AMS - SMART ENERGY

GROUP - 14
Mitesh Kumar Singh
Sweta Kumari
Varun Agarwal

Electricity Provisioning based on Demand Prediction

December 1st, 2018

OVERVIEW

We have to test the performance of different electricity provisioning algorithms based on the demand predictions that we already have. We will be using the below objective function for performance measurement.

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

We are using values as below, unless stated otherwise.

a = b = $4/*kWh*

p = $0.40/*kWh*

GOALS

1. Solve the offline optimization problem, e.g., using tools CVX in Matlab.
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.

1. OFFLINE OPTIMIZATION PROBLEM

In offline optimization problem, we solve it in two ways - static and dynamic.

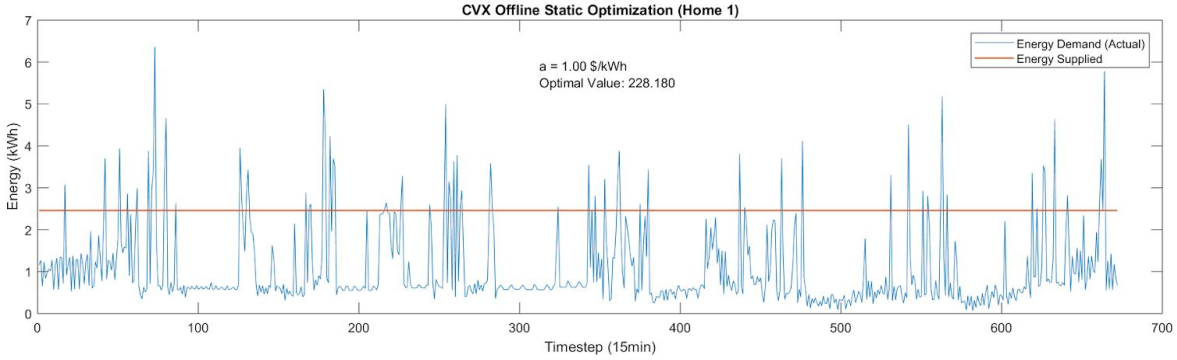
House	Offline Static	Offline Dynamic
1	228.18	196.91
2	266.18	225.21
3	230.51	195.4
4	204.75	165.34
5	238.5	203.62
6	301.8	243.53
7	277.53	244.49
8	275.24	248.79
9	194.65	181.87
10	196.96	169.15

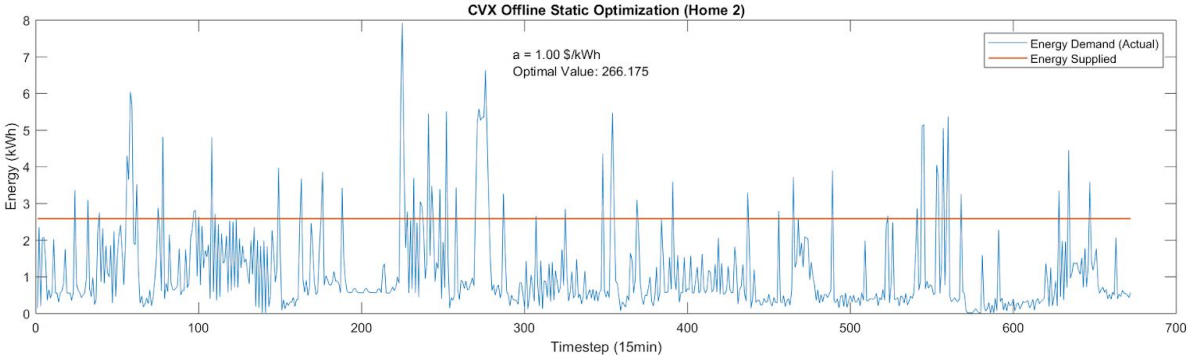
Static Offline optimization

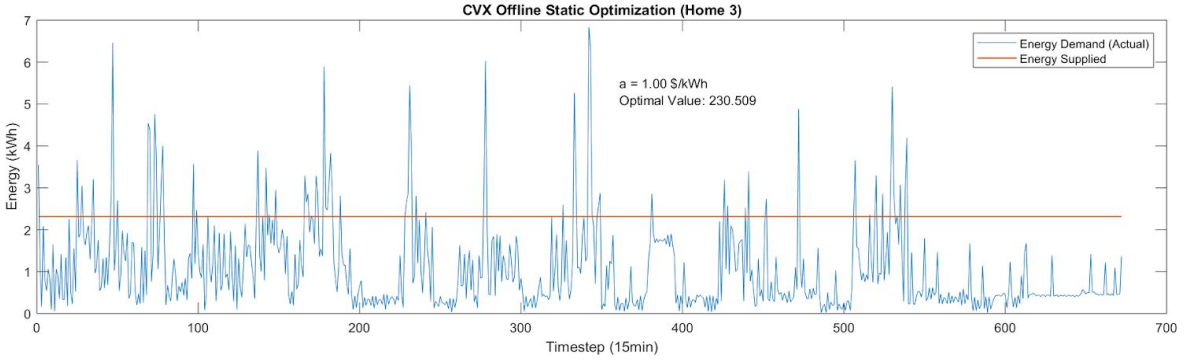
In case of Static Offline optimization, we do not consider the switching cost, so our objective function reduces to below:

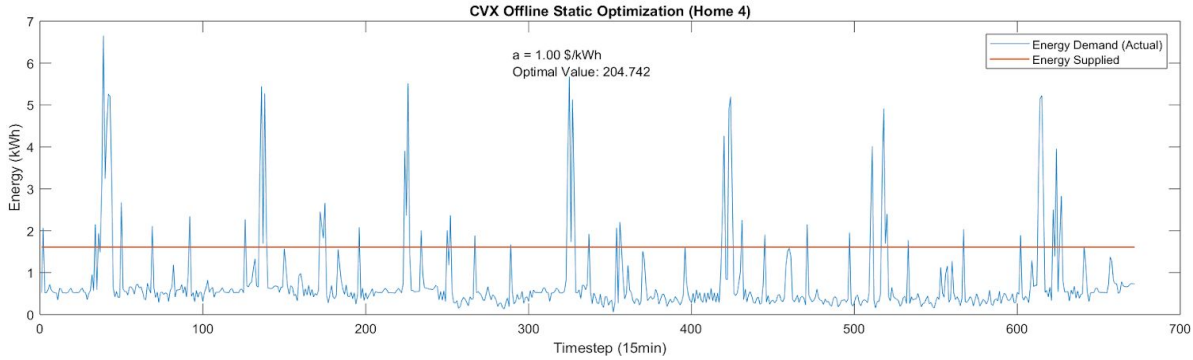
$$\sum_{t=1}^{T} p(t)x(t) + a * \max\{0, y(t) - x(t)\}$$

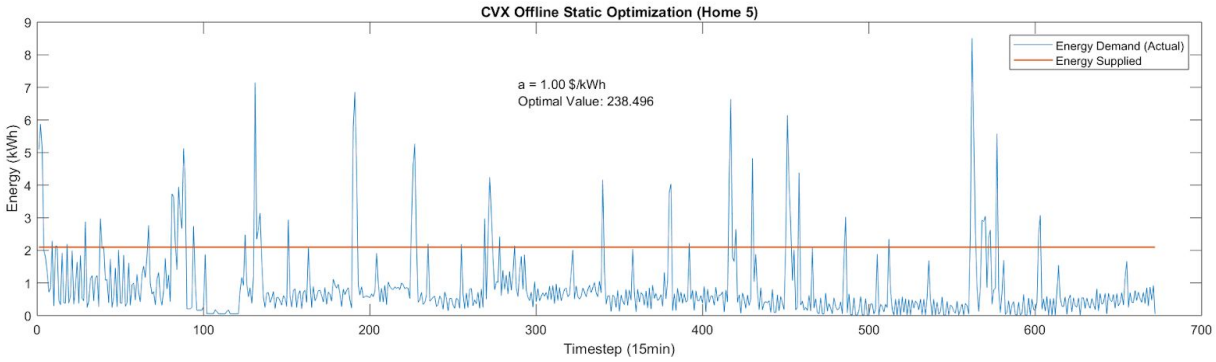
Using the above expression, we have calculated the optimal value for energy supply and plotted the graph for each of the 10 houses.

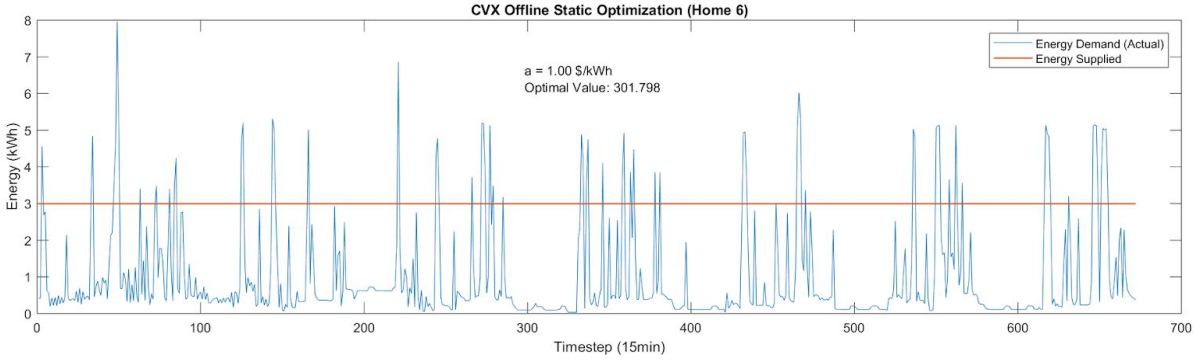

CVX Offline Static Optimization (Home 1)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 228.180
Energy (kWh)
Timestep (15min)

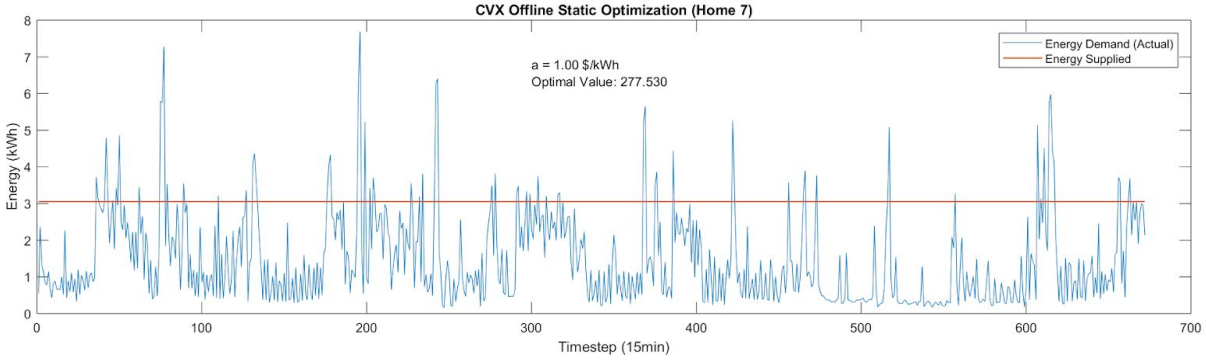

CVX Offline Static Optimization (Home 2)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 266.175
Energy (kWh)
Timestep (15min)

