AMS-559 SMART ENERGY

VIBHOR SHUKLA (112671826)

Electricity Provisioning based on Demand Prediction

April 13, 2020

INTRODUCTION

We have to evaluate the performance of different electricity provisioning algorithms for both offline and online methods.

For online methods, namely Receding Horizon Control and Commitment Horizon Control, we have to use the predictions from at least 2 algorithms as implemented in Homework 1.

The objective function used in the assignment is as follows –

$$\sum_{t=1}^{T} p(t)x(t) + a * \max\{0, y(t) - x(t)\} + b|x(t) - x(t-1)|$$

The constraints are as follows:

a = b = \$4/kWh
p = \$0.40/kWh
T = 672 (2 week)
Time Period = Nov 1 to Nov 14

<u>Offline Optimization Problems</u>

The offline optimization problems are the ones which require the complete knowledge of the future. These are classical optimization problems which know beforehand which cases might fail.
To bridge the performance of the online problems with reference to the offline problems is the main objective at hand.

The types of offline problems to be used in this assignment are Offline Static Algorithm and Offline Dynamic Algorithm.

Online Optimization Problems

The online optimization problems are the ones which do not require the complete knowledge of the future. These are optimization problems which do not know beforehand which cases might fail.
They are helpful in situations where piece by piece information is available in an online manner

The types of online problems to be used in this assignment are Receding Horizon Control (RHC) and Commitment Horizon Control (CHC).

TASKS

1. Solve the offline optimization problem, e.g., using tools CVX in Matlab or Python.
2. Try online gradient descent (with different step size), receding horizon control (with different prediction window size), commitment horizon control (with different commitment levels). For the latter two algorithms, use predictions from at least two prediction algorithms for the default value of a and b.
3. Compare the costs of these algorithms to those of the offline static and dynamic solutions
4. For the best combination of control algorithm and prediction algorithm, vary a and b to see the impacts.
5. Try at least two algorithm selection (one deterministic, one randomized) to see if their performance.

TASK 1

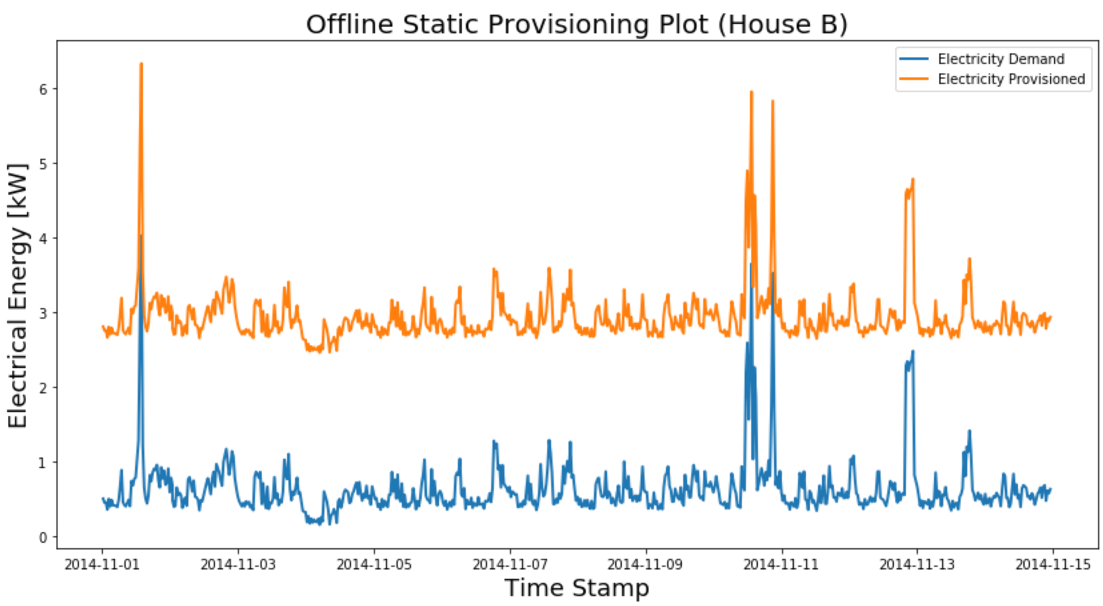
Solving the Offline Optimization Problem using Python.

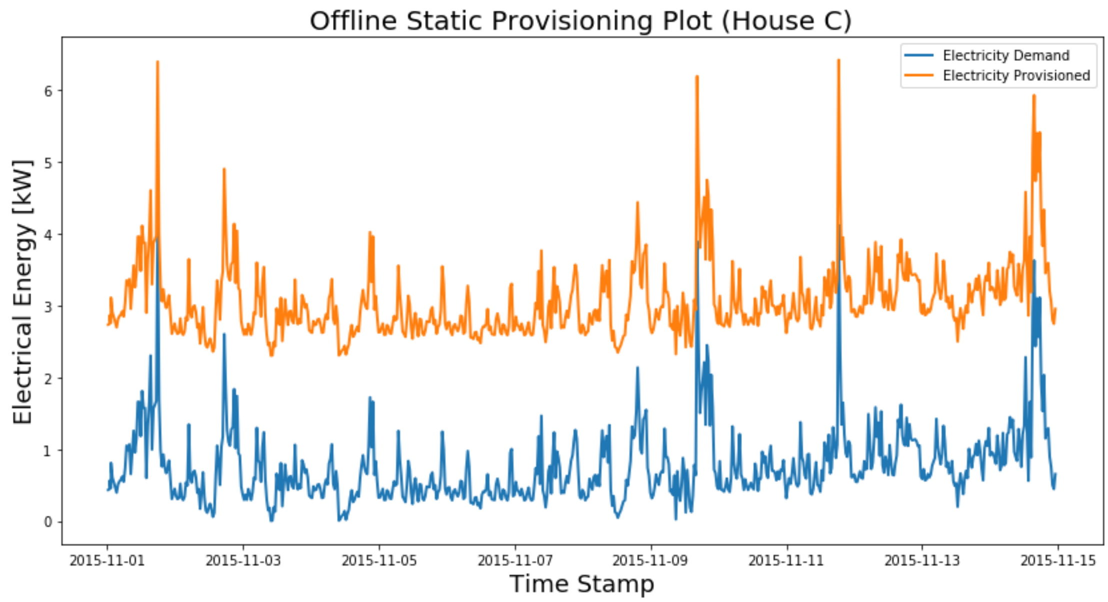
Offline Static Method

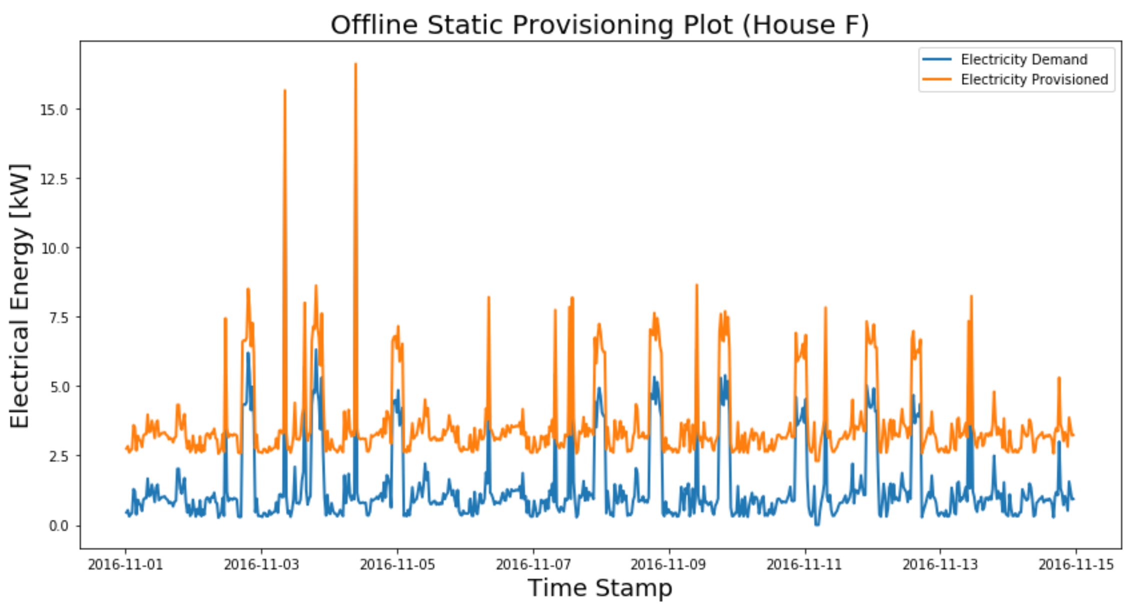
In the Offline Static Method, the switching cost is not used.
Values a = 4 and p =0.4 are used.
The objective function equation is as follows:

$$\sum_{t=1}^{T} p(t)x(t) + a * \max\{0, y(t) - x(t)\}$$

The following are the Offline Static Plots between Energy demand/provisioned and the time period of 14 days, T=672





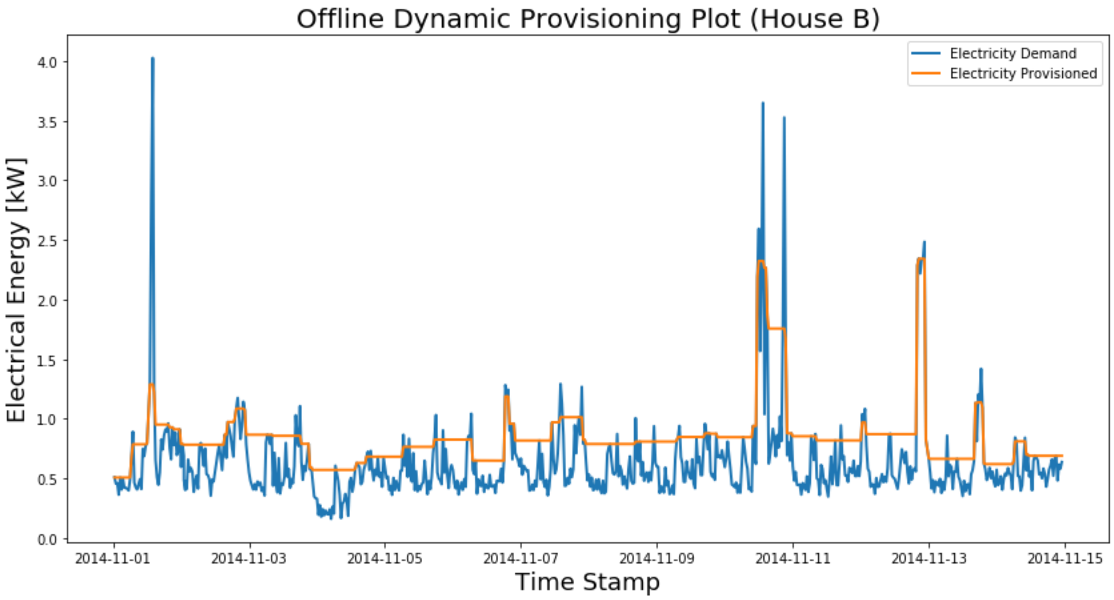


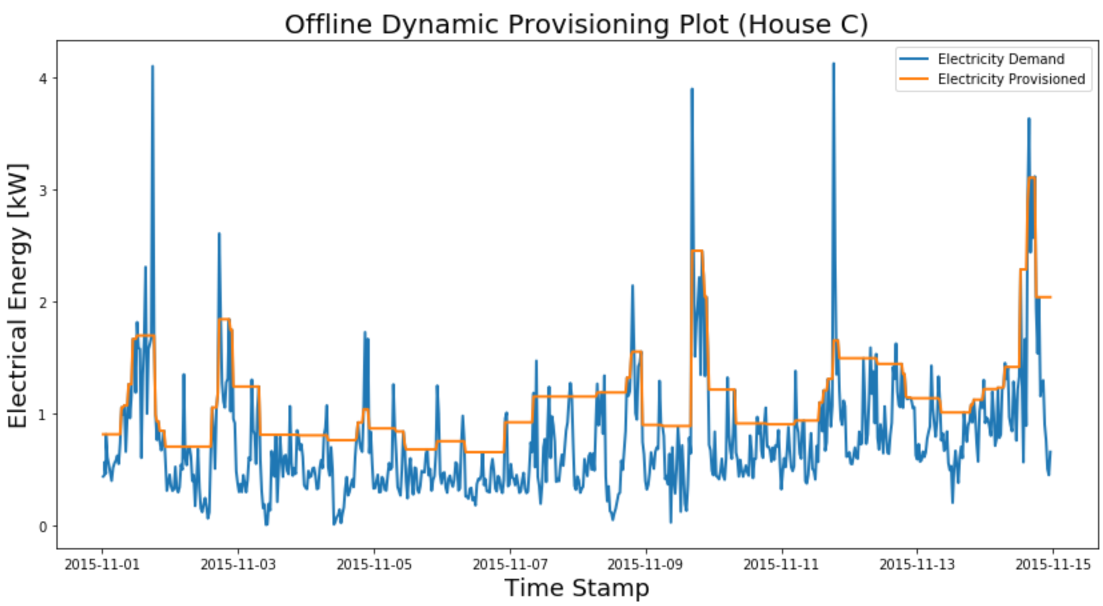
<u>Offline Dynamic Method</u>

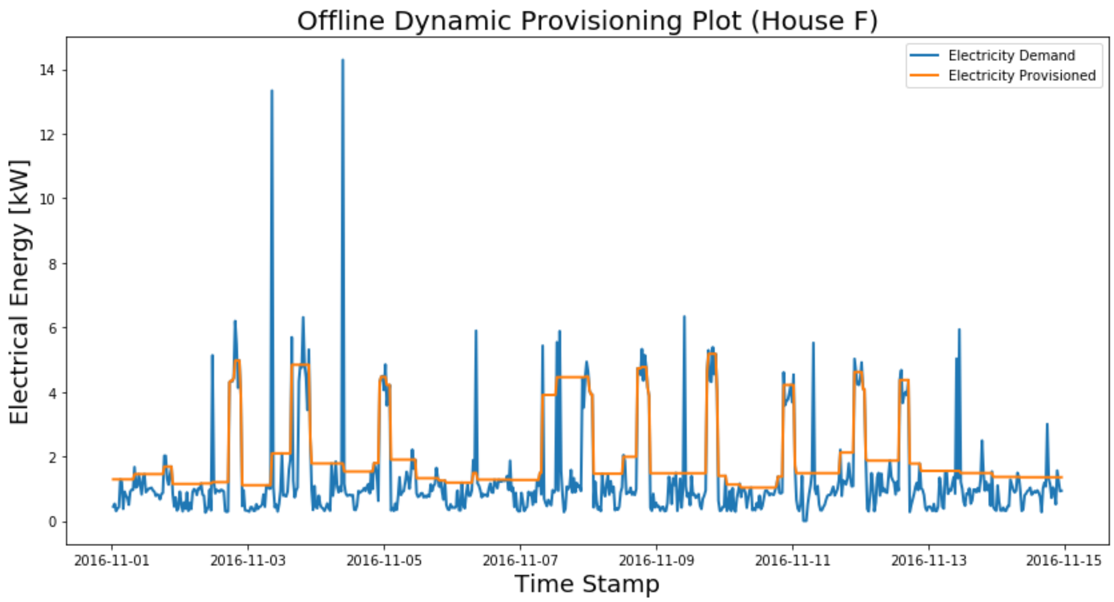
In the Offline Dynamic Method, the switching cost is used alongwith the penalty factor.
Values a = 4, b = 4 and p =0.4 are used.
The objective function equation is as follows:

$$\sum_{t=1}^{T} p(t)x(t) + a * \max\{0, y(t) - x(t)\} + b|x(t) - x(t-1)|$$

The following are the Offline Dynamic Plots between Energy demand/provisioned and the time period of 14 days, T=672


