs of Colorado base. |
| Battery can charge or discharge at 1 MW | |
| Battery has a depth of discharge of 100% | Validated in Specs |
| Efficiency Values<br>0 – 2 MWh: 70%<br>2 – 8 MWh: 80%<br>8 – 10 MWh: 60% | Given by Sponsor |

| | |
|---|---|
| Reserve Chance of Discharge:<br>Regulating: 30%<br>Supplemental: 10%<br>Spinning 10% | |
| Optimization Occurs on One ESS | Given by Sponsor |
| Voltage and Frequency Regulation already in place. | Given by Sponsor<br>Highest price is associated with the highest demand point. |
| Grid and Reserve prices are only realized after the time has passed (every 24 hours) | |
| If discharge occurs during reserve sale, it occurs at the highest available grid price. | |
| The battery is continuously aware of its own state of charge. | The battery is likely to be aware of its own status including its SOC. |
| The battery will only charge and discharge at maximum speed. | Assumed to simplify optimization model and allow for better results validation. |
| Optimization assumes energy is always discharged during insurance regions at no additional profit. | Assumed to simplify optimization model and allow for better results validation. |

Robust Optimization:
Since the prices are only realized after the entire 24-hour period occurs, it does not make sense to use a stochastic model for this 24-hour region. However, an unmodified deterministic solution would miss the uncertainty native to the problem. As such, the team decided to use Robust Optimization (RO) to model the uncertainty from the forecast.

Robust optimization generates realistically pessimistic values for the purchase and sale prices of the system. These prices are found using the residual (error) distribution from the forecast. This residual was estimated to fit the Generalized Extreme Value distribution (Figure 20) and was generated for each horizon in the 24-hour forecast.

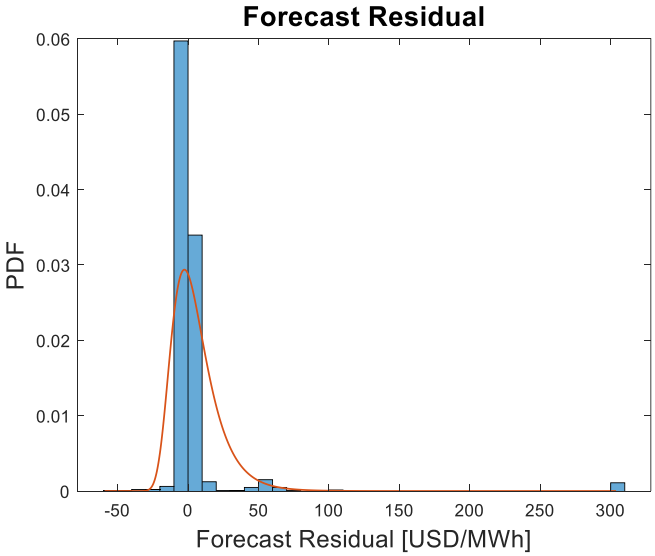


**Figure 20: Residual Price Distribution**

This change adds some complexity to the optimization problem. The entire process is described below.

Where $C_t^R$ is the robust cost at time t and source i, and $q_{t,i}$ is the charge or discharge rate at time t and from source i. $C_{RES}$ is the additional profit from ensuring to a reserve.

$$\min J = \sum_{t=1}^{24} C_{t,i}^R \times q_{t,i} - C_{RES} \quad (11)$$

The robust cost is generated from current probability value plus or minus the robust factor depending on what the output is (sale or purchase).

$$C_{t,i}^R = CDF\left(Prob\left(C_{i,t}\right) \pm \zeta_{t,i}\right) \quad (12)$$

The result must be feasible.

$$0 \leq Prob\left(C_{t,i}\right) \pm \zeta_{t,i} \leq 1 \quad (13)$$

The robust factor is based on an arbitrary robustness value (0 for optimistic and 1 for pessimistic ranges).

$$\zeta_{t,i} = A_{t,i}(2\Upsilon - 1) \quad 0 \leq \Upsilon \leq 1 \quad (14)$$

The scale for each cost horizon is based on its maximum and minimum ranges.

$$A_{t,i} = \min\left(\min\left(C_{t,i}\right), 1 - \max\left(C_{t,i}\right)\right) \quad (15)$$

The state of charge is equal to the last state of charge plus the efficiency times the attempted charge rate (1 hour at 1 MWh) minus the attempted discharge rate (1 hour at 1 MWh).

$$SOC_t = SOC_{t-1} + \eta \times qRC_{t,i} - qRD_t \quad (16)$$

Only one action can be taken at a time.

$$\sum_i qRC_{t,i} + qRD_t + HOLD_t = 0 \quad (17)$$

The SOC of the system must stay above the desired residual at every hour.

$$SOC_t \geq SOC_{\min} \quad (18)$$

The SOC of the system at the end of the optimization must be greater than the target charge (to allow for tomorrow).

$$SOC_{end} \geq SOC_{target} \quad (19)$$

The entire optimization process is split into three sections: Section 1 assumes that no energy is sold to the reserve, Section 2 assumes that energy is sold during the on-peak period, and Section 3 assumed that energy is sold during the off-peak period. At the end of the optimization, the minimum value between all three optimizations is chosen as the schedule. This choice was made to reduce the complexity of the system constraints.