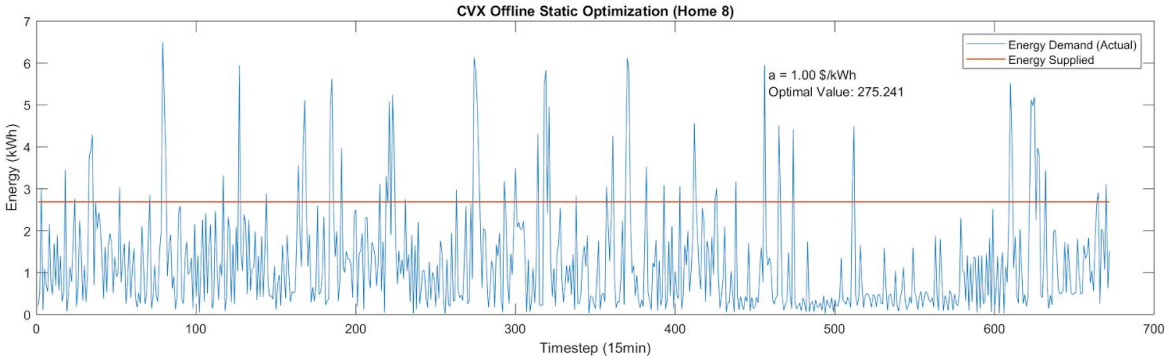

CVX Offline Static Optimization (Home 3)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 230.509
Energy (kWh)
Timestep (15min)

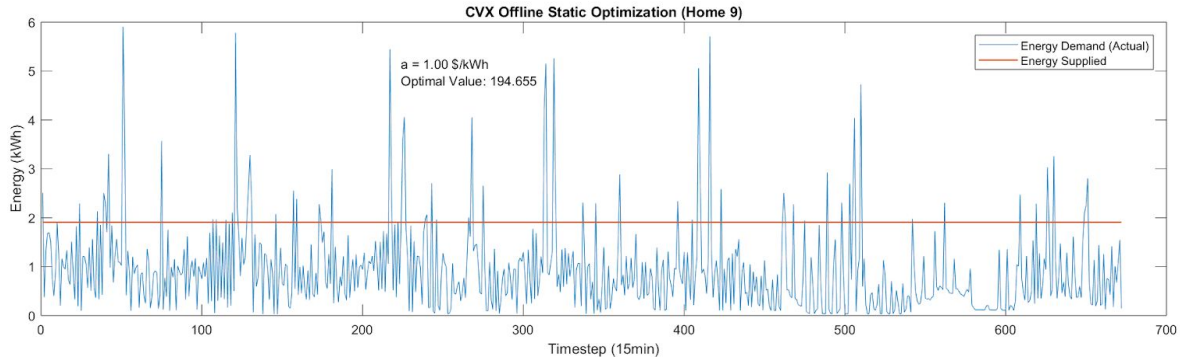

CVX Offline Static Optimization (Home 4)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 204.742
Energy (kWh)
Timestep (15min)

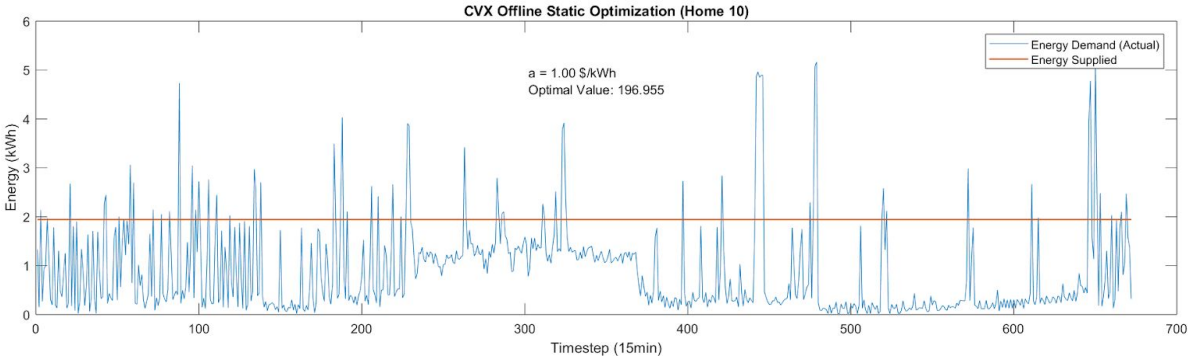

CVX Offline Static Optimization (Home 5)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 238.496
Energy (kWh)
Timestep (15min)


CVX Offline Static Optimization (Home 6)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 301.798
Energy (kWh)
Timestep (15min)


CVX Offline Static Optimization (Home 7)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 277.530
Energy (kWh)
Timestep (15min)


CVX Offline Static Optimization (Home 8)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
Optimal Value: 275.241
Energy (kWh)
Timestep (15min)



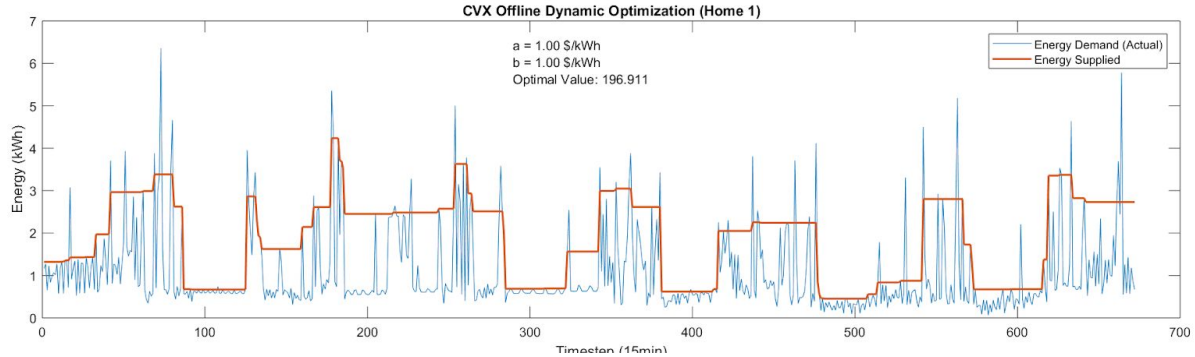


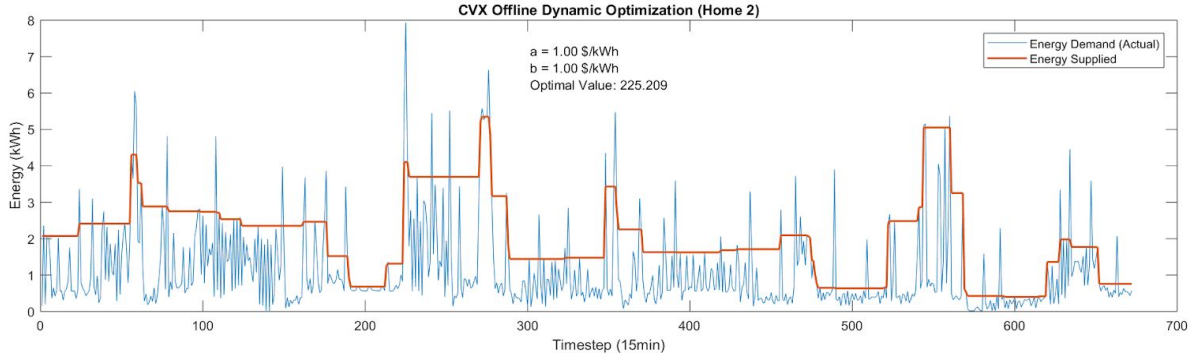
Dynamic Offline Optimization

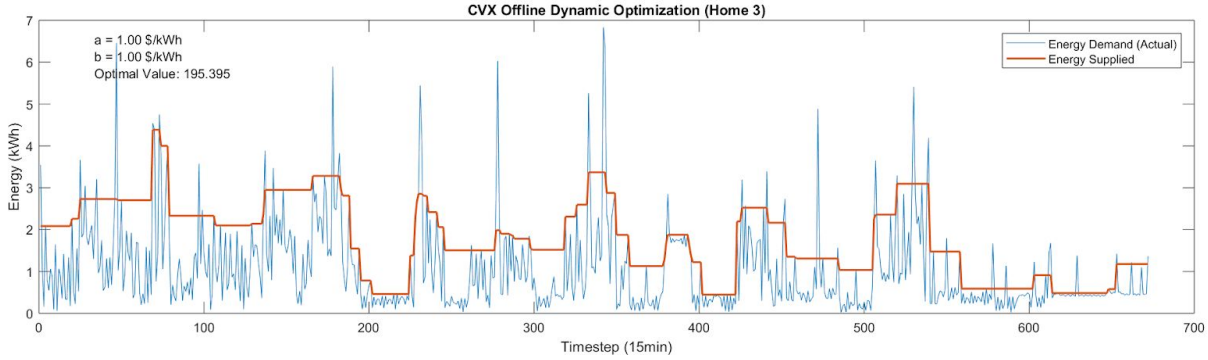
In case of Dynamic Offline Optimization, we consider both the penalty(a = 1) and switching cost(b=1) and our performace measure is defined by below equation:

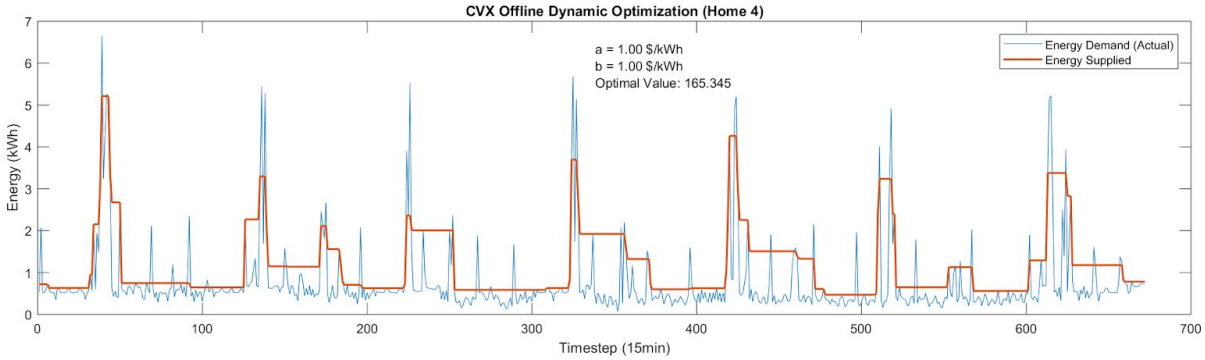
$$\sum_{t=1}^{T} p(t)x(t) + a * \max\{0, y(t) - x(t)\} + b|x(t) - x(t-1)|$$