Offline Dynamic Provisioning Plot (House B)
Electricity Demand
Electricity Provisioned
Electrical Energy [kW]
Time Stamp
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
2014-11-01
2014-11-03
2014-11-05
2014-11-07
2014-11-09
2014-11-11
2014-11-13
2014-11-15


Offline Dynamic Provisioning Plot (House C)
Electricity Demand
Electricity Provisioned
Electrical Energy [kW]
Time Stamp
0
1
2
3
4
2015-11-01
2015-11-03
2015-11-05
2015-11-07
2015-11-09
2015-11-11
2015-11-13
2015-11-15



Comparing the Offline Static and Dynamic Method Costs

House	Offline Static Cost ($)	Offline Dynamic Cost ($)
B	1059.70	319.68
C	1091.84	417.62
F	1252.08	1029.77

TASK 2

Application of different provisioning algorithms

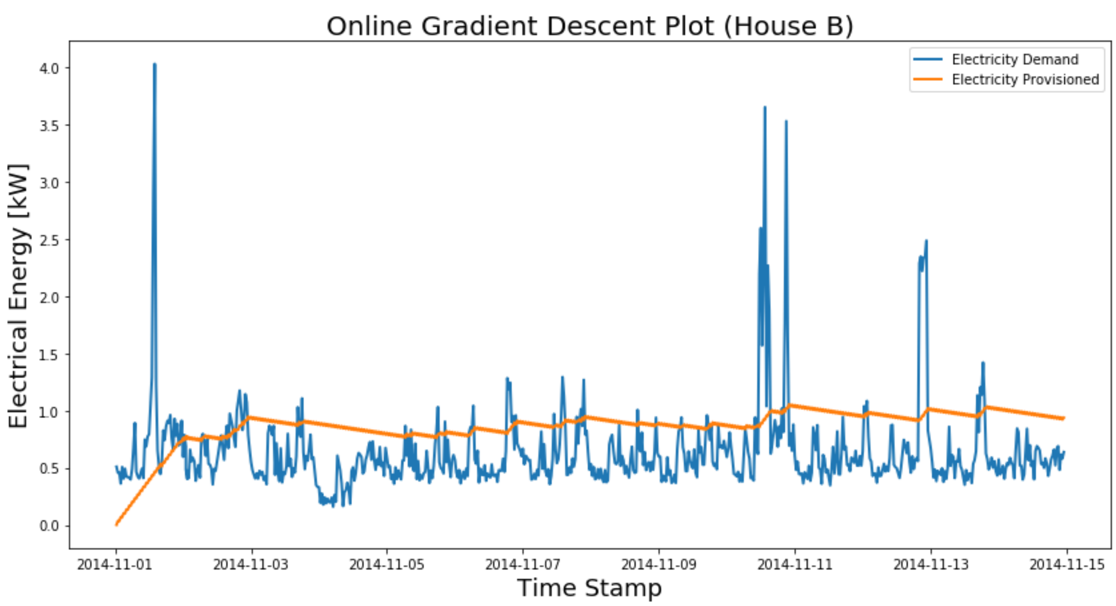
Online Gradient Descent

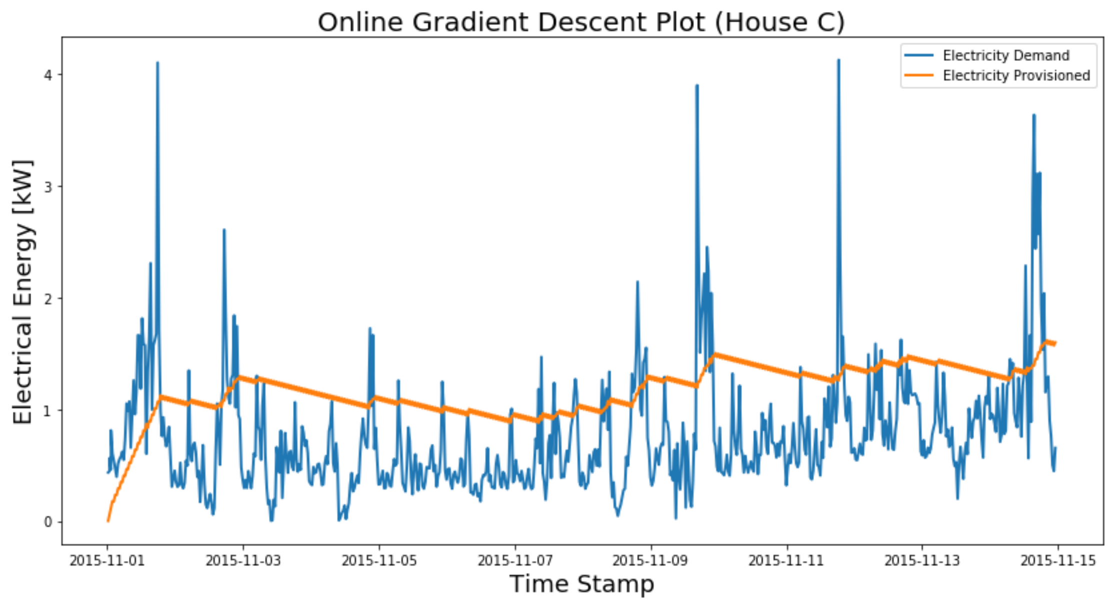
It is an online method, which means the new value is derived from the previous value.
So, x(t+1) is derived from x(t) using learning rate and a factor as represented below:

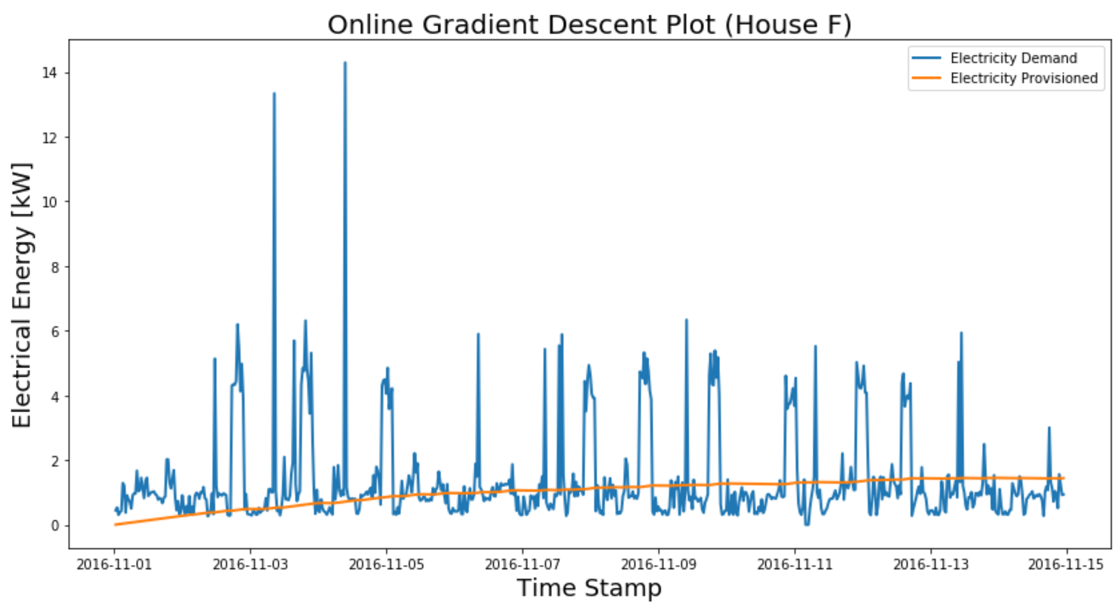
$$x(t+1) = x(t) - learningRate * (d\, f(x(t))/x(t))$$

Learning rate is essential in determining the X value at *t+1* time.
Hence, the Online Gradient Descent was plotted for the most optimal learningRate (eta) values so as to reduce the cost of the Objective Function.

The following are the Online Gradient Descent Plot between Energy demand/provisioned and the time period of 14 days, T=672





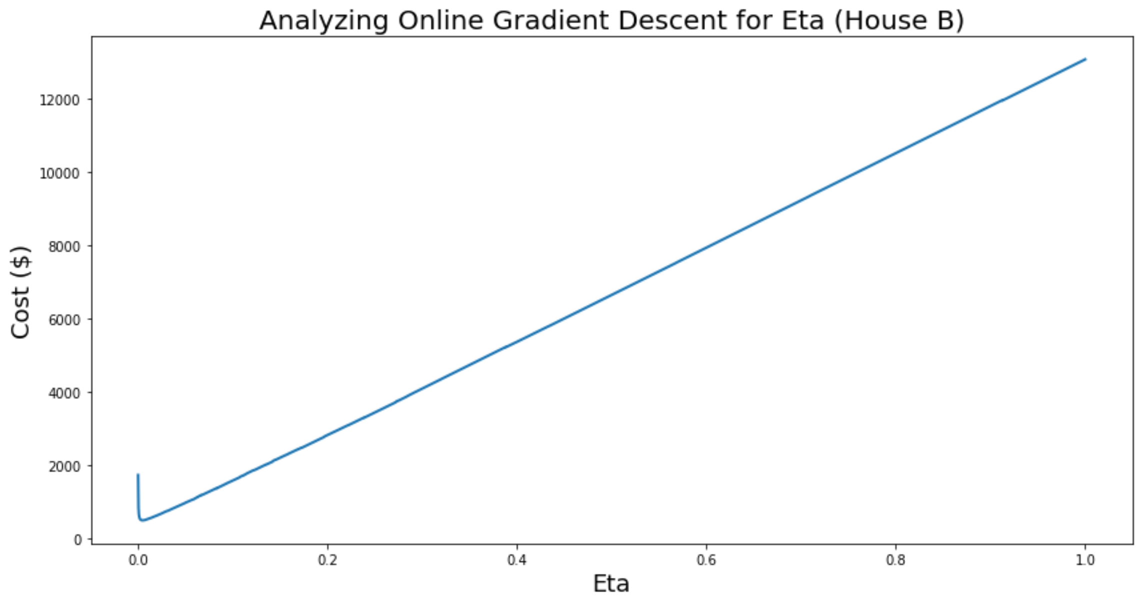


Comparing the Online Gradient Descent Costs for B,C,F Houses

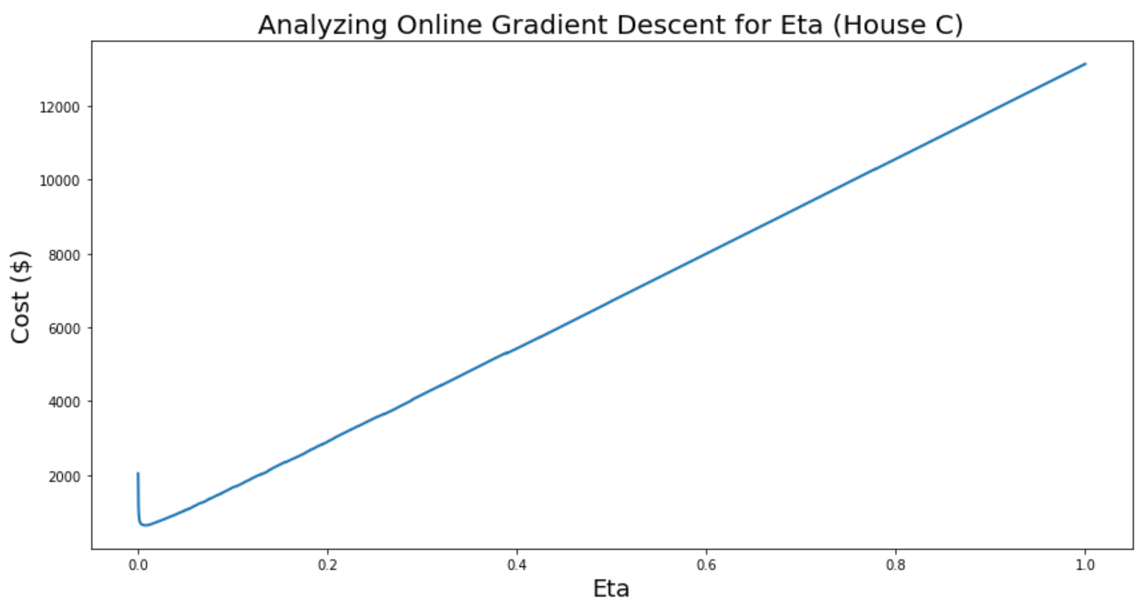
House	Online Gradient Cost ($)
B	480.99
C	639.34
F	1965.38