Check Node:
Since the forecast will still have some inaccuracies even when considering the robust factor, the battery system might try to break its own constraints. For instance, if the battery fails to charge a few hours in a row but successfully discharges as expected, the battery SOC will be lower than predicted by the optimizer. These issues might cause the system to discharge all residual energy or not have enough energy to ensure to the grid.

The Check Node's job is to ensure this does not occur by setting minimum charge amounts and ensuring that the current true SOC stays above that region. Any deviations will be handled by the system, either by stopping the action or preventing reserve sale periods for the current or next day (Figure 21).

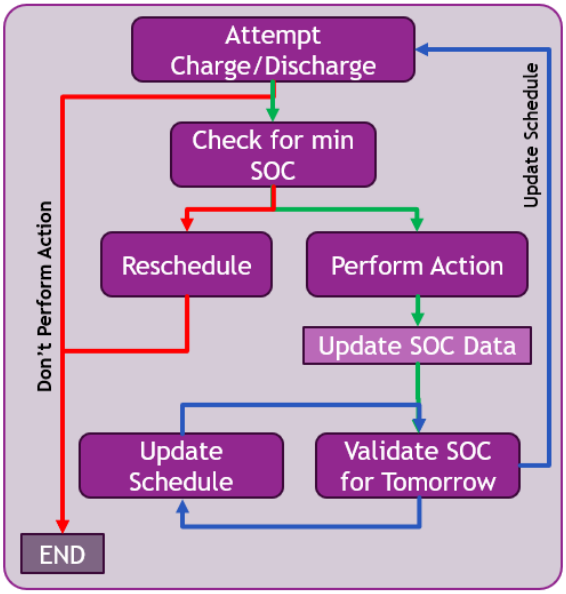


**Figure 21: Check Node Structure**

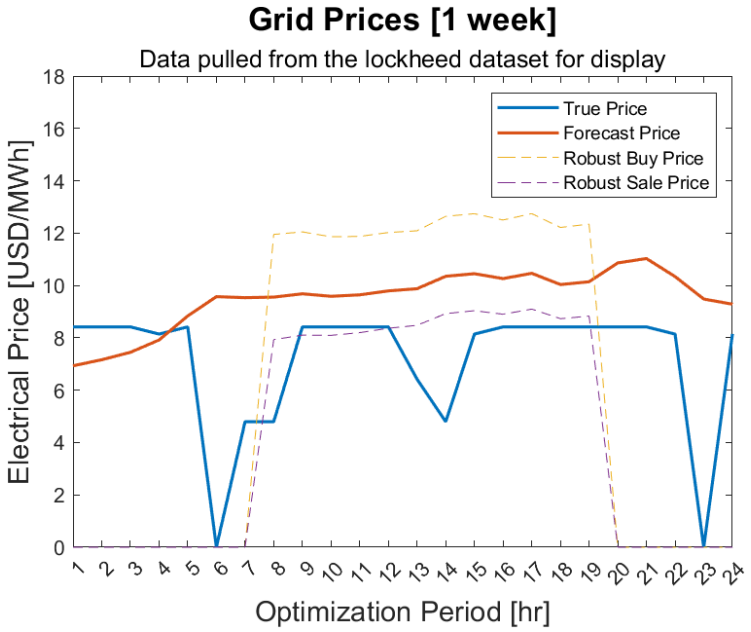


**Figure 22: Robust Forecast Grid Price for Risk Reduction**

***Appendix 3d – Final Results***

The final system simulated periods of the dataset and performed a 50-scenario Monte Carlo simulation on the results of the optimization. The system, once strung together, made a decent amount of profit over a month but inaccuracies in price forecasting could still cause major losses for the system. For instance, as seen in (Figure 23) below, the system made money over the month's

region but lost around $1000 in a short period of time. Even though the money was made back the next day, this issue could cause major short-term or long-term losses in the system.