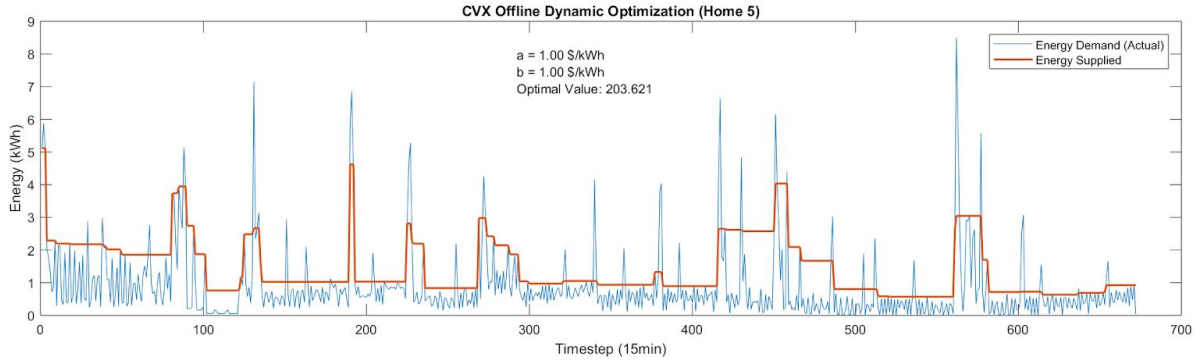
Similar to static offline optimization, we have also calculated the optimal cost and have plotted the graph for each house.

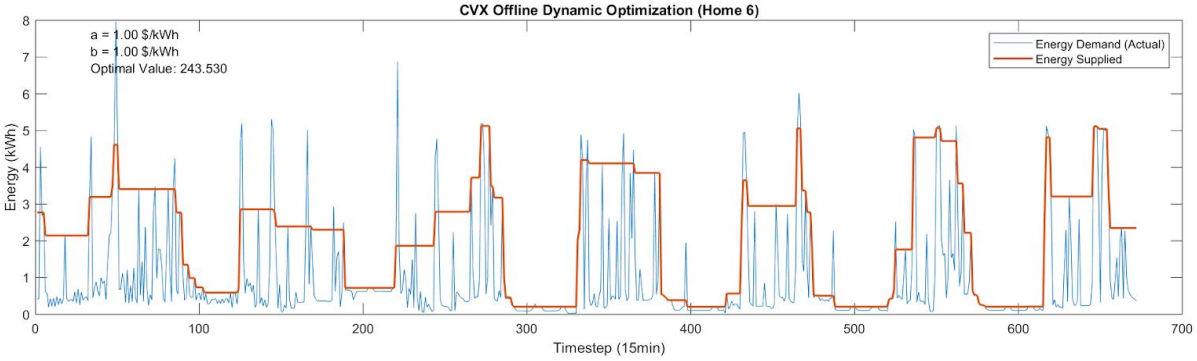

CVX Offline Dynamic Optimization (Home 1)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 196.911
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)

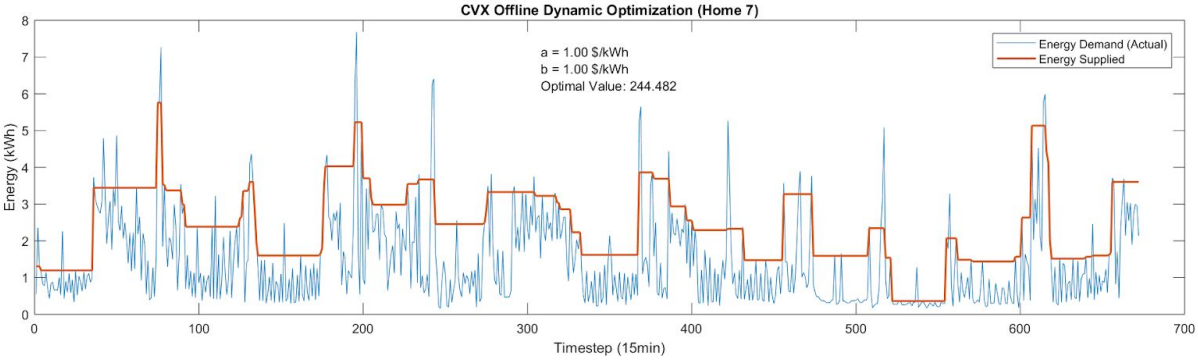

CVX Offline Dynamic Optimization (Home 2)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 225.209
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)

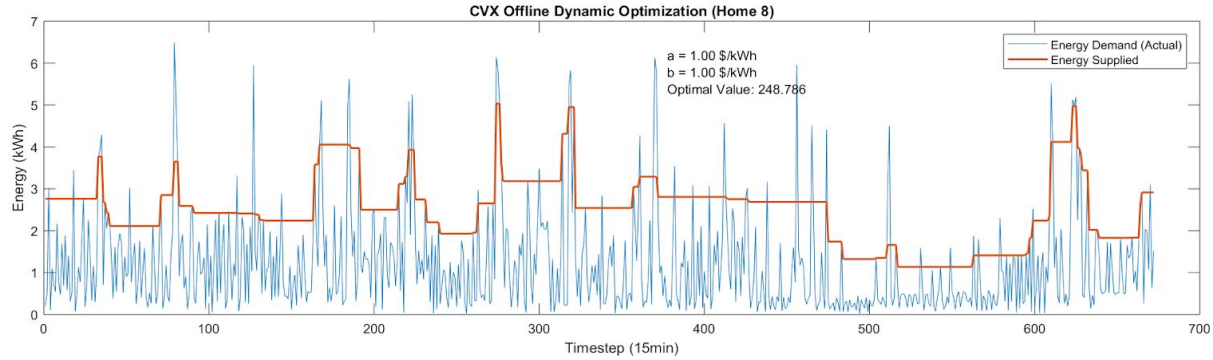

CVX Offline Dynamic Optimization (Home 3)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 195.395
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)

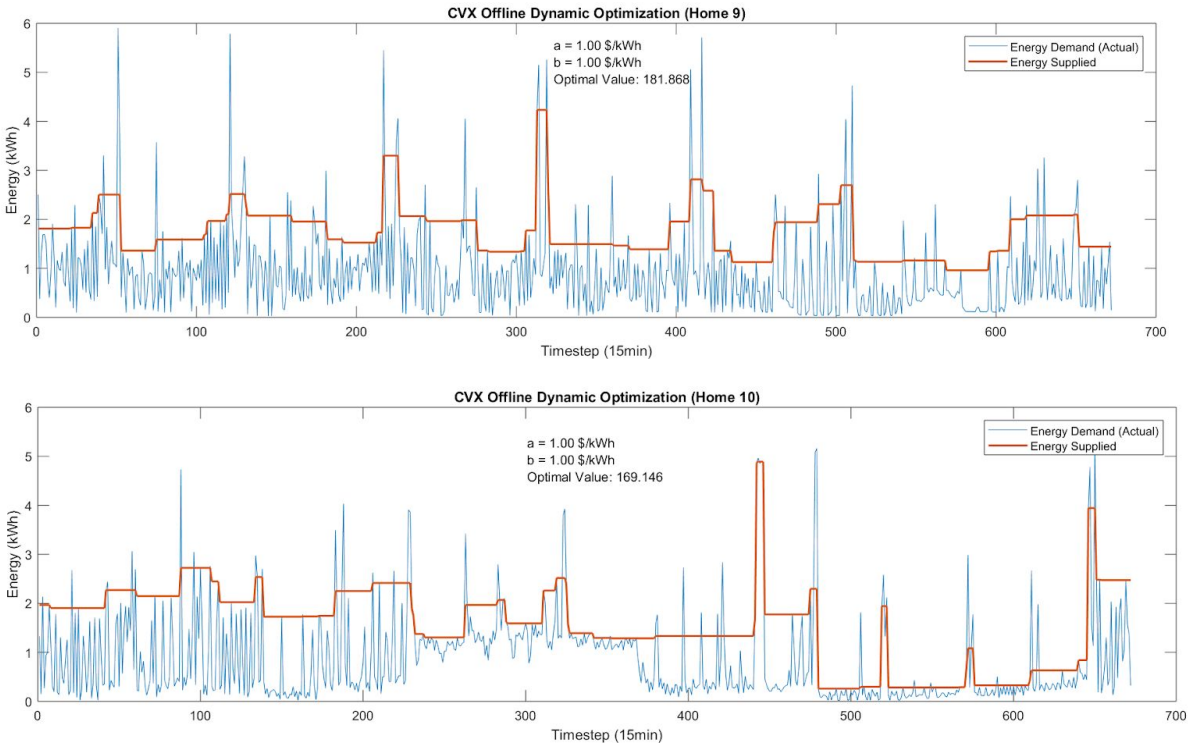

CVX Offline Dynamic Optimization (Home 4)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 165.345
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)


CVX Offline Dynamic Optimization (Home 5)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 203.621
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)


CVX Offline Dynamic Optimization (Home 6)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 243.530
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)


CVX Offline Dynamic Optimization (Home 7)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 244.482
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)


CVX Offline Dynamic Optimization (Home 8)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 248.786
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)



2. DIFFERENT CONTROL ALGORITHMS

Online Gradient Descent

In case of online gradient descent, we find the value of x(t) iteratively using the previous values(x(t-1). The equation is as below:

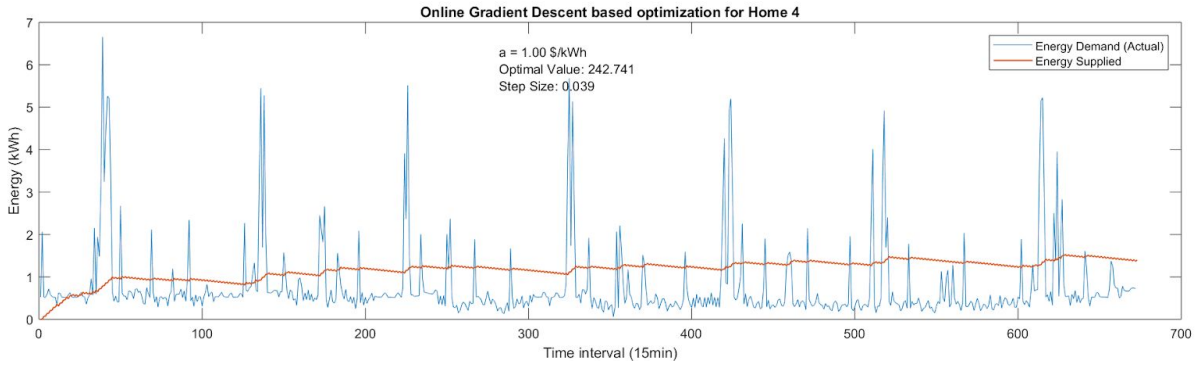
$$x(t+1) = x(t) - learningRate * (\partial f(x(t))/x(t)))$$

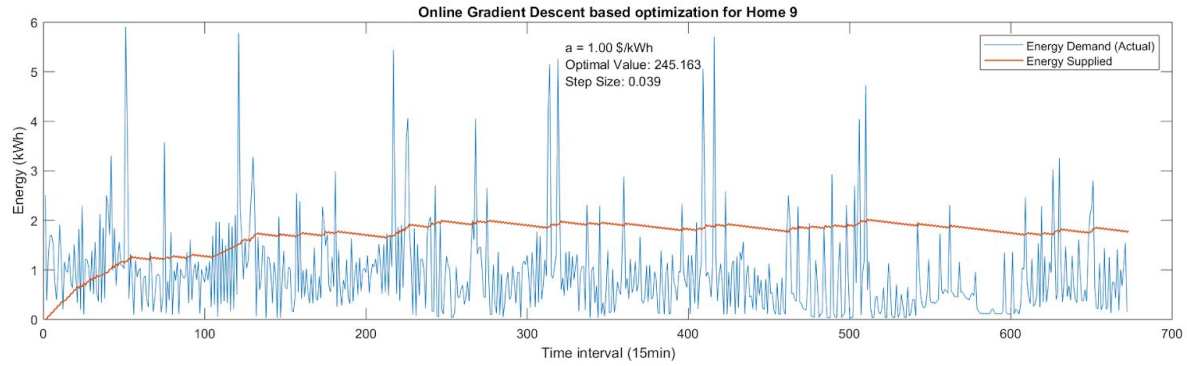
We tried multiple learning rate and also analysed the sensitivity of learning rate by plotting a graph(Later in the text) of how objective function value is changing for a range of learning rate. The table below shows how online gradient descent with different learning rate is performing in comparison to offline optimization solutions.

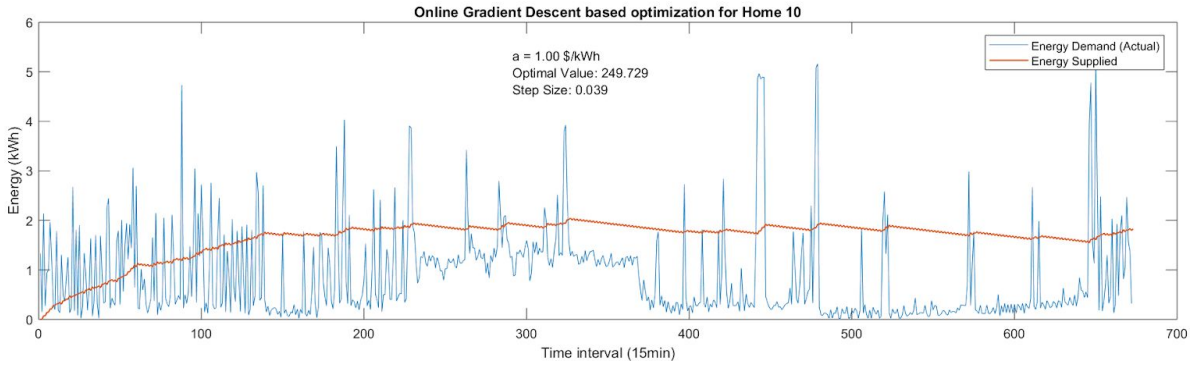
House	Offline Static	Offline Dynamic	OGD | Learning Rate = 0.039	OGD | Learning Rate = 0.01	OGD | Learning Rate = 0.1
4	230.51	195.4	242.74	244.42	279.38
9	194.65	181.87	245.16	277.69	270.89
10	196.96	169.15	249.73	279.7	275.39

Learning Rate: 0.039

In this case, we have considered just 3 houses, viz. 4, 9, 10 for our algorithm analysis. Next 3 graphs denote the performance of online gradient descent with learning rate of 0.039 for houses 4, 9 and 10 respectively

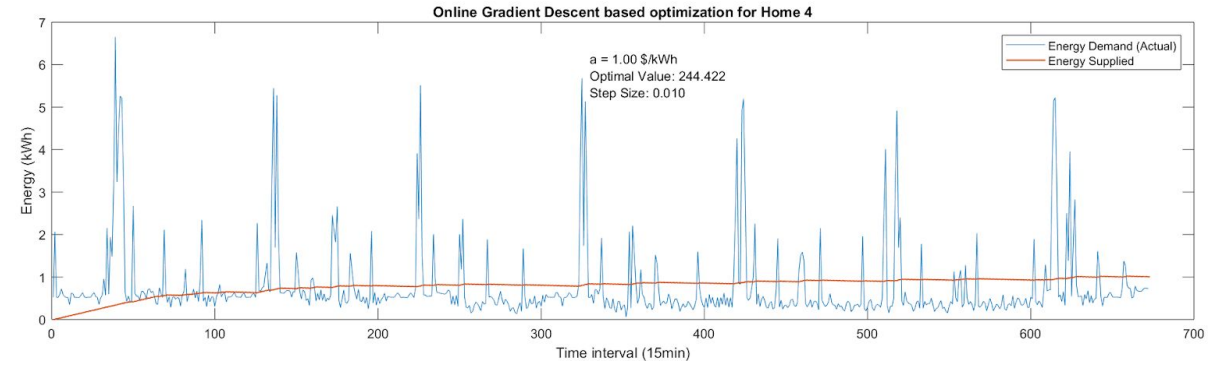


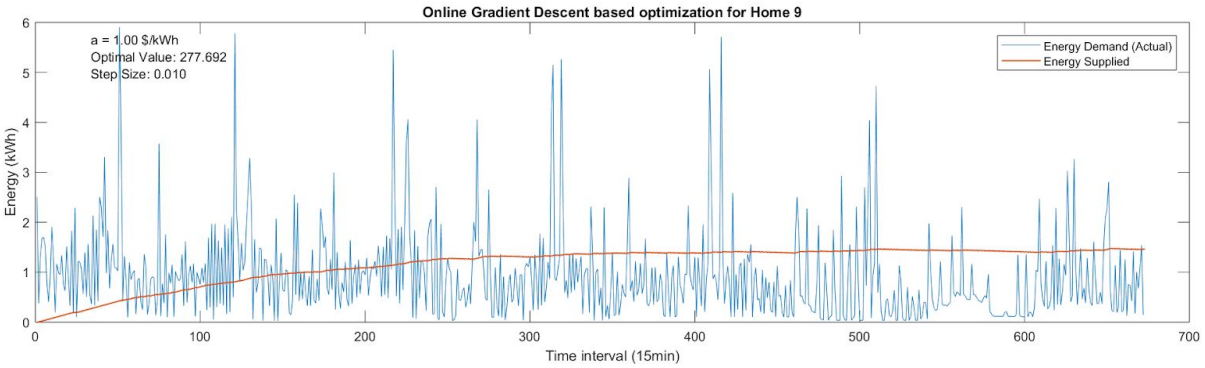


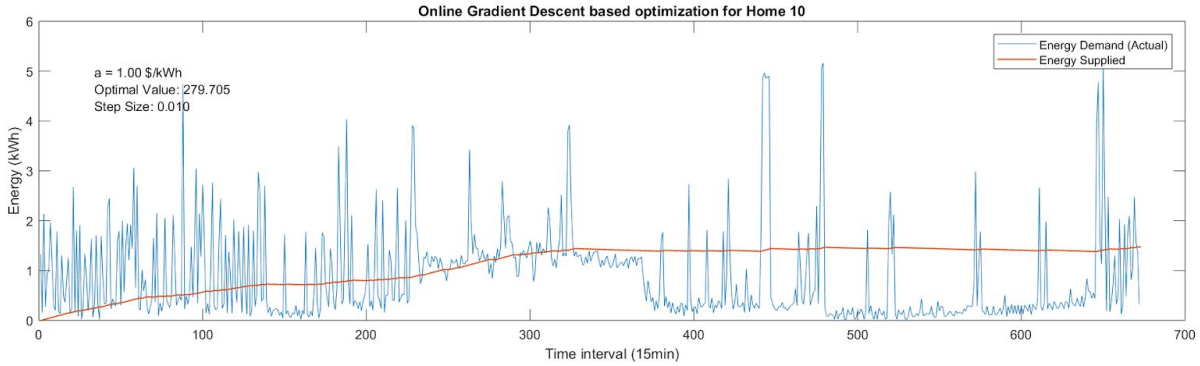


Learning Rate: 0.01

Next 3 graphs denote the performance of online gradient descent with learning rate of 0.01 for houses 4, 9 and 10 respectively.