Sensitivity Analysis for Online Gradient Descent

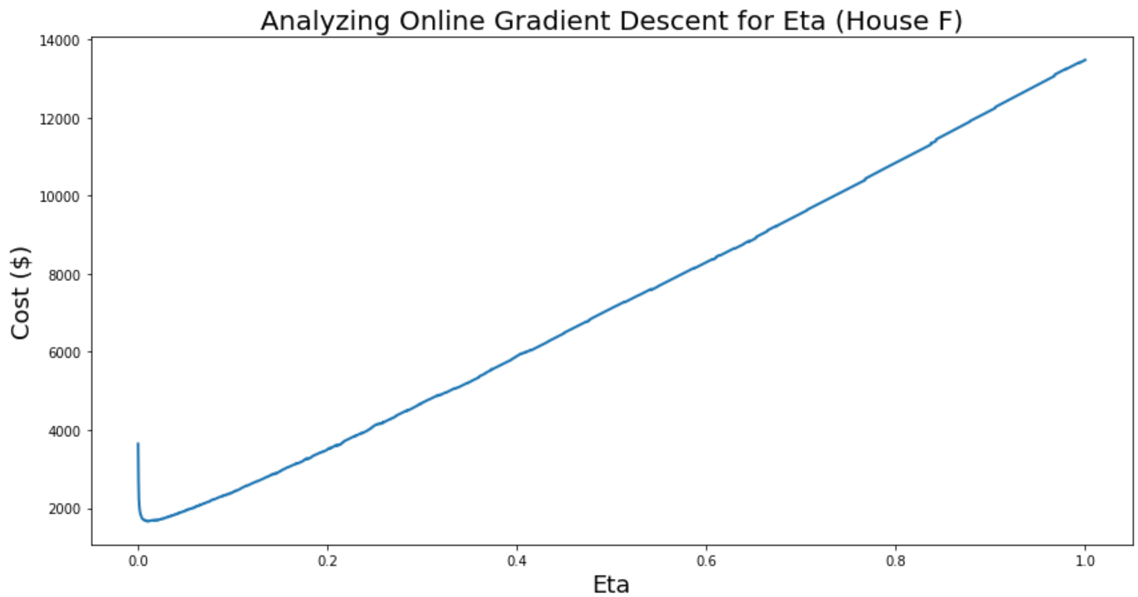
The following are the Online Gradient Descent Plot between Cost ($) and LearningRate



Optimum eta = 0.0046, Optimum Cost = $ 480.99



Optimum eta = 0.0079, Optimum Cost = $ 639.34



Optimum eta = 0.0016, Optimum Cost = $ 1658.98

Receding Horizon Control (RHC)

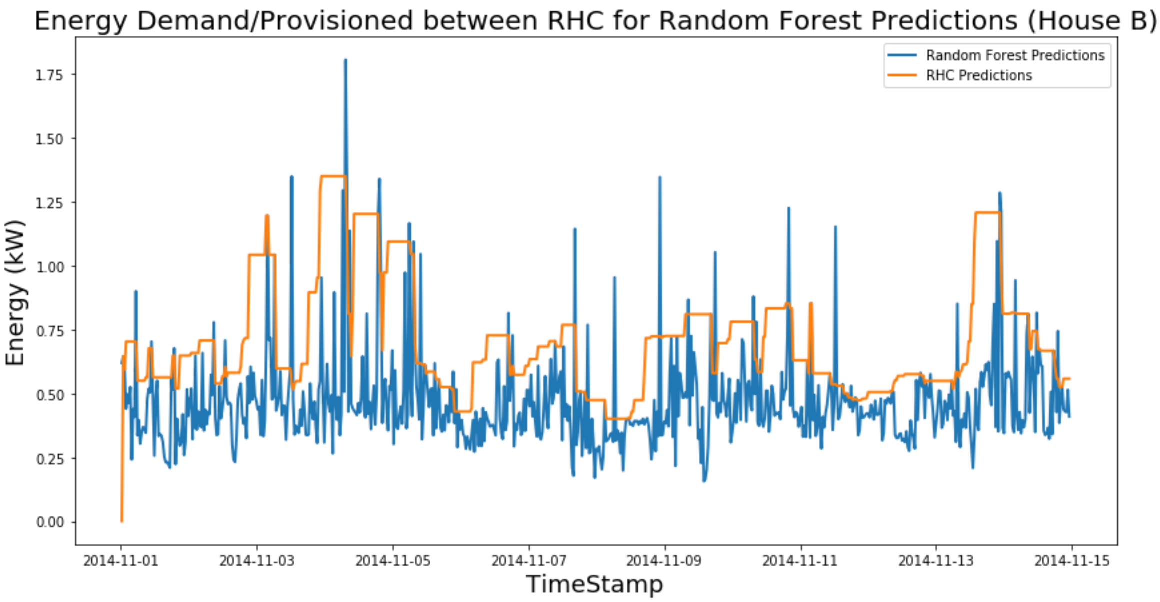
Receding Horizon Control is often used to control the process while satisfying certain constraints (x>=0 in our case). It has been used in chemical plants and oil refineries since the 1980s.

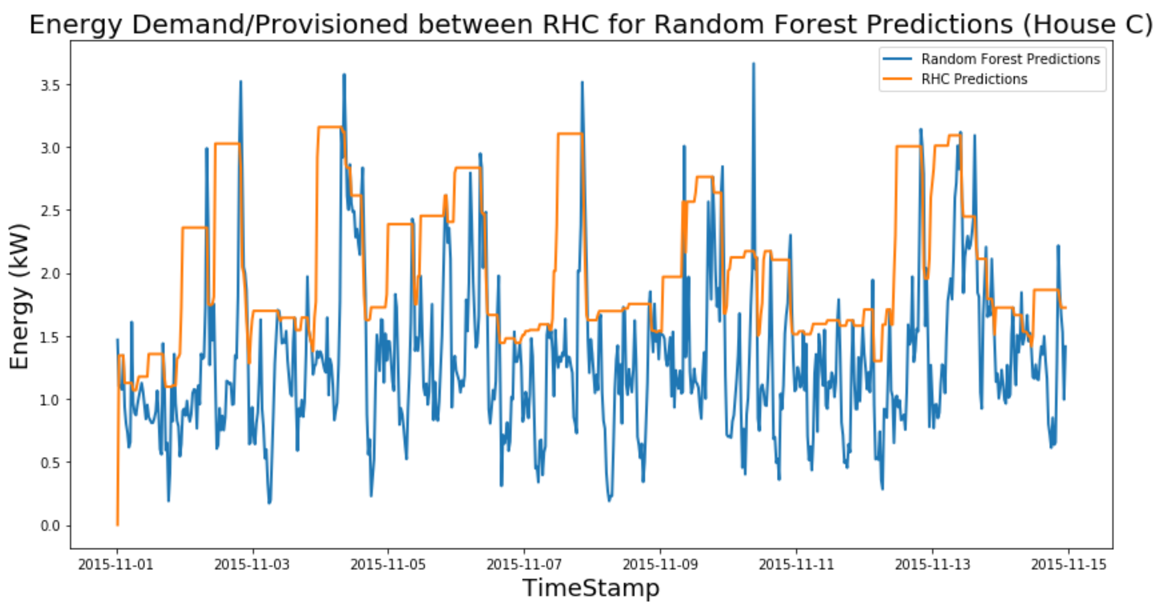
RHC is used with the Arima and Random Forest Algorithm to minimize the objective function value. By doing an analysis with varying 'w' (window size) values, the optimum window was found to be 19.

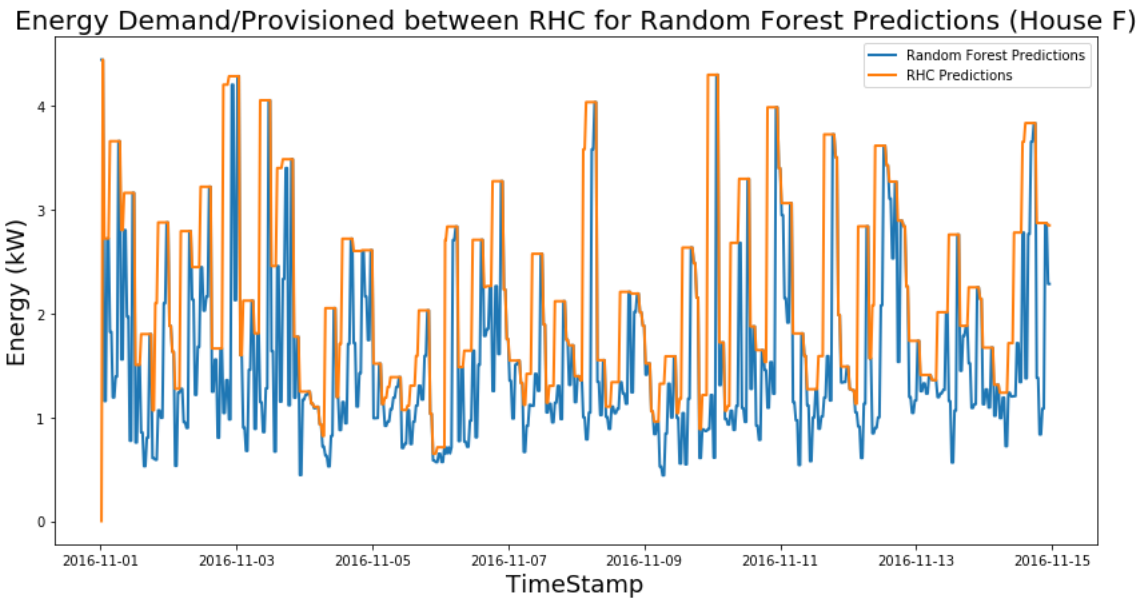
Upon doing the RHC analysis, we came across the fact that House B has the lowest RHC values for the Arima and Random Forest Algorithms.

RHC + Random Forest Prediction

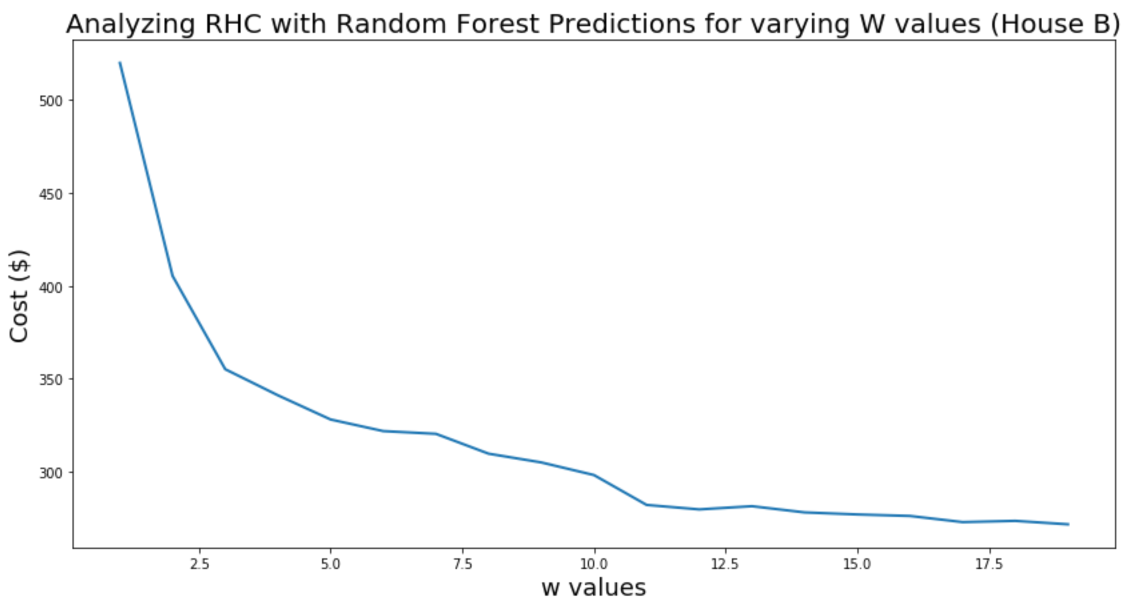
The following are the RHC Plot + Random Forest Predictions between Energy and the time period of 14 days, T=672

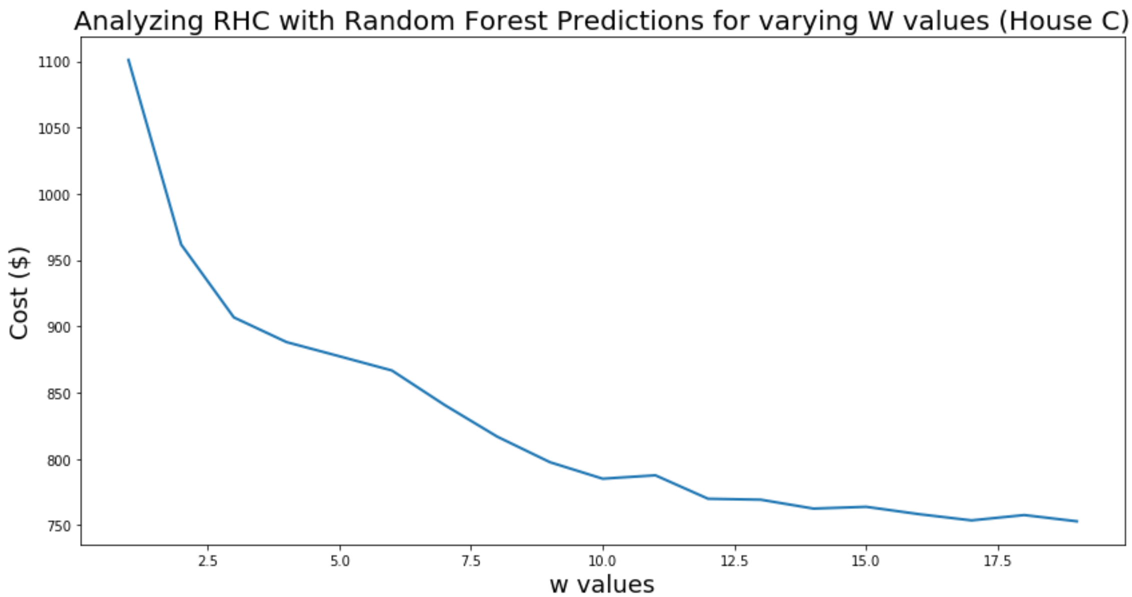

Energy Demand/Provisioned between RHC for Random Forest Predictions (House B)
Random Forest Predictions
RHC Predictions
Energy (kW)
TimeStamp
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2014-11-01
2014-11-03
2014-11-05
2014-11-07
2014-11-09
2014-11-11
2014-11-13
2014-11-15