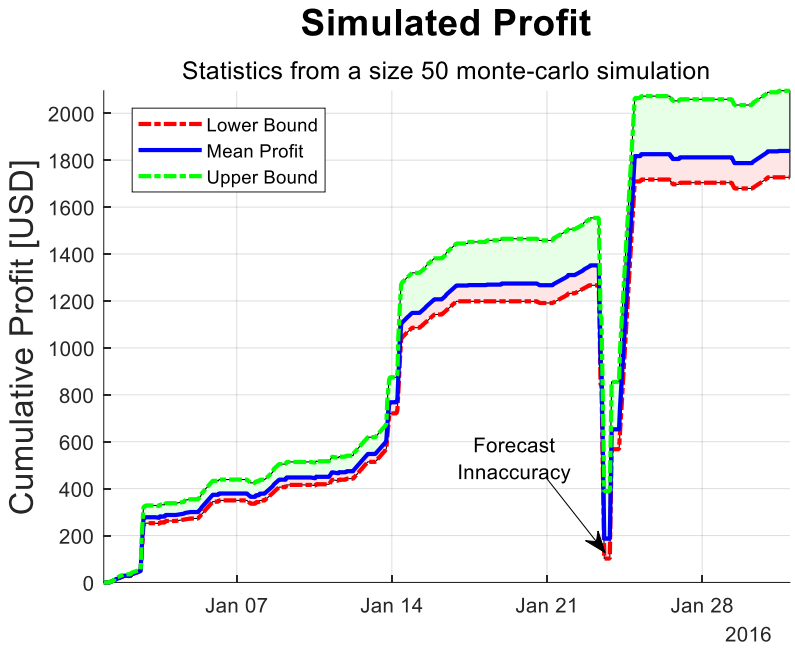


**Figure 23: Simulation Results**

However, despite this issue, the cumulative results did show promise. The Upper, Mean, and Lower profits were $2,096, $1,840, and $1,727 respectively. Additionally, most of the price drops were immediately re-made the next period- showing that the system was able to "invest" for the next day. Additionally, the system operated within parameters: managing its charge and discharge times to keep the SOC within the expected region (Figure 24).

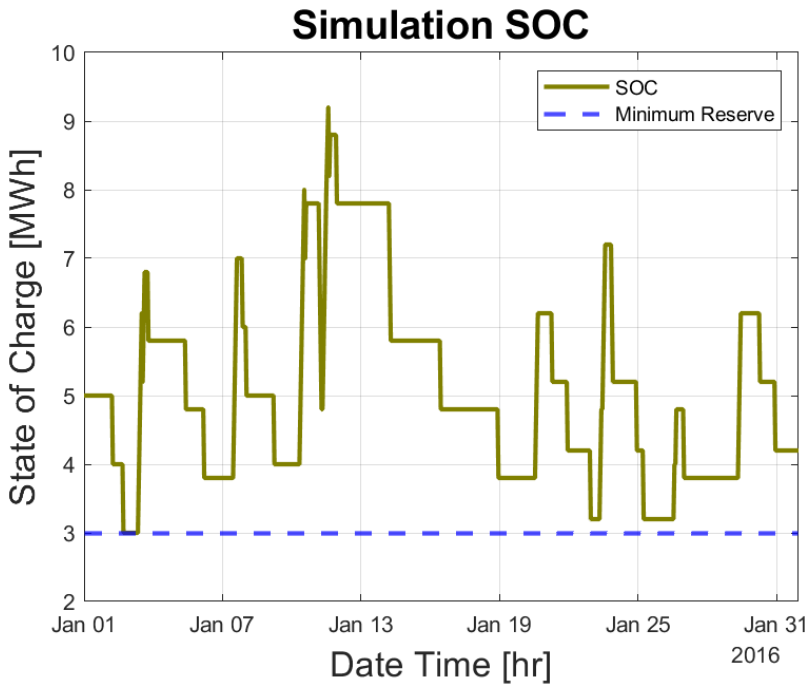


**Figure 24: Simulated SOC over a Month**

## Appendix 4 – Future Work

Though the team believes that the current solution does meet the customer needs, there is plenty of room for improvement. To help improve the product even further, and open a path for future development, suggestions and ideas are briefly discussed below.

### *Appendix 4a: Forecast Changes*

As stated, the forecast is imperfect which adds high error and variability into the system. The team thinks that upon further analysis, a more accurate forecast can be generated. To achieve this, the unavailability should be incorporated as training data into the price forecast. This should be done since long-consistent prices seem to be related to unavailability. The team suspects that this affected the quality of the linear regression model and skewed the data away from spike prediction. Additionally, the team recommends improved pattern recognition. Namely, more analysis into the contribution to the price from different lags, seasonal differences, or even seasonality in the differenced data.

Finally, the team recommends using a computational method or more robust linear program when predicting the data. Though a exploratory linear regression program is suggested for testing and evaluation, the final results would likely be improved with a computational approach do to the non-linearity in the data.

### *Appendix 4b: Optimization Changes*

Arguably the most important portion of the project, the optimizer has some limitations. It is difficult to directly assess the optimality of the current solution, but the team firmly believes that the system would benefit from the ability to look towards the future. This would take the form of either a two-stage stochastic process or a multi-stage stochastic process.

As of now, the robust optimization uses a deterministic solution. Namely, all inputs are known- or known within a range- for an optimization to occur. Only one iteration is necessary; there is only one scenario. This is unrealistic, but due to prices only being realized after 24 hours have occurred, a stochastic approach is not possible. However, the team would like to implement a stochastic "second step" towards the next 24-hour period. The goal is to optimize now with respect to tomorrow's problems and generate mitigation strategies for different scenarios the next day. Once the first 24-hour period is realized, the second period will become deterministic, and the cycle continues.

This would greatly increase the complexity of the system and the computation times, but the team believe that there is plenty of tolerance for accuracy improvements and, with an improved forecast, the stochastic solution would create a smarter, more informed, battery schedule to follow.