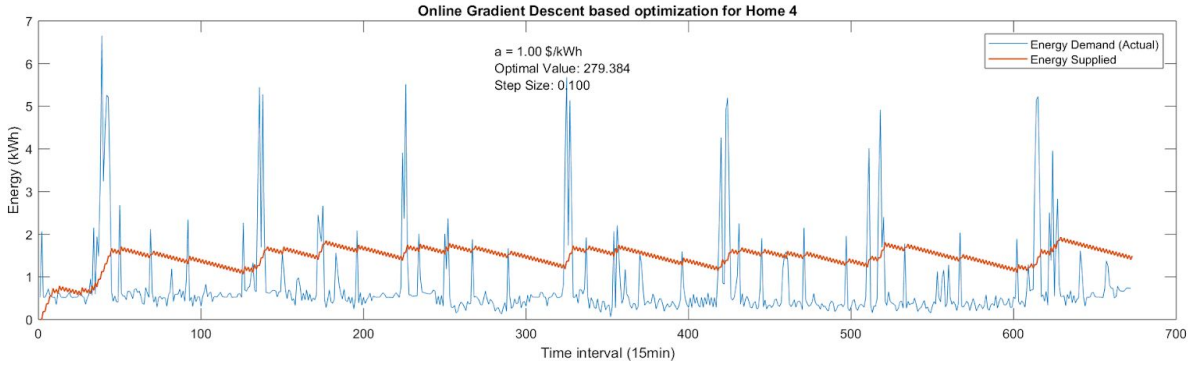


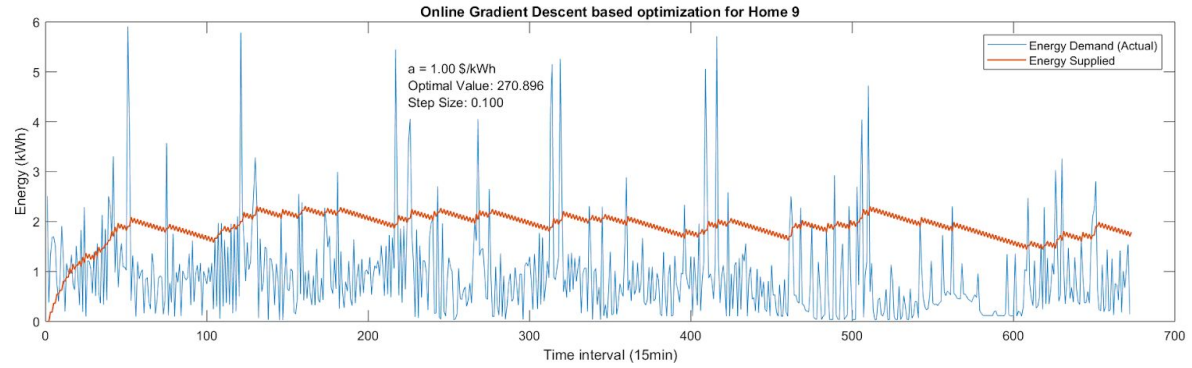


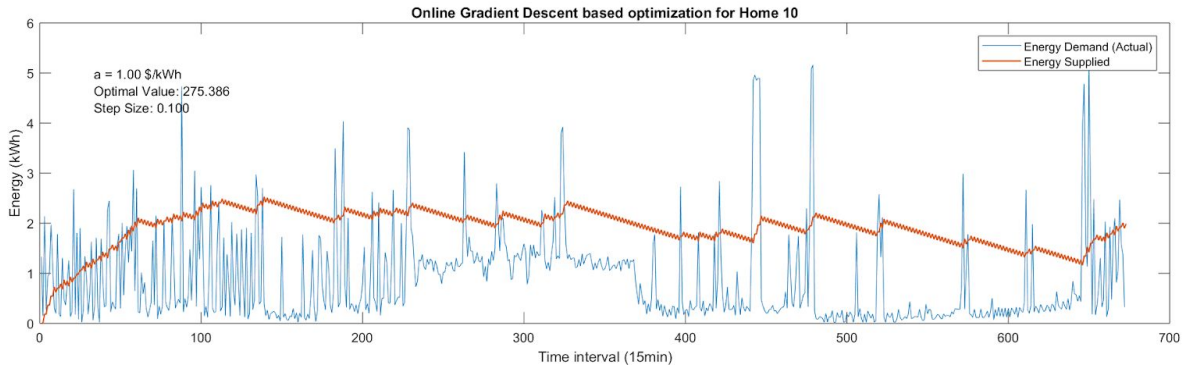


Learning Rate: 0.1

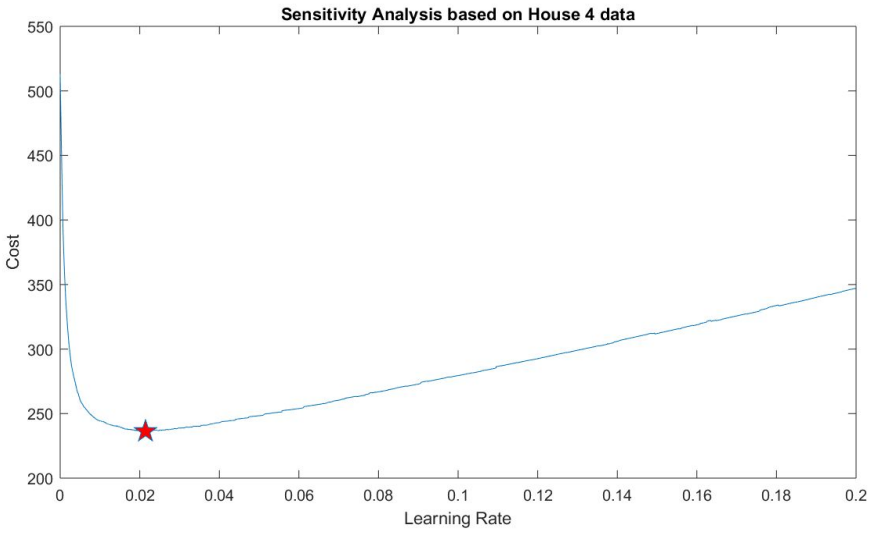
Next 3 graphs denote the performance of online gradient descent with learning rate of 0.1 for houses 4, 9 and 10 respectively.







Sensitivity Analysis for Online Gradient Descent



Receding Horizon Control

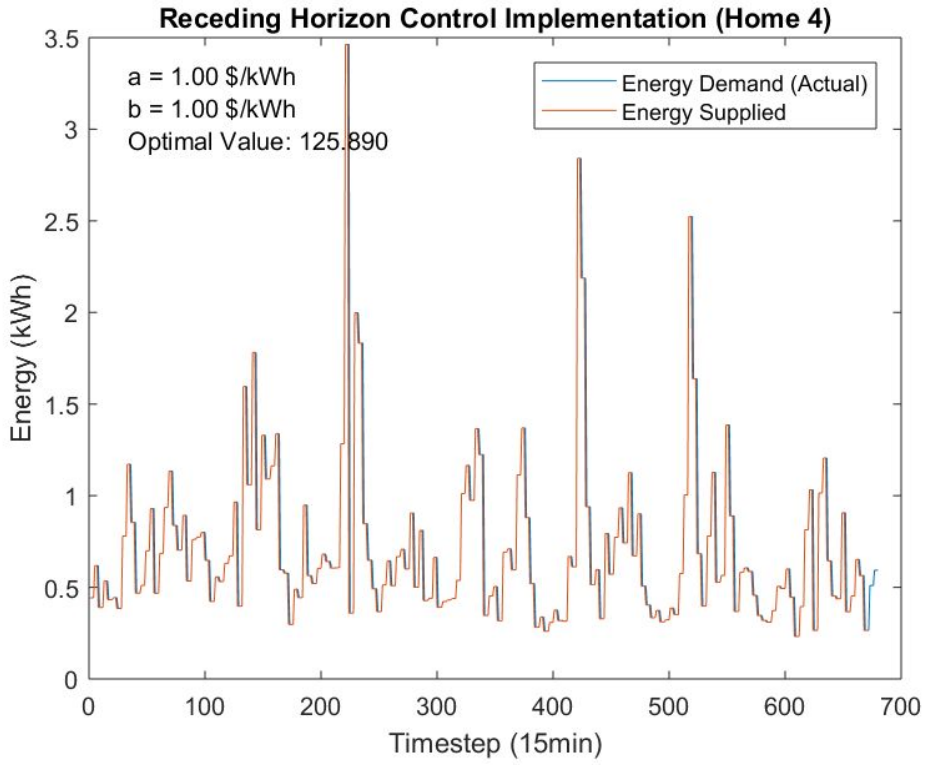
RHC is a control algorithm, popularly known as Model Predictive Control. It is often used in control systems like automated cars to predict steering angle movement, acceleration and speed with time.

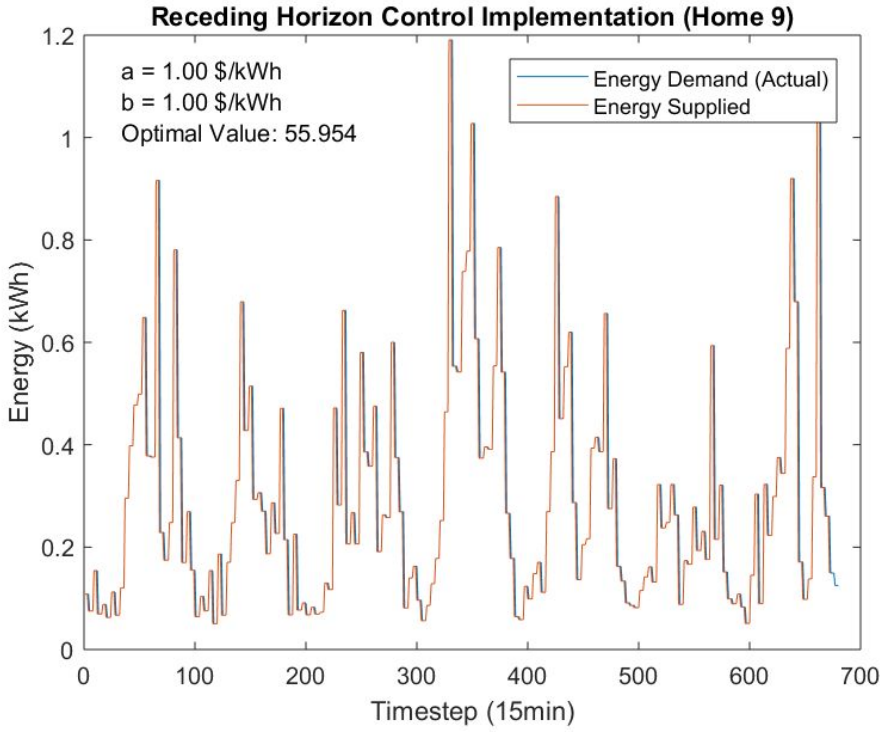
For the electricity provisioning problem, we performed RHC over different window sizes for all 3 houses - House 4, House 9 and House 10. Our goal was to minimize the above-mentioned objective function. As ground truth values of demand are not present for RHC, we used our predictions from Assignment 1. We specifically used 2 algorithms for prediction