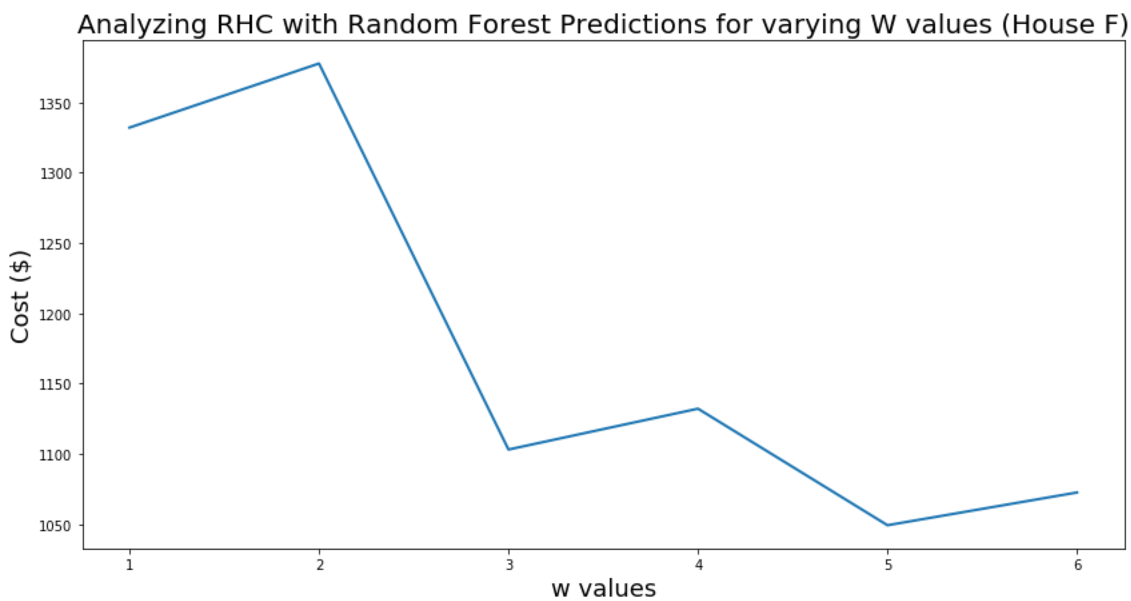

Energy Demand/Provisioned between RHC for Random Forest Predictions (House C)
Random Forest Predictions
RHC Predictions
Energy (kW)
TimeStamp
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
2015-11-01
2015-11-03
2015-11-05
2015-11-07
2015-11-09
2015-11-11
2015-11-13
2015-11-15



Sensitivity Analysis of RHC + Random Forest

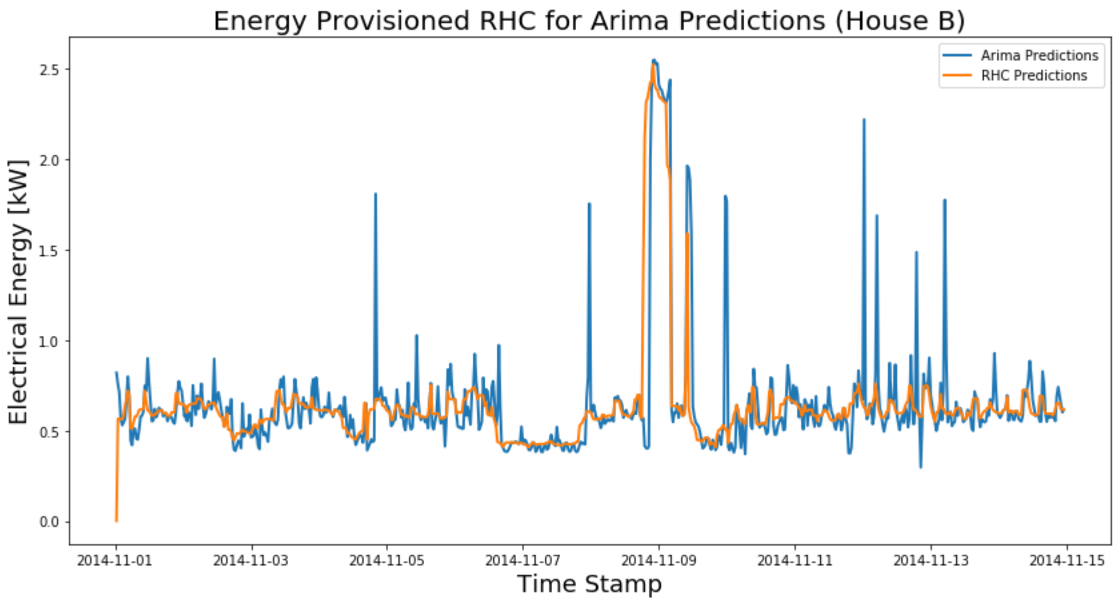


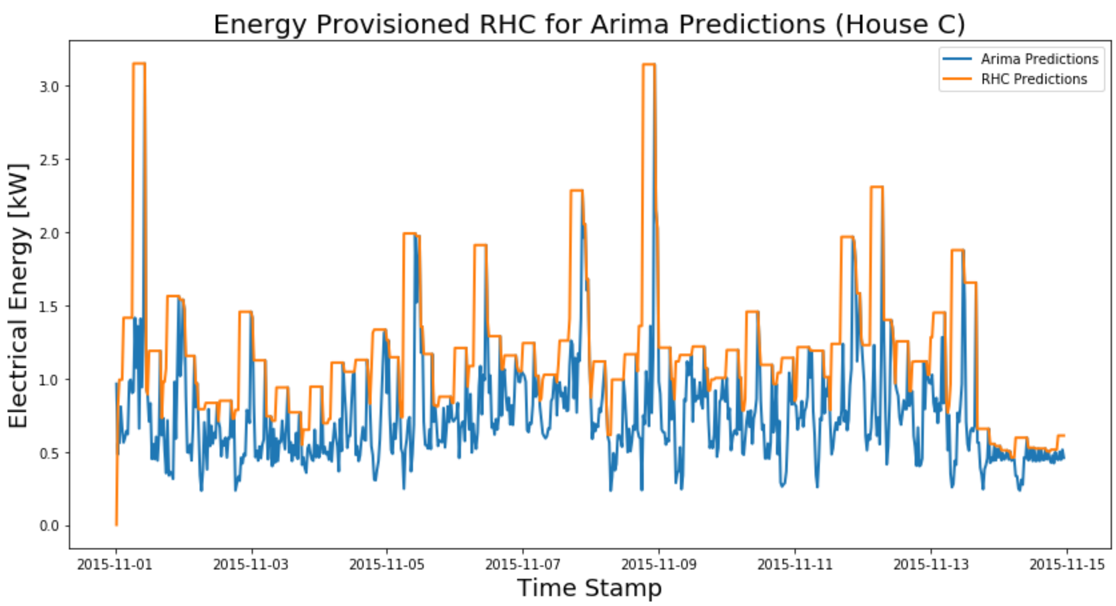


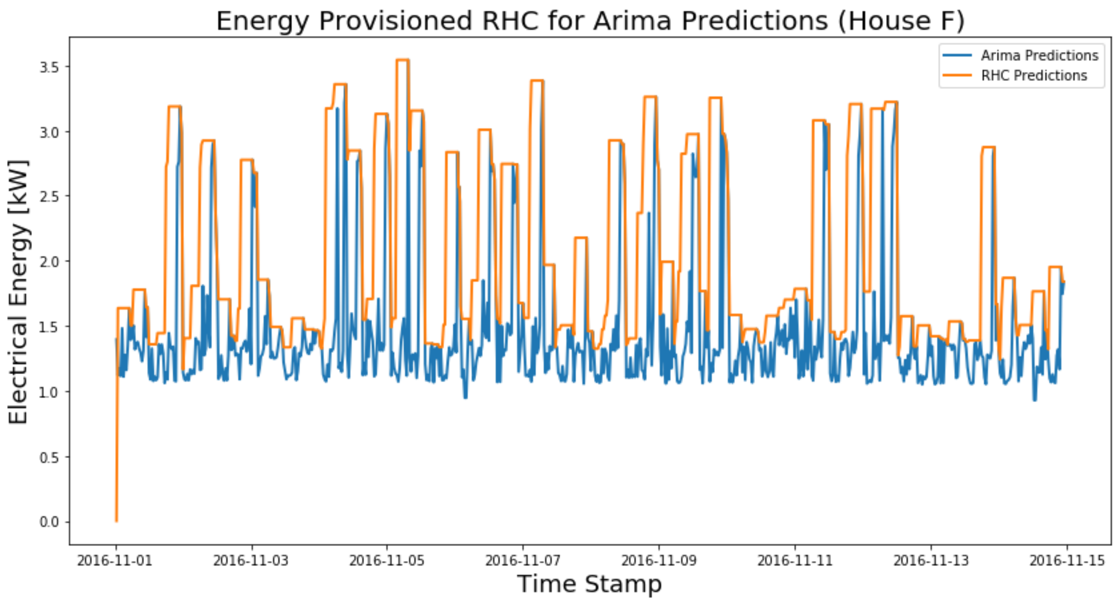


RHC + Arima Prediction

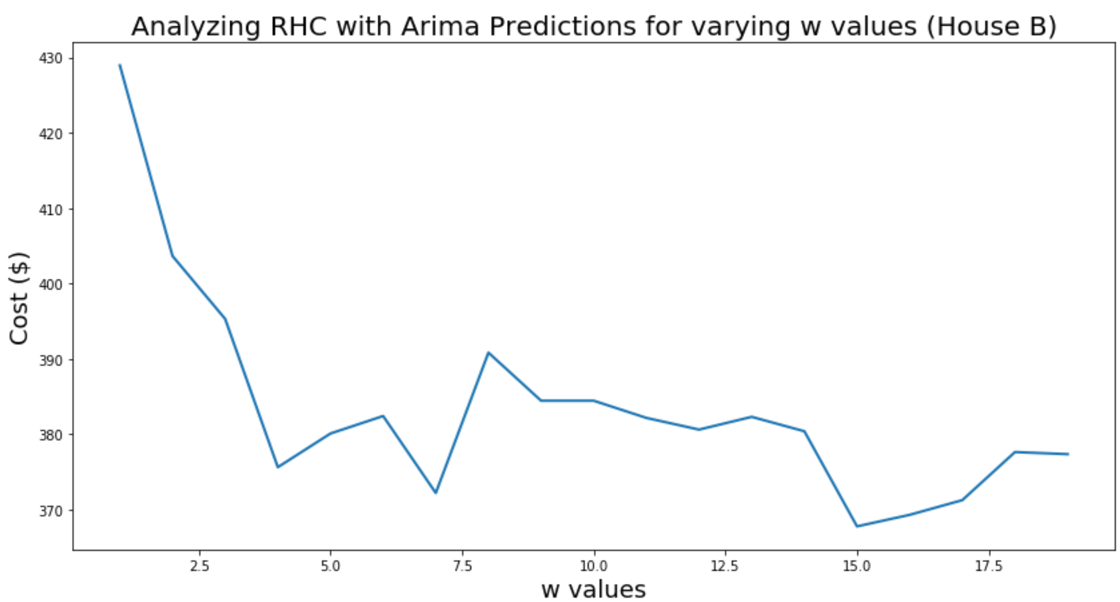
The following are the RHC Plot + Arima Predictions between Energy and the time period of 14 days, T=672

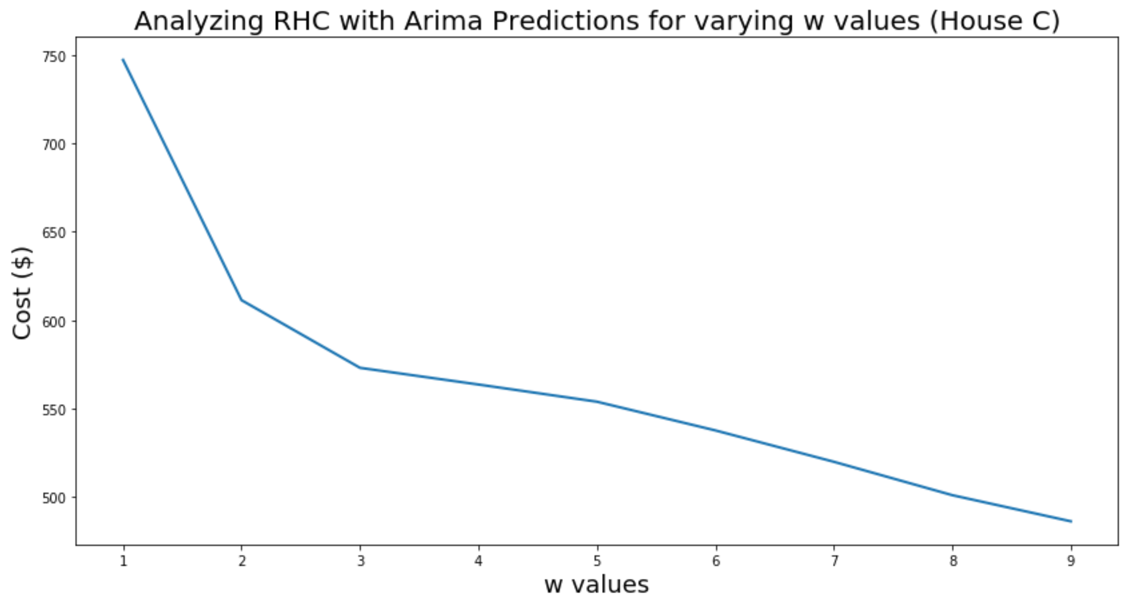

Energy Provisioned RHC for Arima Predictions (House B)
Arima Predictions
RHC Predictions
Electrical Energy [kW]
Time Stamp
2014-11-01
2014-11-03
2014-11-05
2014-11-07
2014-11-09
2014-11-11
2014-11-13
2014-11-15

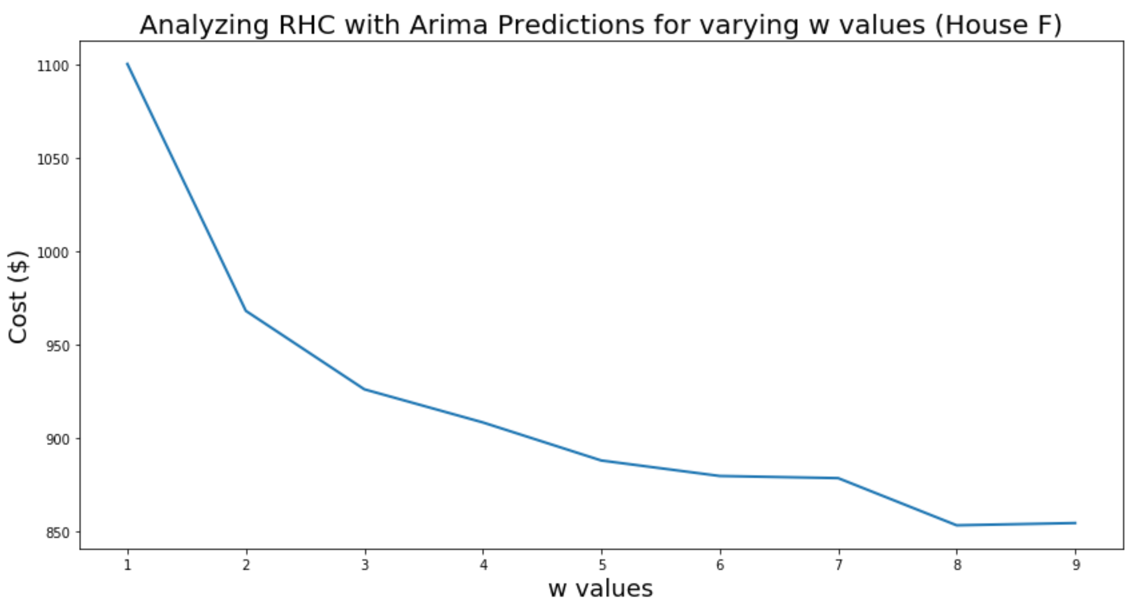

Energy Provisioned RHC for Arima Predictions (House C)
Arima Predictions
RHC Predictions
Electrical Energy [kW]
Time Stamp
2015-11-01
2015-11-03
2015-11-05
2015-11-07
2015-11-09
2015-11-11
2015-11-13
2015-11-15



Sensitivity Analysis for RHC + Arima







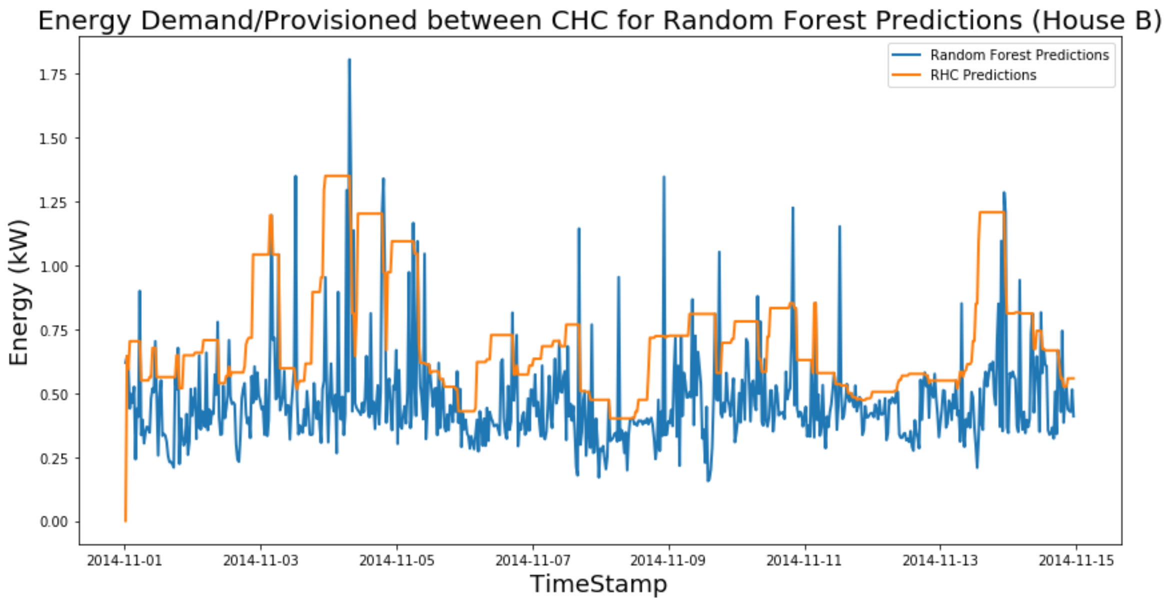
Commitment Horizon Control (CHC)

Running CHC involves finding the best window size and the commitment window size in order to reduce the objective function cost.

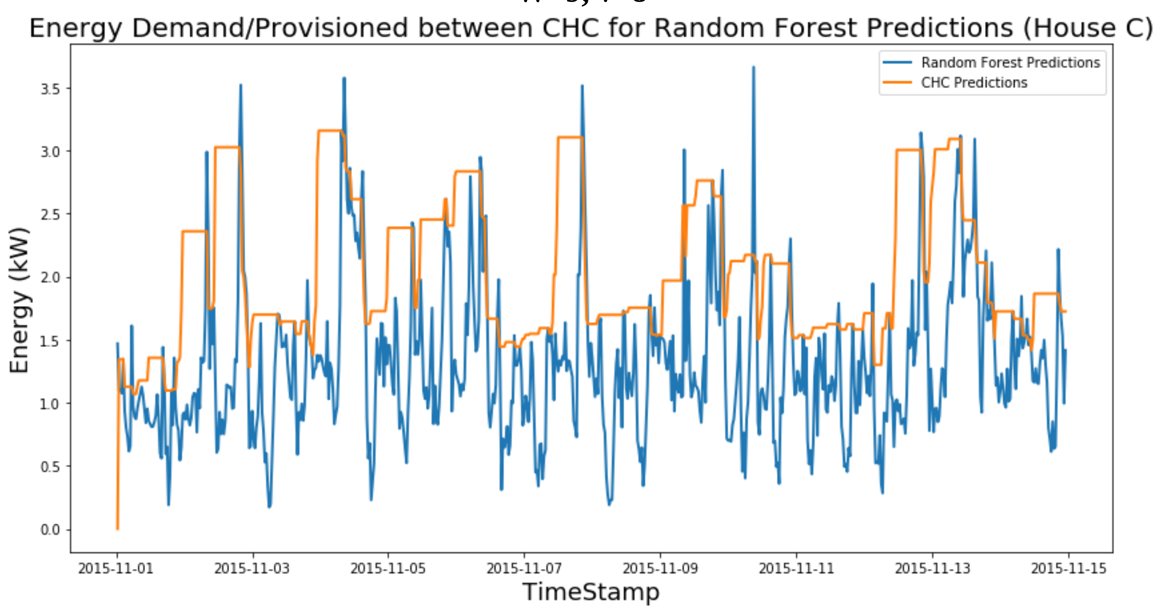
CHC + Random Forest Prediction

The following are the CHC Plot + Random Forest Predictions between Energy and the time period of 14 days, T=672

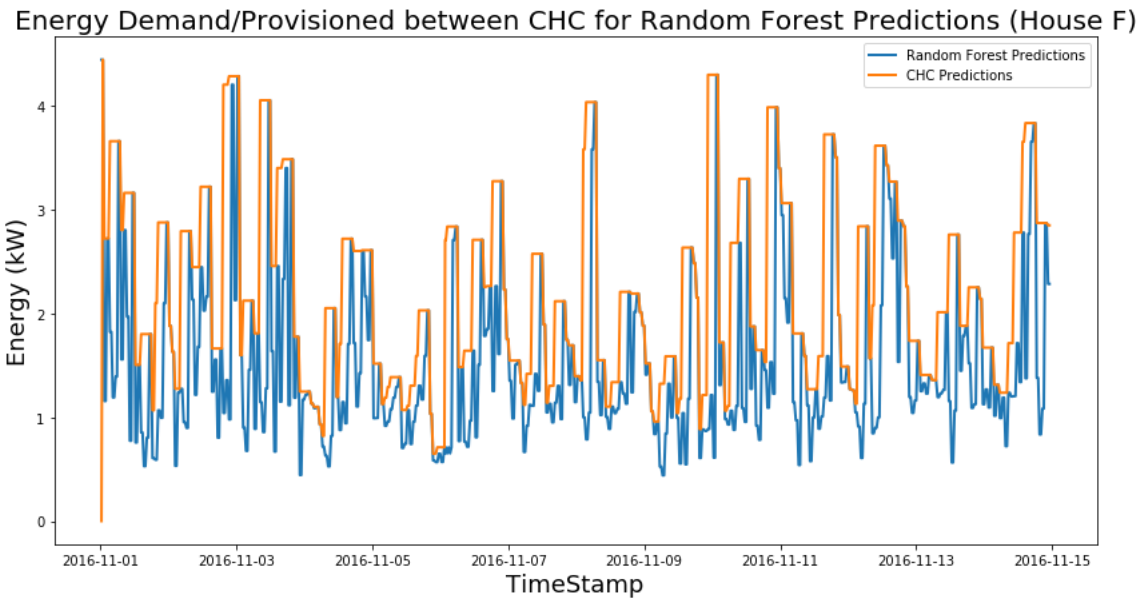
W = 19, v =16



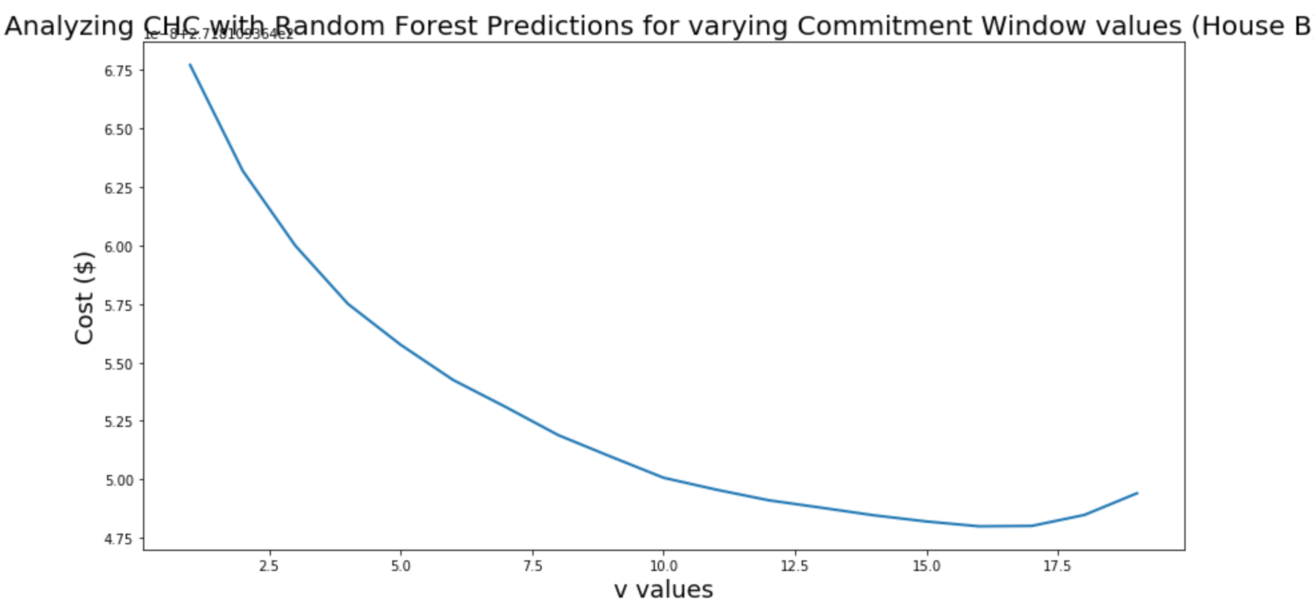
W=9, v=8



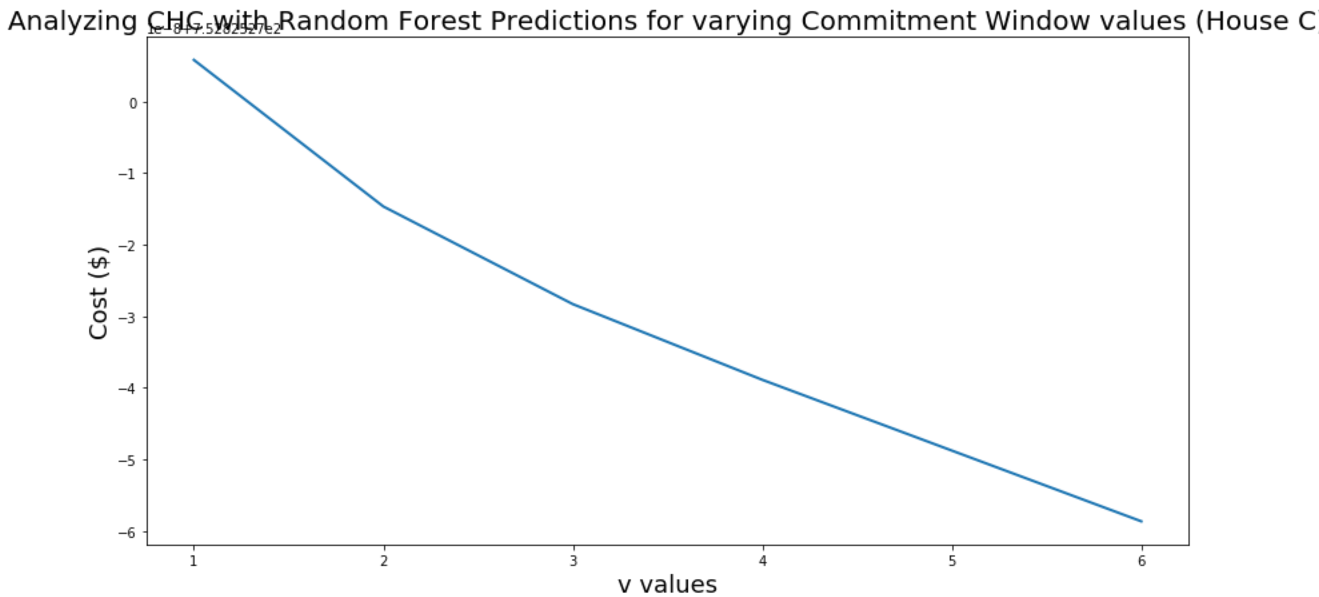
W = 9, v=8



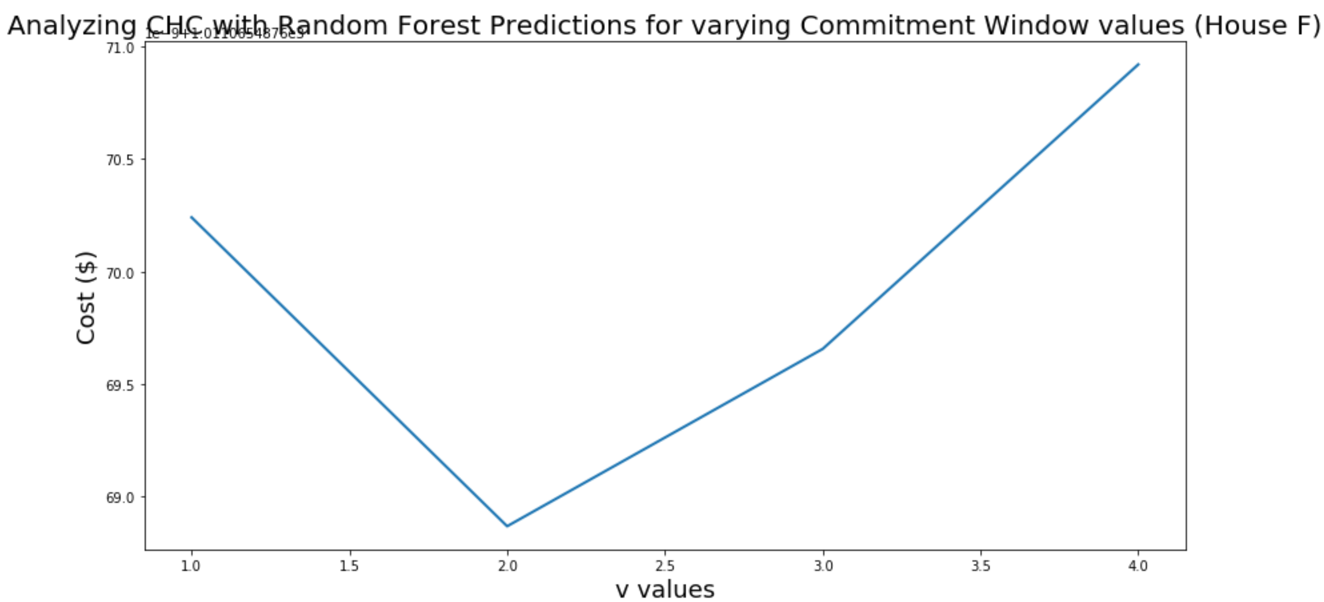
<u>Sensitivity Analysis for CHC + Random Forest</u>