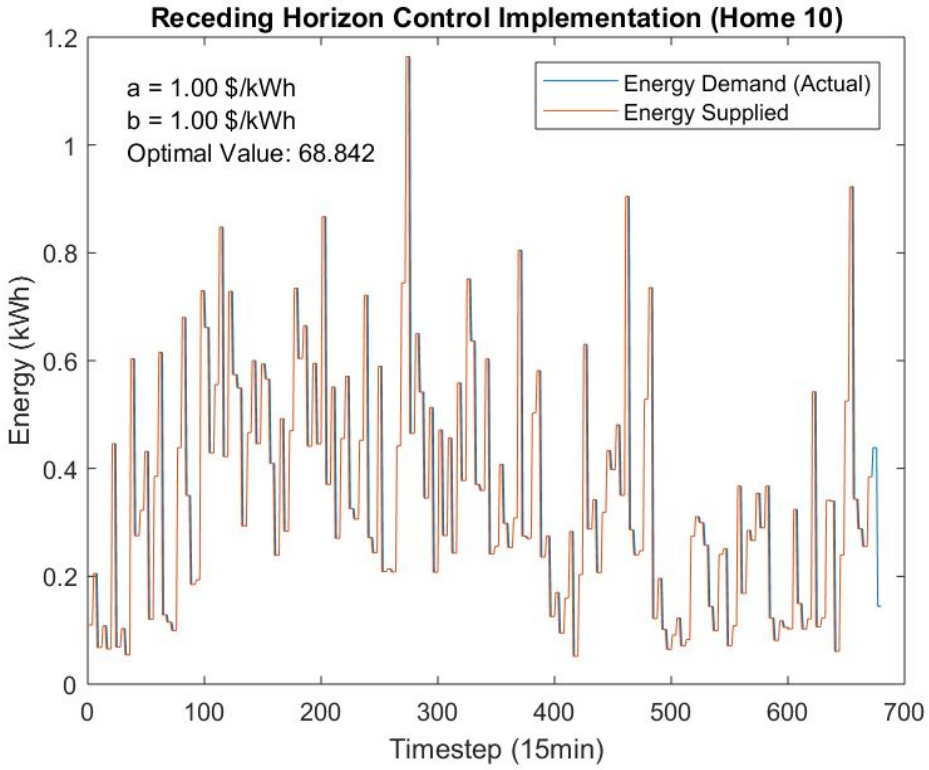
i) Random Forest
ii) Ridge Regression

We ran RHC for a total of 6 times - 2 prediction values for 3 houses. We found that values predicted from Random Forest regressor gave lower objective function values than Ridge Regressor (as seen in the plots and table below). Further, House 9 had the best objective values among all 3 houses.

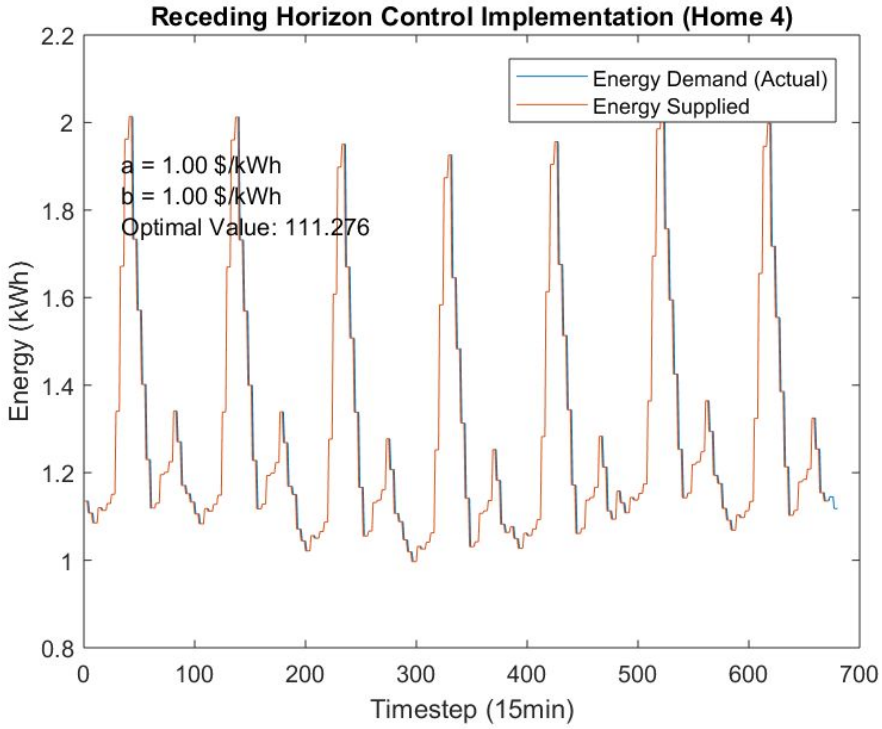
Using Random Forest for demand prediction

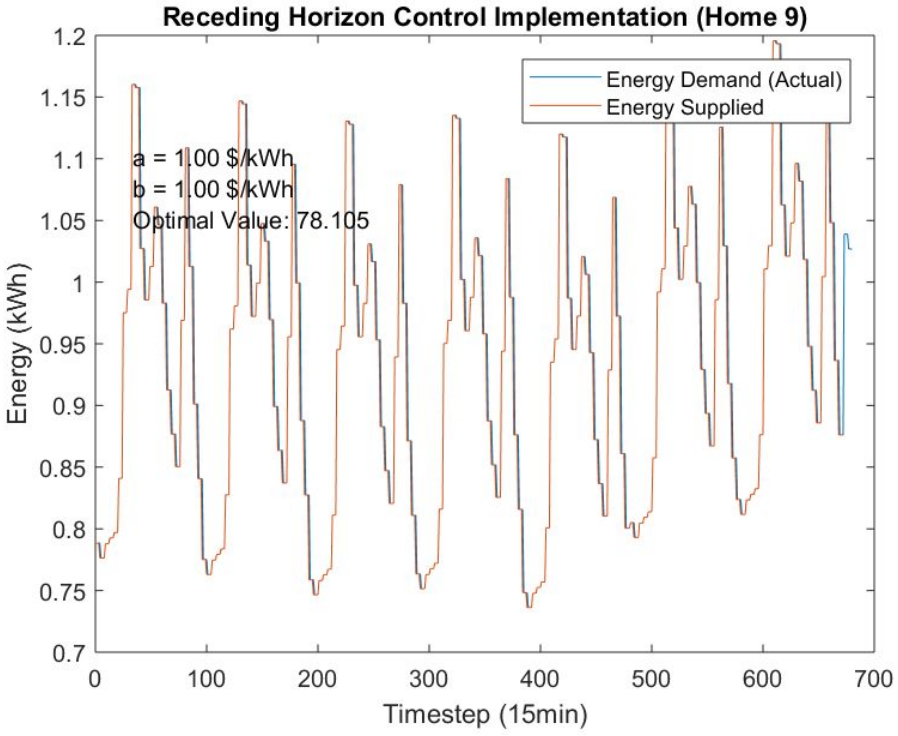

Receding Horizon Control Implementation (Home 4)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 125.890
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)
0
0.5
1
1.5
2
2.5
3
3.5
0
100
200
300
400
500
600
700

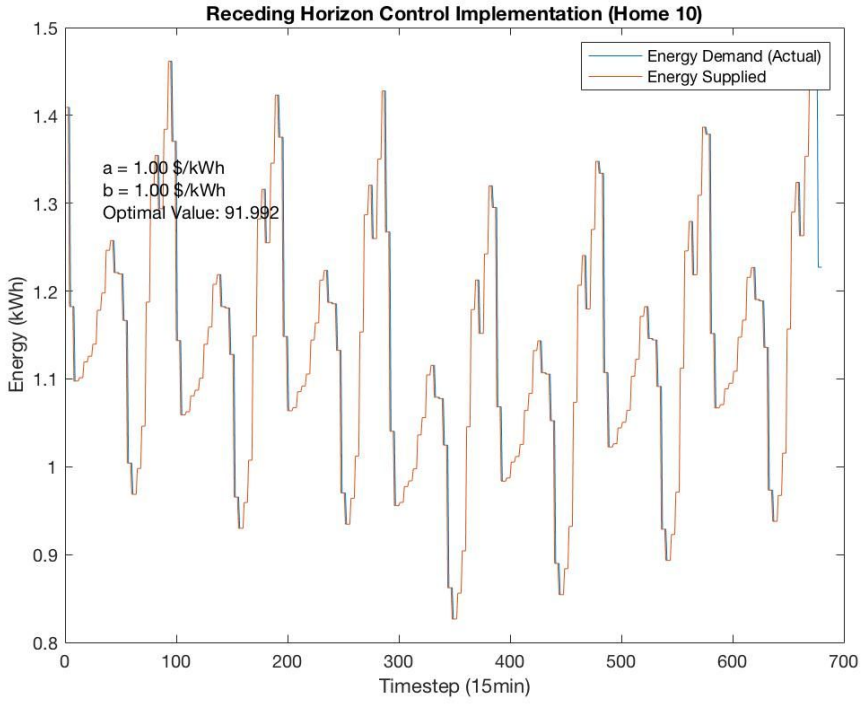

Receding Horizon Control Implementation (Home 9)
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 55.954
Energy Demand (Actual)
Energy Supplied
Energy (kWh)
Timestep (15min)
0
0.2
0.4
0.6
0.8
1
1.2
0
100
200
300
400
500
600
700



Using Ridge Regression for demand prediction




Receding Horizon Control Implementation (Home 9)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 78.105
Energy (kWh)
Timestep (15min)
1.2
1.15
1.1
1.05
1
0.95
0.9
0.85
0.8
0.75
0.7
0
100
200
300
400
500
600
700


Receding Horizon Control Implementation (Home 10)
Energy Demand (Actual)
Energy Supplied
a = 1.00 $/kWh
b = 1.00 $/kWh
Optimal Value: 91.992
Energy (kWh)
Timestep (15min)
1.5
1.4
1.3
1.2
1.1
1
0.9
0.8
0
100
200
300
400
500
600
700

Commitment Horizon Control

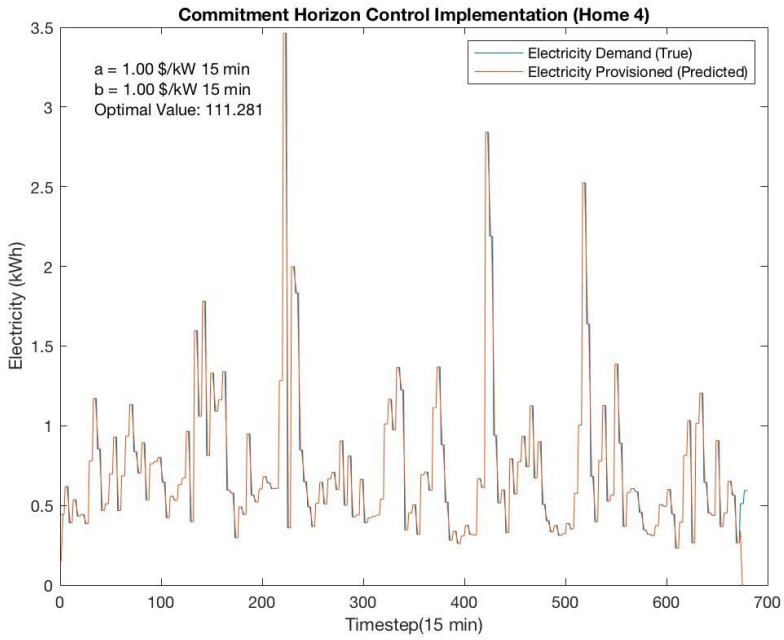
We ran the CHC algorithm 6 times - 2 Prediction Values from Random Forest and Ridge Regression algorithms were used for 3 different house - 4, 9 and 10.