Optimum V =16



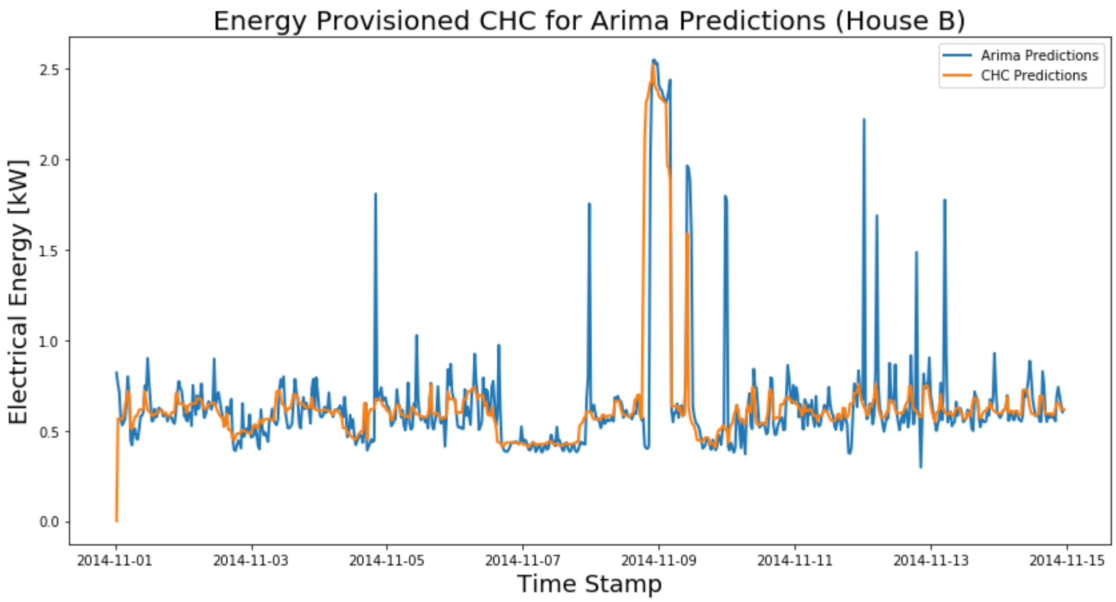
Optimum V =6

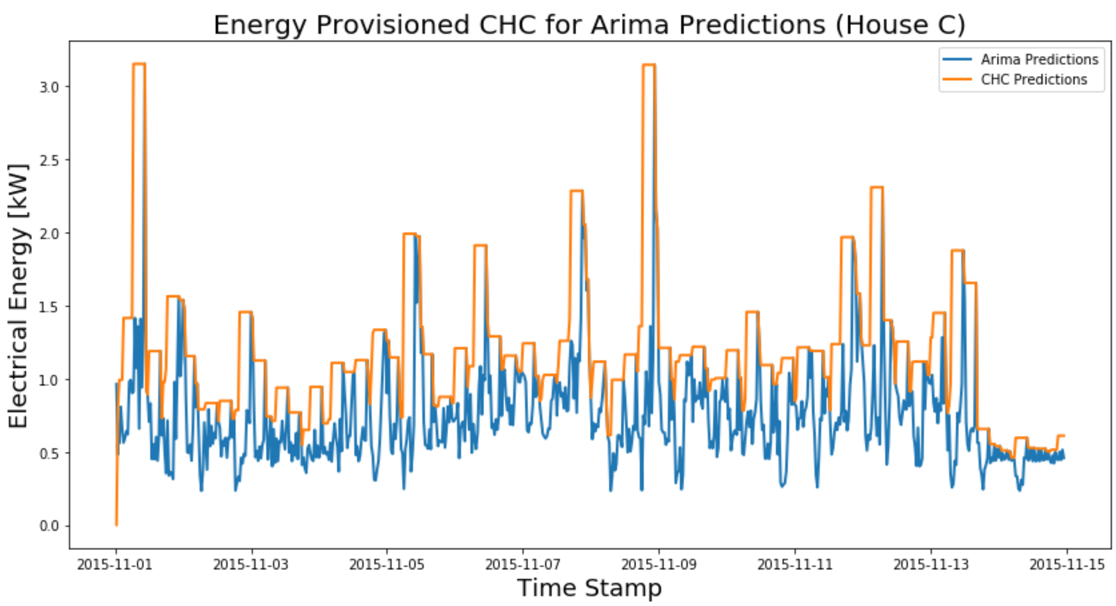


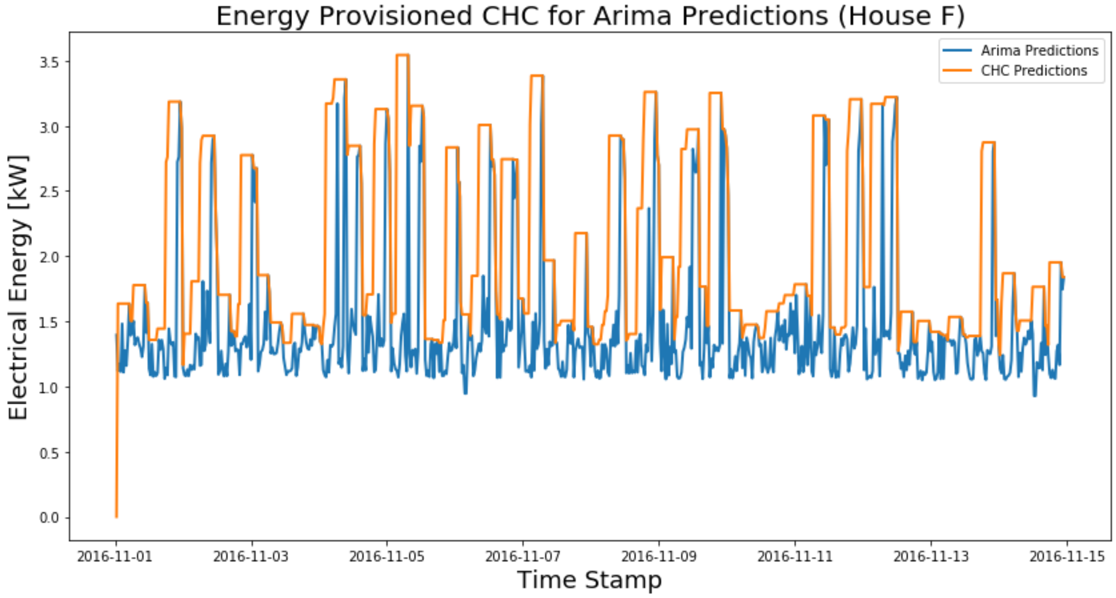
Optimum V = 2

CHC + Arima Prediction

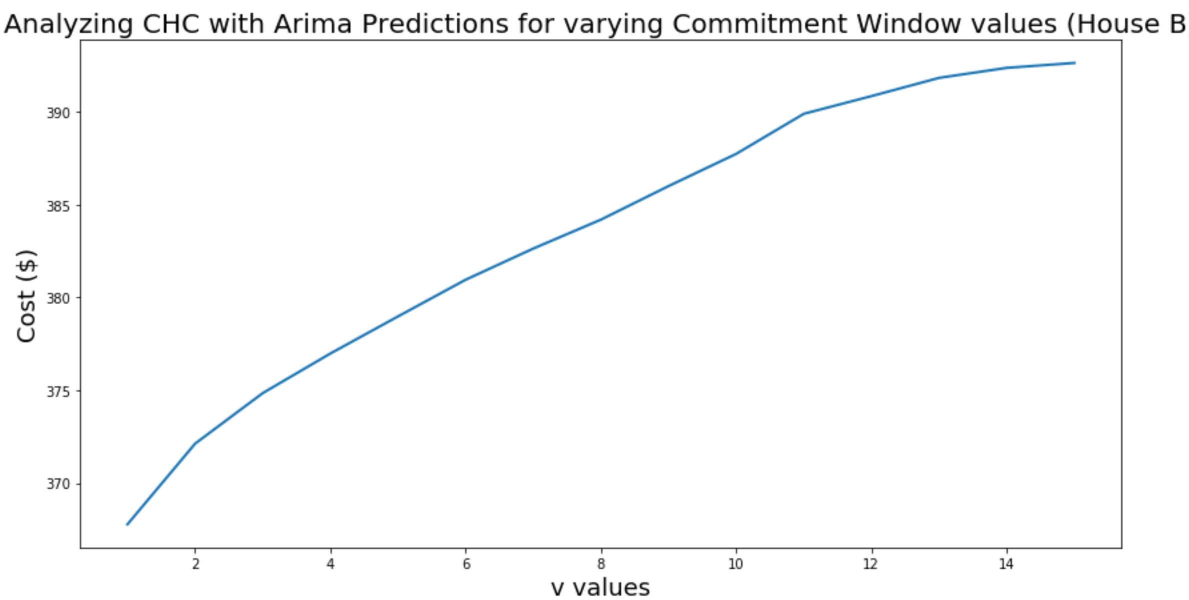
The following are the CHC Plot + Random Forest Predictions between Energy and the time period of 14 days, T=672

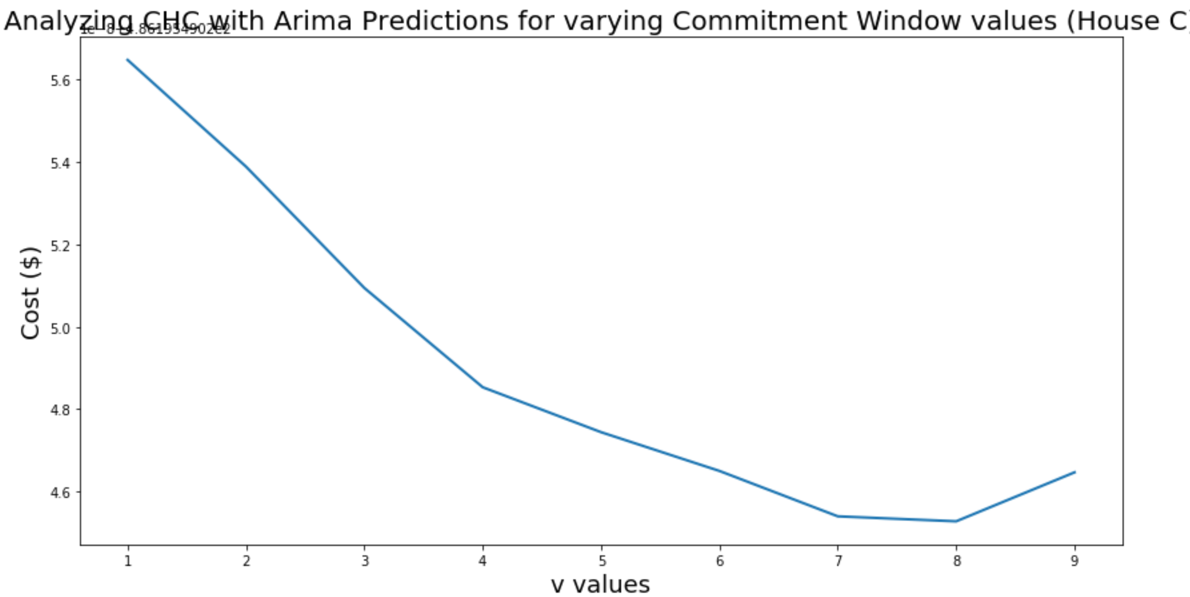

Energy Provisioned CHC for Arima Predictions (House B)
Arima Predictions
CHC Predictions
Electrical Energy [kW]
Time Stamp
2014-11-01
2014-11-03
2014-11-05
2014-11-07
2014-11-09
2014-11-11
2014-11-13
2014-11-15

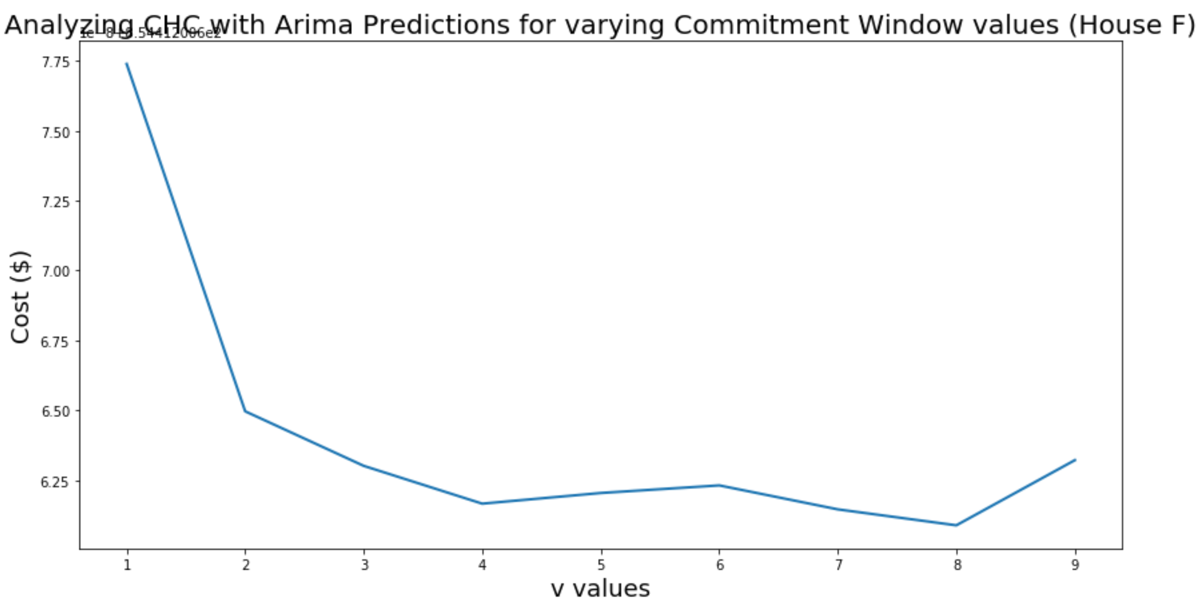

Energy Provisioned CHC for Arima Predictions (House C)
Arima Predictions
CHC Predictions
Electrical Energy [kW]
Time Stamp
2015-11-01
2015-11-03
2015-11-05
2015-11-07
2015-11-09
2015-11-11
2015-11-13
2015-11-15



<u>Sensitivity Analysis for CHC + Arima</u>



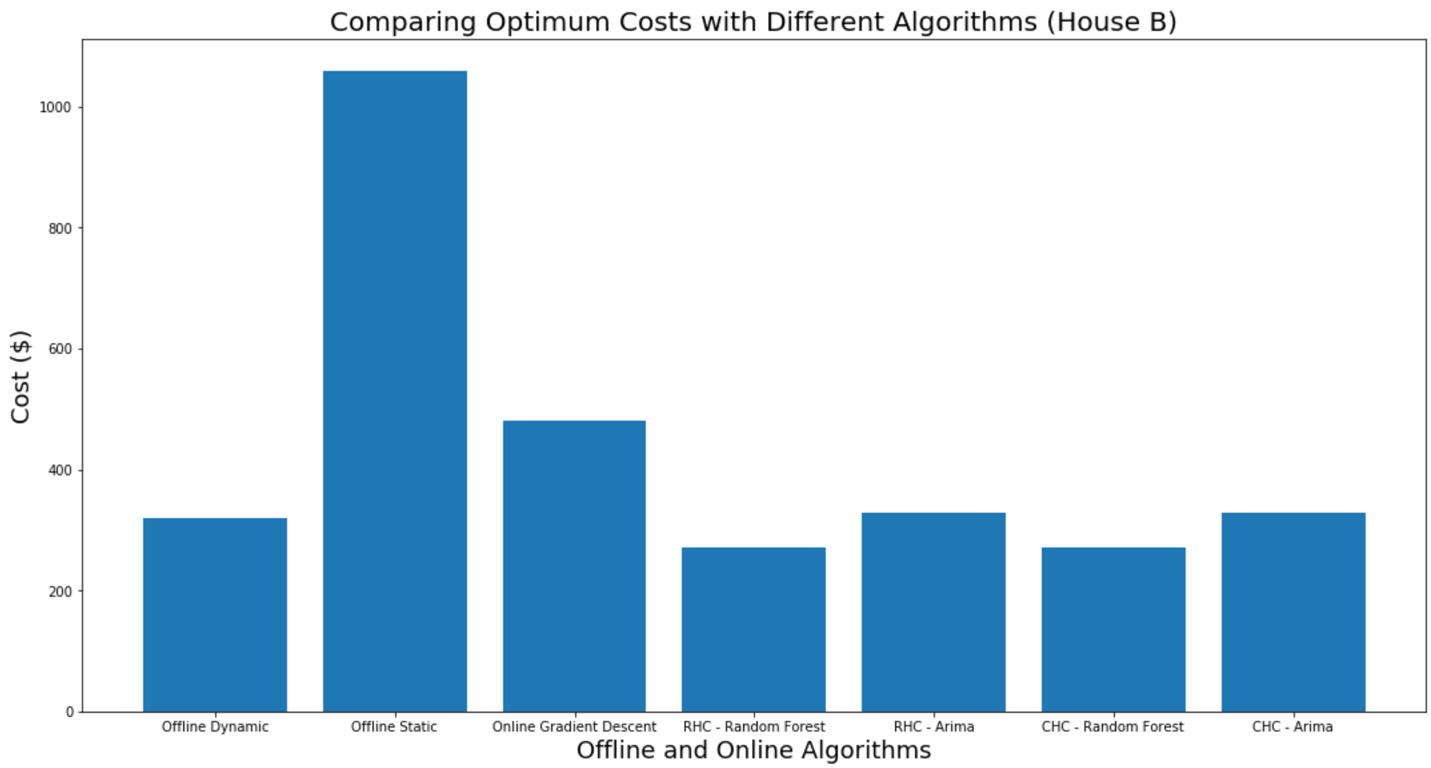

Analyzing CHC with Arima Predictions for varying Commitment Window values (House C)
5.6
5.4
5.2
5.0
4.8
4.6
Cost ($)
1
2
3
4
5
6
7
8
9
v values


Analyzing CHC with Arima Predictions for varying Commitment Window values (House F)
7.75
7.50
7.25
7.00
6.75
6.50
6.25
Cost ($)
1
2
3
4
5
6
7
8
9
v values

TASK 3

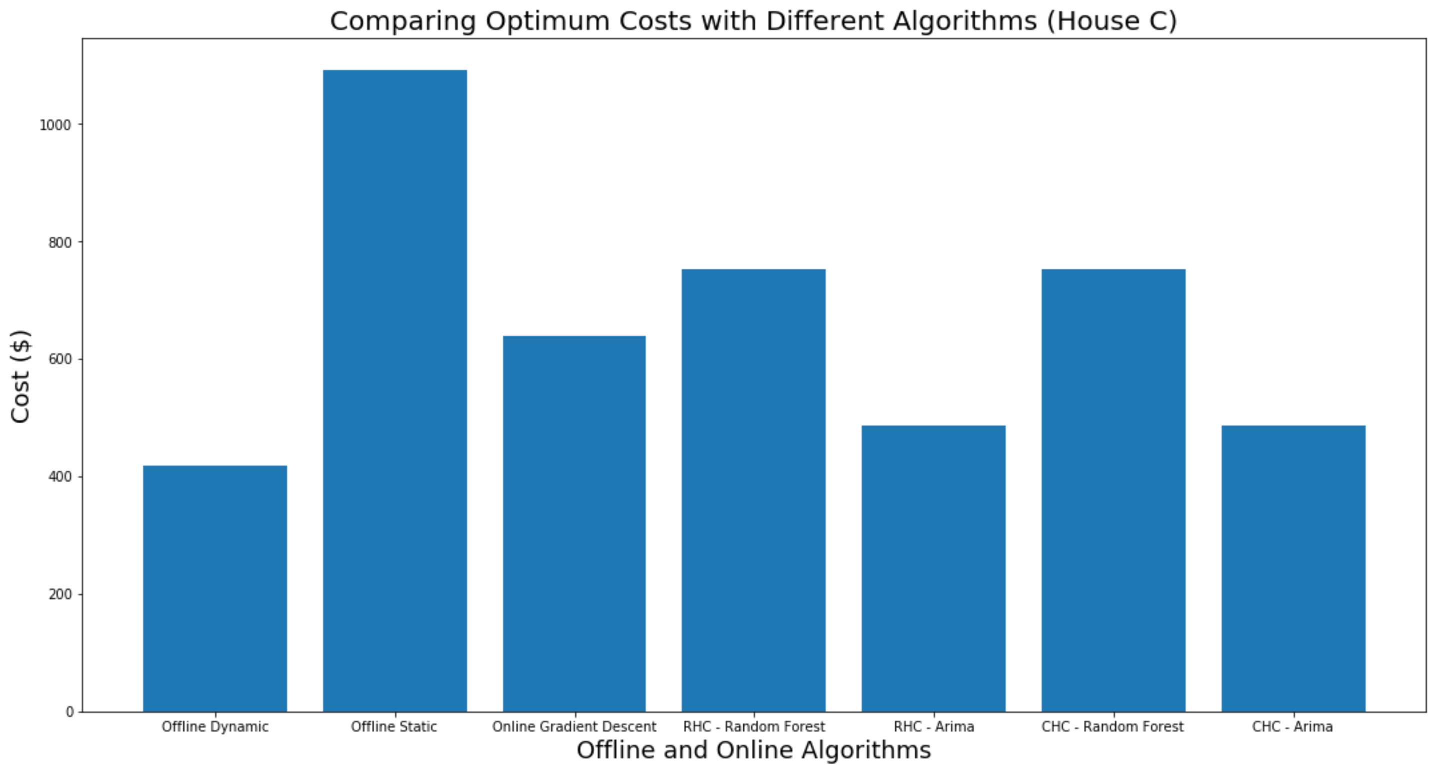
Comparing Costs of different provisioning algorithms

House B, Best Algorithm = CHC + random forest



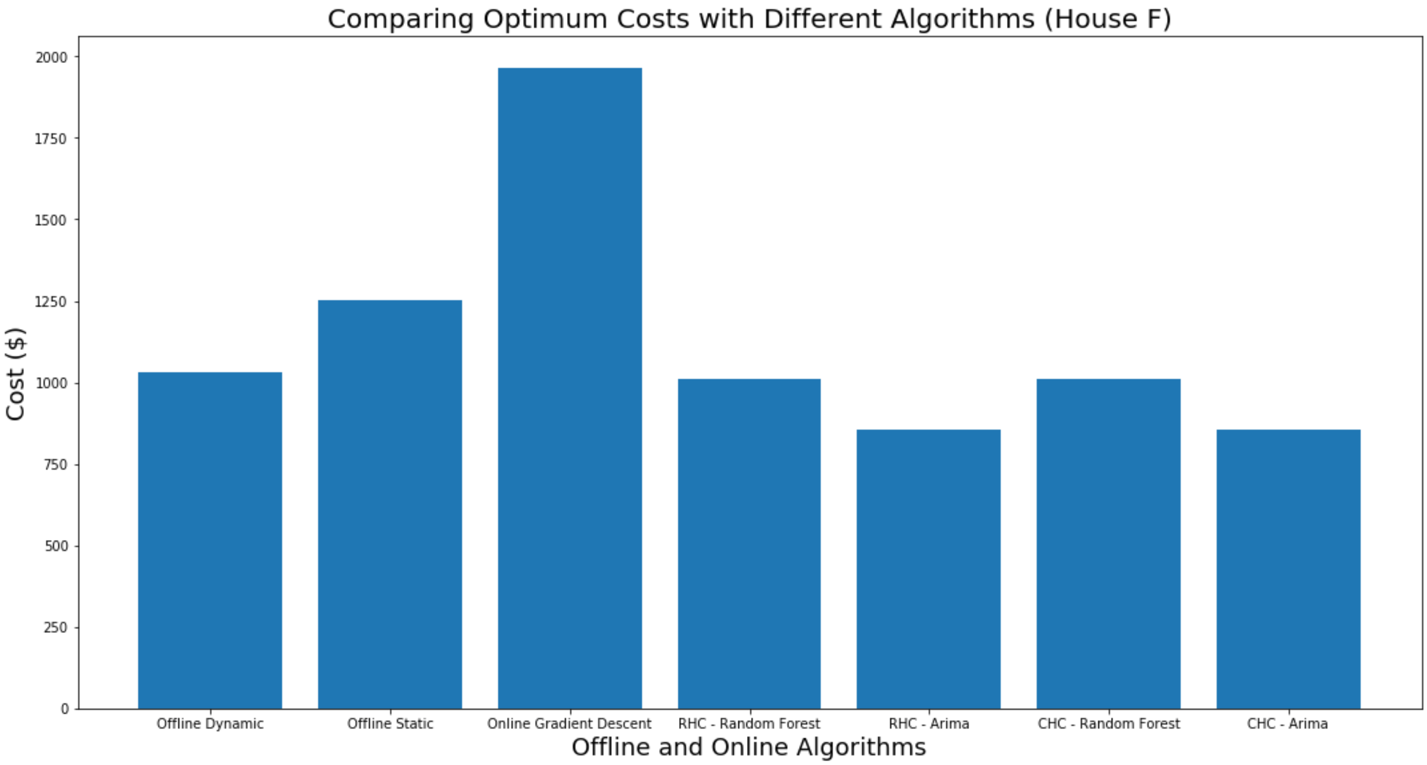
	Algorithm	Cost
0	Offline Dynamic	319.680000
1	Offline Static	1059.700000
2	Online Gradient Descent	480.990000
3	RHC - Random Forest	271.810936
4	RHC - Arima	328.750000
5	CHC - Random Forest	271.810000
6	CHC - Arima	328.750000

House C, Best Algorithm = CHC + Arima



	Algorithm	Cost
0	Offline Dynamic	417.62000
1	Offline Static	1091.84000
2	Online Gradient Descent	639.34000
3	RHC - Random Forest	752.82527
4	RHC - Arima	486.19549
5	CHC - Random Forest	752.83000
6	CHC - Arima	486.19549