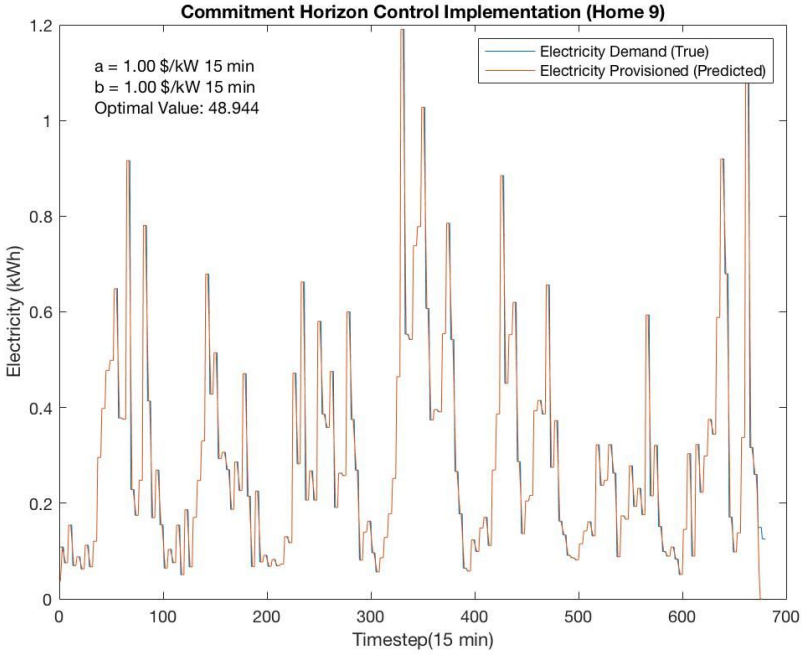
Prediction Horizon Size - 5
Commitment Horizon Size - 3

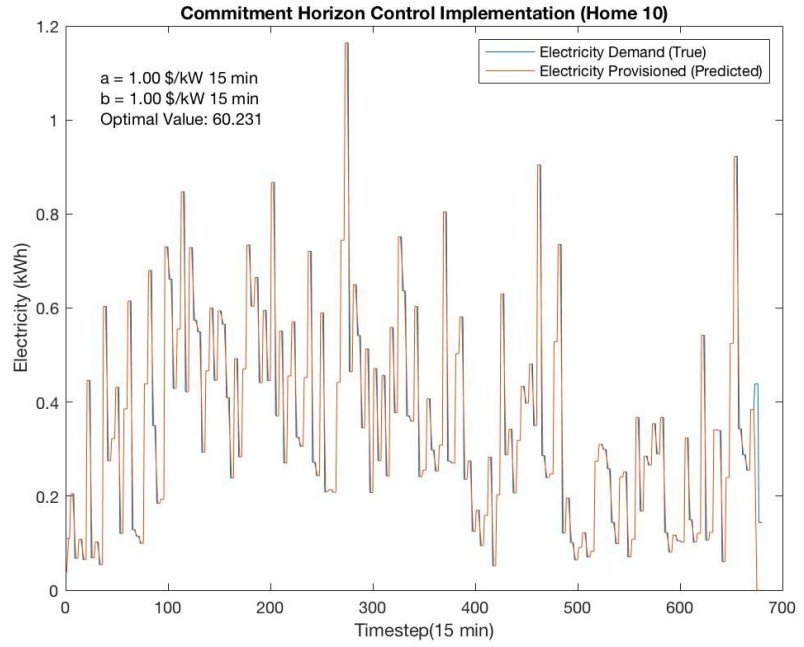
Predictions from Random Forest regressor performed better. Further, CHC performed better than RHC and OGD in most of the cases.

Further, we also varied the prediction horizon size (5, 3) and found that there was not much difference in varying the window sizes for such low values. Though we could have increased the window size to high values like 100, we didn't do that because higher window size predictions do not account for uncertain events of the future.

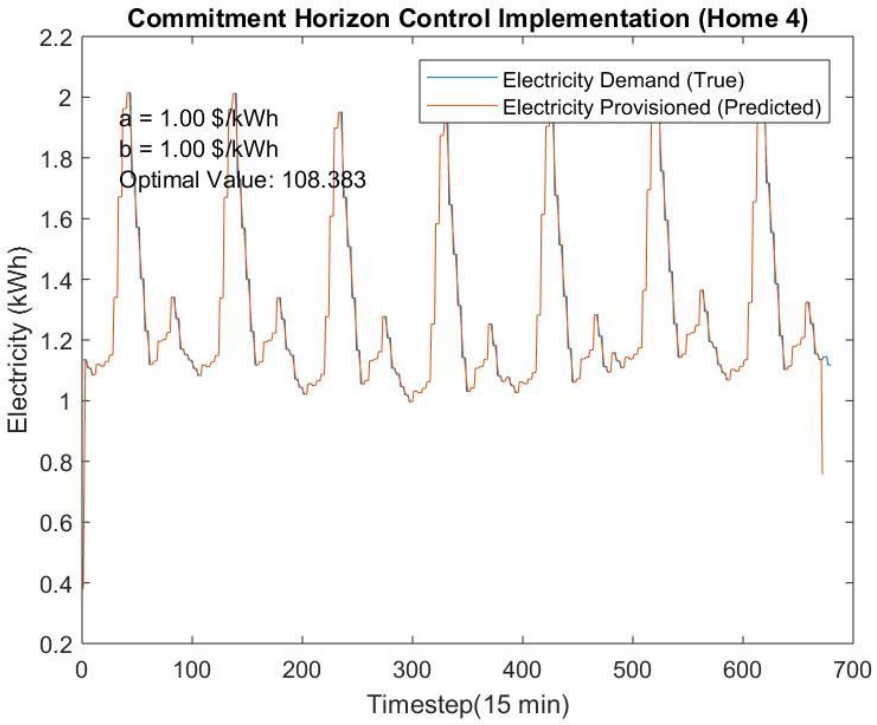
Using Random Forest for demand prediction

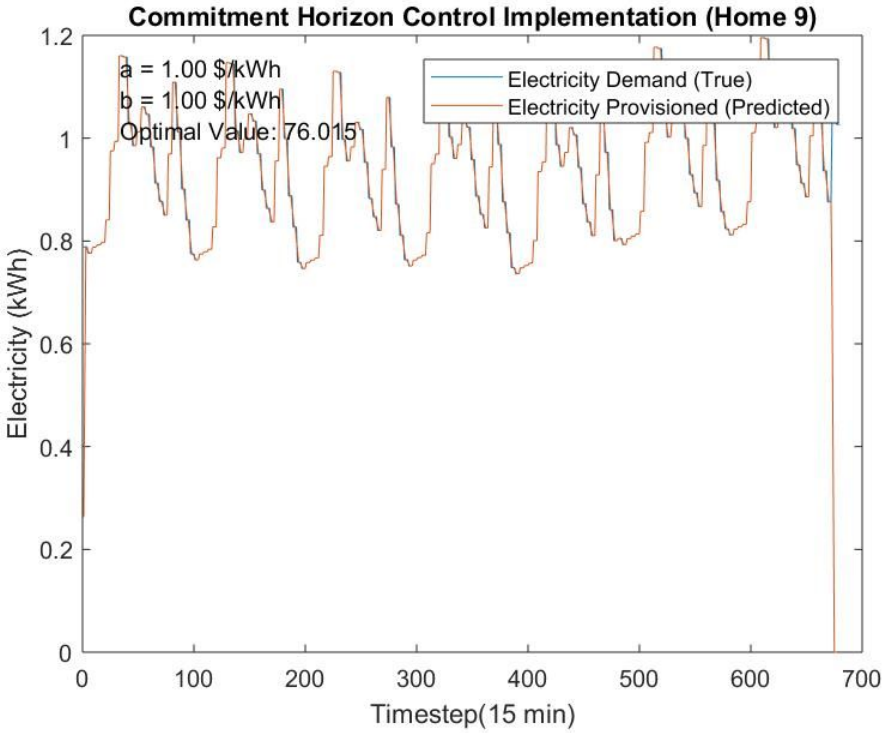


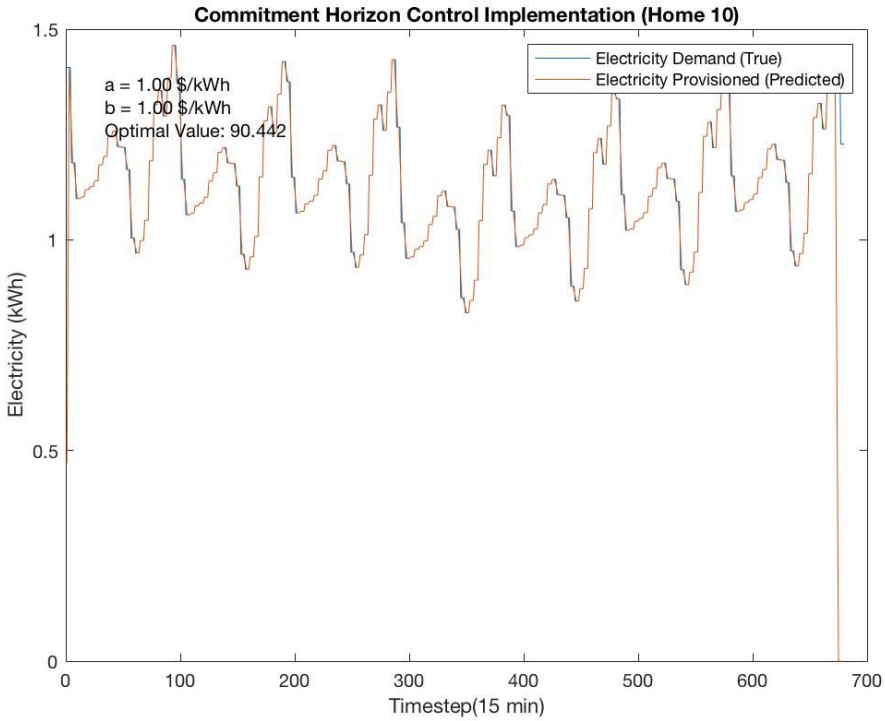

Commitment Horizon Control Implementation (Home 9)
a = 1.00 $/kW 15 min
b = 1.00 $/kW 15 min
Optimal Value: 48.944
Electricity Demand (True)
Electricity Provisioned (Predicted)
Electricity (kWh)
Timestep(15 min)
0
0.2
0.4
0.6
0.8
1
1.2
0
100
200
300
400
500
600
700