House F, Best Algorithm = CHC + Arima

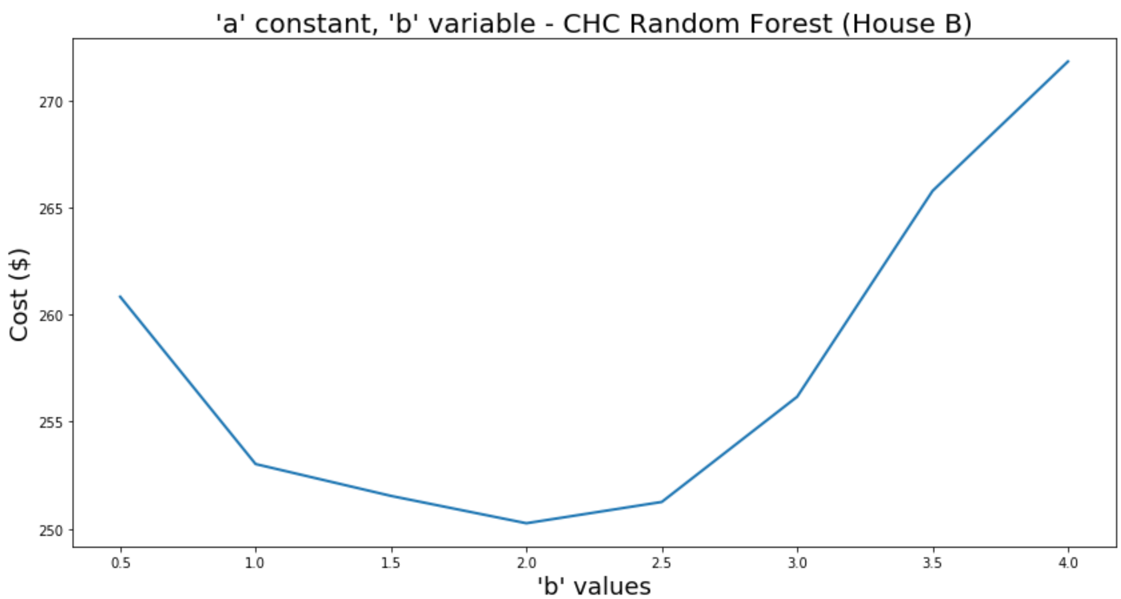


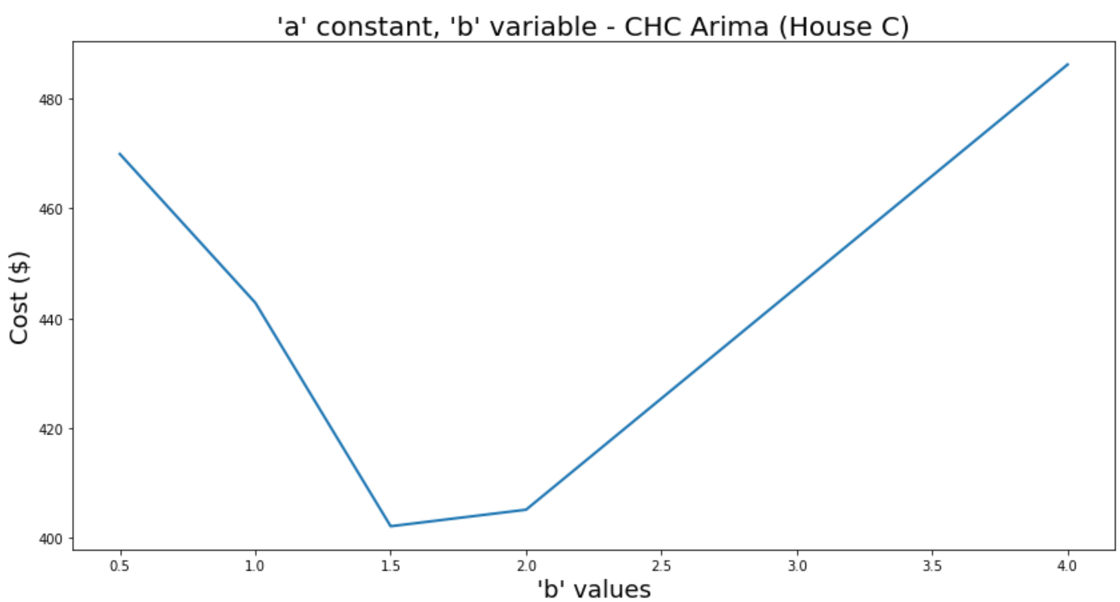
	Algorithm	Cost
0	Offline Dynamic	1029.770000
1	Offline Static	1252.080000
2	Online Gradient Descent	1965.380000
3	RHC - Random Forest	1011.065488
4	RHC - Arima	854.412006
5	CHC - Random Forest	1011.070000
6	CHC - Arima	854.412006

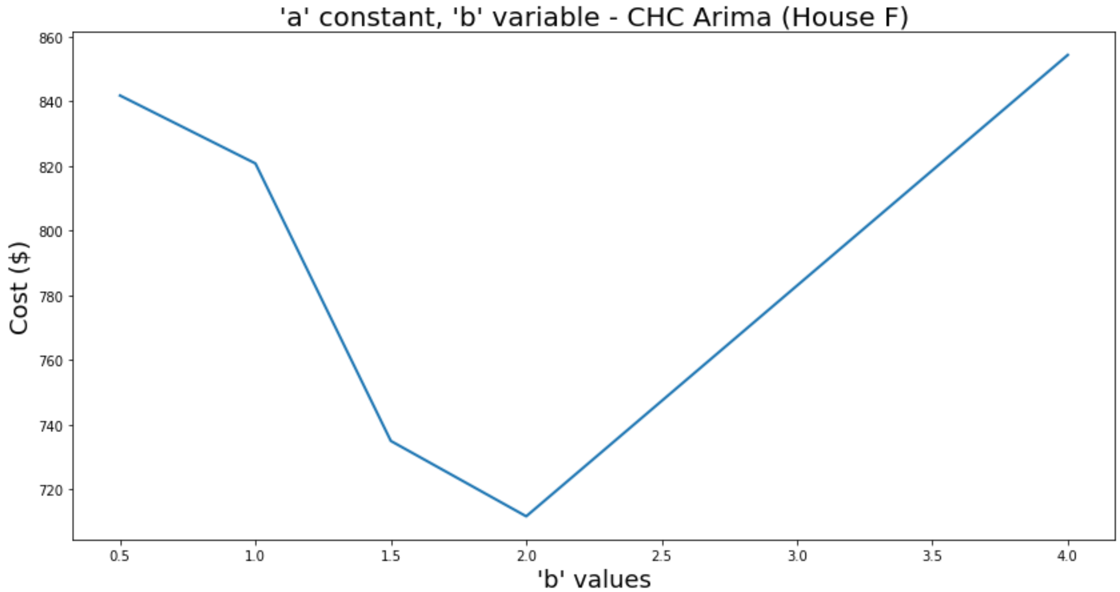
TASK 4

Vary 'a' and 'b' values for the best algorithm

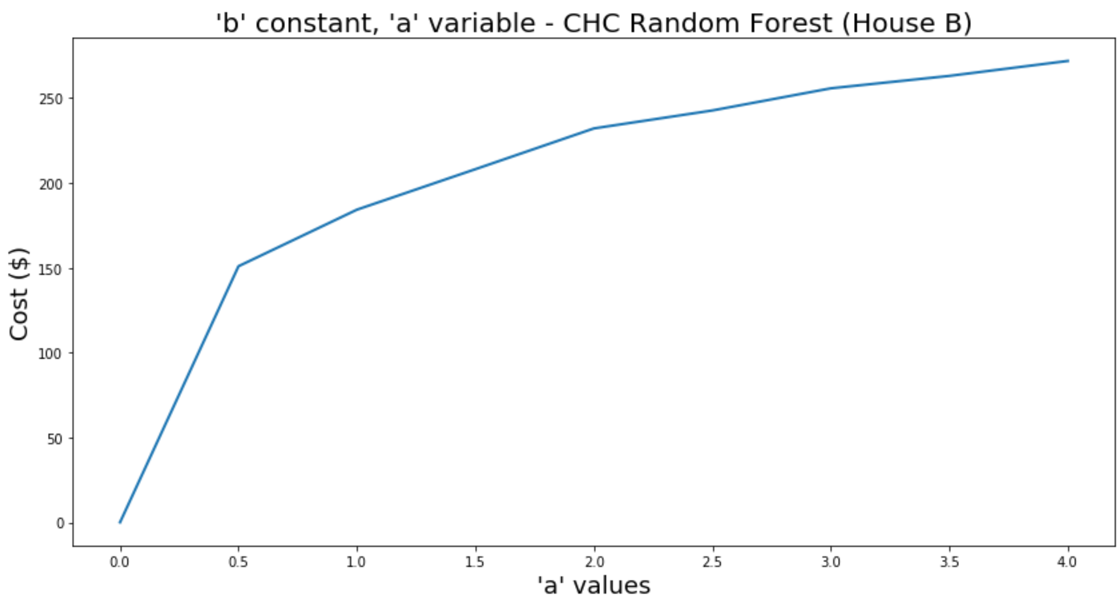
Constant 'a' , Variable 'b'

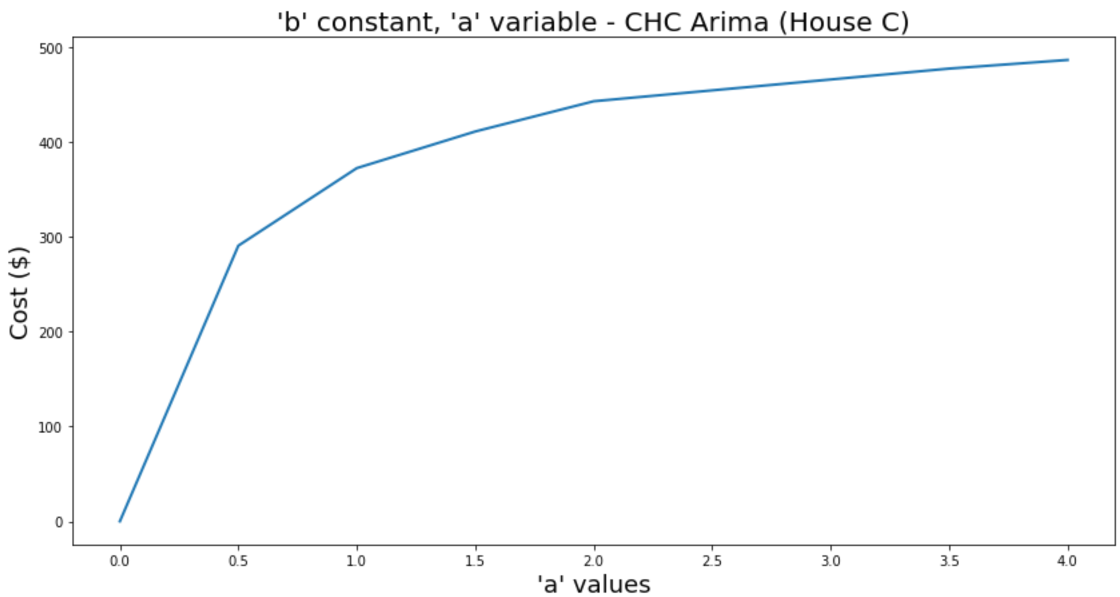


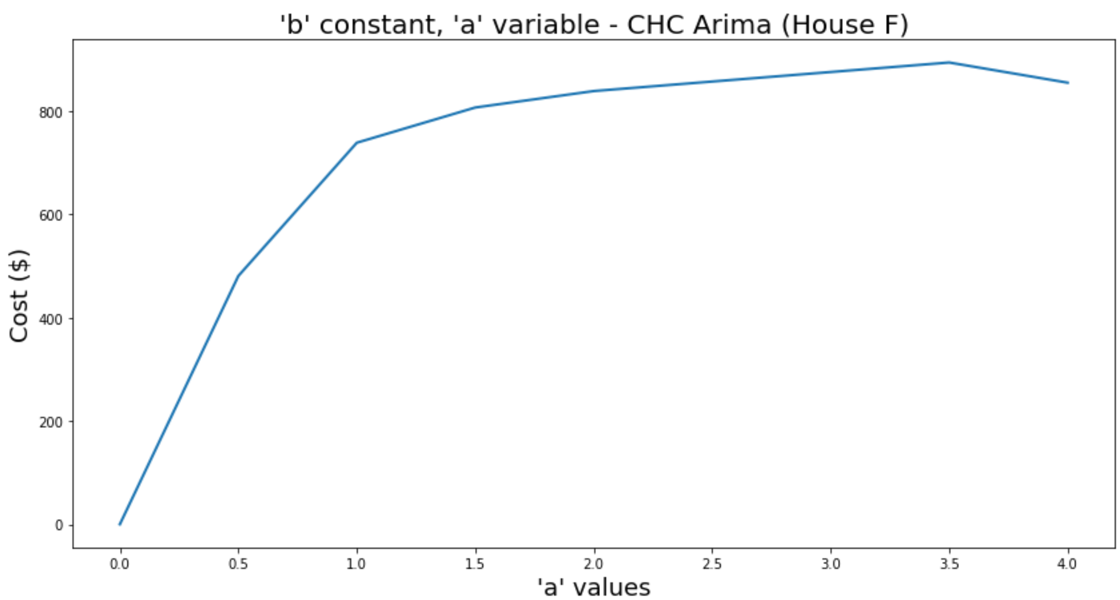




Constant ‘b’ , Variable ‘a’




'b' constant, 'a' variable - CHC Arima (House C)
Cost ($)
'a' values
0
100
200
300
400
500
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


'b' constant, 'a' variable - CHC Arima (House F)
Cost ($)
'a' values
0
200
400
600
800
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

TASK 5

Deterministic and randomized algorithm selection

To select the best algorithm, 'REGRET' heuristic is being used here.

Deterministic Algorithm (Weighted Majority Algorithm)

Steps:

1. Set up initial weights for each agent (3 in our case)
2. Pick a window size (4 in our case)
3. Iterate over the time frame of T=672
4. In a given window, at each point of time, get the best algorithm to be used based upon the difference between the control algorithm cost and the offline dynamic cost (**REGRET**)
5. In the window, if the truth value at time t is not found at index i, then penalize the agent by multiplying its weight by (1-epsilon).
6. Epsilon is calculated using the formula sqrt(log(N)/ **mistakes** by agent i)
7. A mistake by a given agent means the agent was not able to correctly predict the truth value.
8. At a given point of time, the algorithm to be picked will be selected by the agent with the **highest weight.**
9. The total cost of the deterministic algorithm is found by summing up the cost at the respective index by a given control algorithm.

$$W_t(A) \triangleq \sum_{i \in S_t(A)} w_t(i), \qquad W_t(B) \triangleq \sum_{i \in S_t(B)} w_t(i)$$

Choose A if $W_t(A) \geq W_t(B)$, or B otherwise.

Update $w_{t+1}(i) = \begin{cases} w_t(i), & \text{if expert } i \text{ is correct.} \\ w_t(i)(1-\varepsilon) & \text{otherwise.} \end{cases}$

```
House B => Total Cost by Deterministic Algorithm = $ 163.26
House C => Total Cost = $ 262.24
House F => Total Cost = $ 515.47
```

Randomized Algorithm (Randomized Weighted Majority Algorithm)

Steps:

1. Set up initial weights for each agent (3 in our case)
2. Pick a window size (4 in our case)
3. Iterate over the time frame of T=672
4. In a given window, at each point of time, get the best algorithm to be used based upon the difference between the control algorithm cost and the offline dynamic cost (**REGRET**)
5. In the window, if the truth value at time t is not found at index i, then penalize the agent by multiplying its weight by (1-epsilon).
6. Epsilon is calculated using the formula sqrt(log(N)/ **mistakes** by agent i)
7. A mistake by a given agent means the agent was not able to correctly predict the truth value.
8. The difference:
   Probabilities for each agent is found by dividing the agent weight by the sum of weights and sorting them in ascending order. Then a cumulative probabilities array is formed.
   A **random number** is generated and the interval of the cumulative probabilities wherein it lies gets the agent id and hence, decides the online algorithm which should be used to evaluate the cost at that point of time.

9. The total cost of the randomized algorithm is found by summing up the cost at the respective index by a given control algorithm.

Weights of the agent found:

```
House B => Final Weights of the 3 agents:  [0.85, 0.68, 0.92]
Total Cost by Randomized Algorithm = $ 177.09

House C => Final Weights of the 3 agents:  [0.64, 0.77, 0.98]
Total Cost by Randomized Algorithm = $ 265.64

House F => Final Weights of the 3 agents:  [1.0, 0.58, 0.97]
Total Cost by Randomized Algorithm = $ 537.24
```

References

- https://en.wikipedia.org/wiki/Model_predictive_control
- https://stackoverflow.com/questions/12444716/how-do-i-set-the-figure-title-and-axes-labels-font-size-in-matplotlib
- https://matplotlib.org/3.2.1/api/_as_gen/matplotlib.pyplot.bar.html
- https://www.cvxpy.org/_modules/cvxpy/atoms/elementwise/exp.html