Commitment Horizon Control Implementation (Home 10)
a = 1.00 $/kW 15 min
b = 1.00 $/kW 15 min
Optimal Value: 60.231
Electricity Demand (True)
Electricity Provisioned (Predicted)
Electricity (kWh)
Timestep(15 min)
0
0.2
0.4
0.6
0.8
1
1.2
0
100
200
300
400
500
600
700

Using Ridge Regression For Demand Prediction







3. Comparison of algorithms

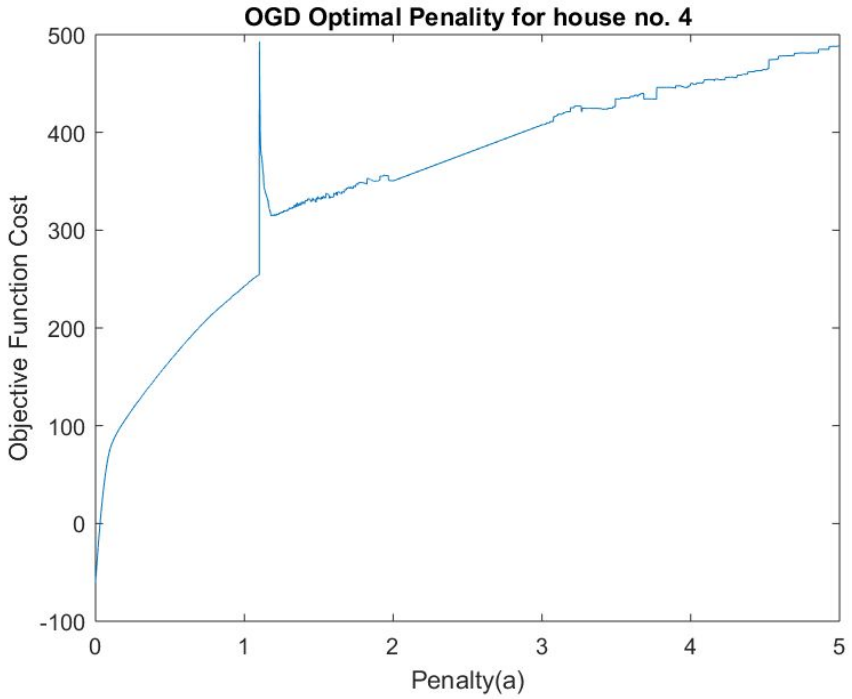
After trying different algorithms, we tried putting their outputs side by side for comparison. Not just the algorithms, but we also took account of their performances based on certain constants in their equations. Below table sums up the comparison. We have done the comparison for houses 4, 9 and 10.

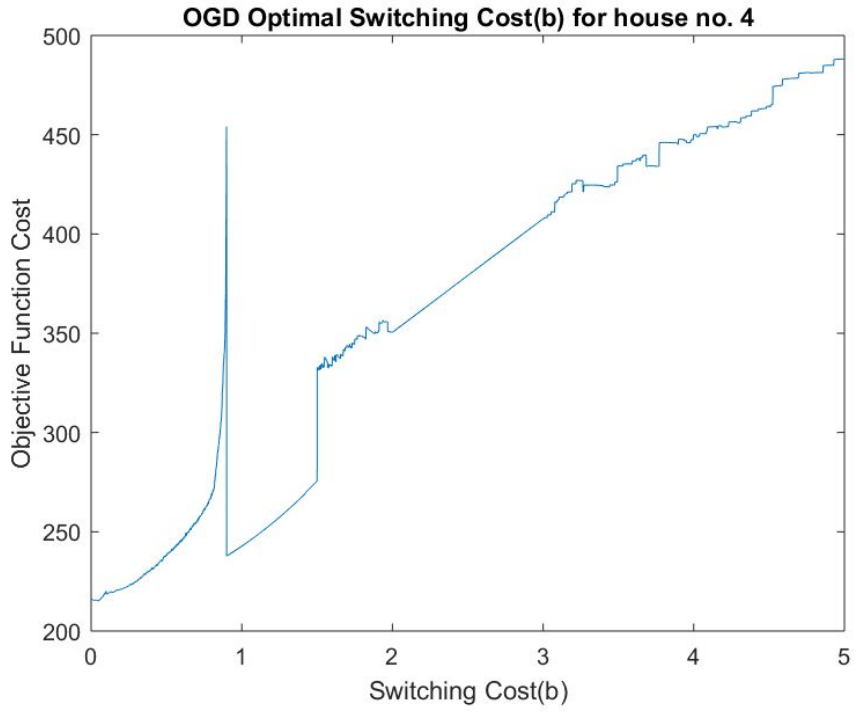
Method/House	4	9	10
Offline Static	230.51	194.65	196.96
Offline Dynamic	195.4	181.87	169.15
OGD | Learning Rate = 0.039	242.74	245.16	249.73
OGD | Learning Rate = 0.01	244.42	277.69	279.7
OGD | Learning Rate = 0.1	279.38	270.89	275.39
RHC | Prediction Horizon = 5.0 | Ridge Regression	111.26	78.05	100
RHC | Prediction Horizon = 3.0 | Ridge Regression	200.62	190.77	175.34
RHC | Prediction Horizon = 10.0 | Ridge Regression	190.67	178.24	160.11
RHC | Prediction Horizon = 5.0 | Random Forest	125.89	55.94	68.84
RHC | Prediction Horizon = 3.0 | Random Forest	140.62	70.67	80.23
RHC | Prediction Horizon = 10.0 | Random Forest	120.76	58.98	70.54
CHC | Prediction Horizon = 5.0 | Commitment Horizon = 3.0 | Ridge Regression	108.38	76.01	90.44
CHC | Prediction Horizon = 3.0 | Commitment Horizon = 3.0 | Ridge Regression	115.35	79.67	92.21
CHC | Prediction Horizon = 5.0 | Commitment Horizon = 3.0 | Random Forest	111.28	48.94	60.23
CHC | Prediction Horizon = 3.0 | Commitment Horizon = 3.0 | Random Forest	113.82	51.49	63.56

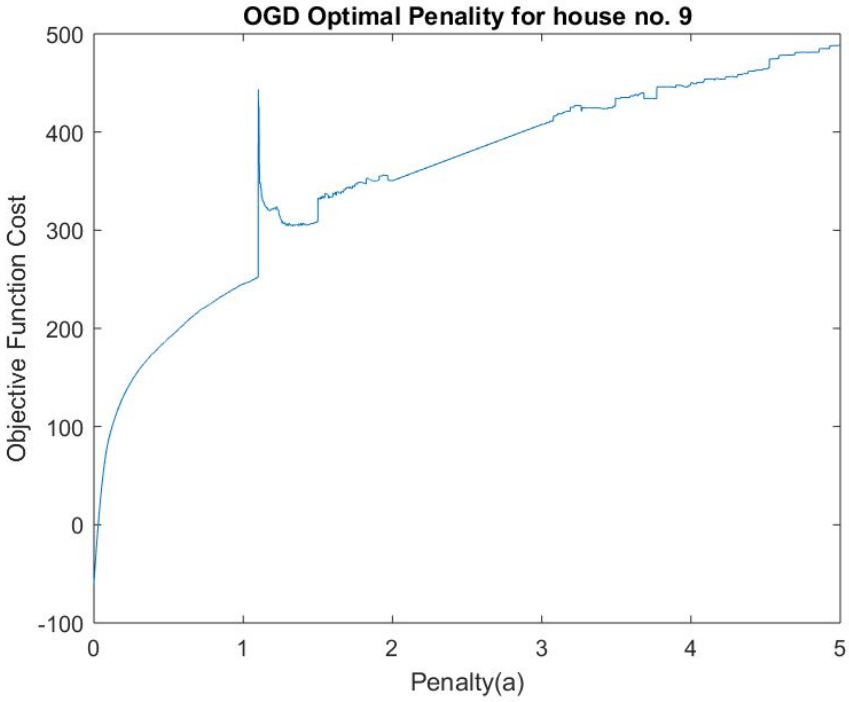
4. Trying multiple values of a and b

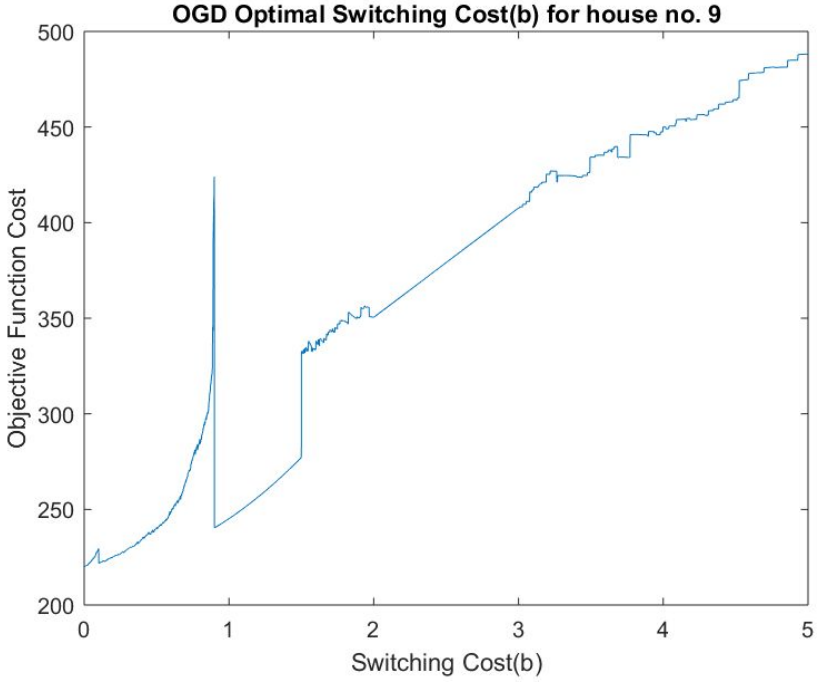
We tried changing the values of penalty(a) and switching costs(b) in our objective function to see how our algorithms perform. We did this analysis and plotted graphs for each of the combination as shown below.

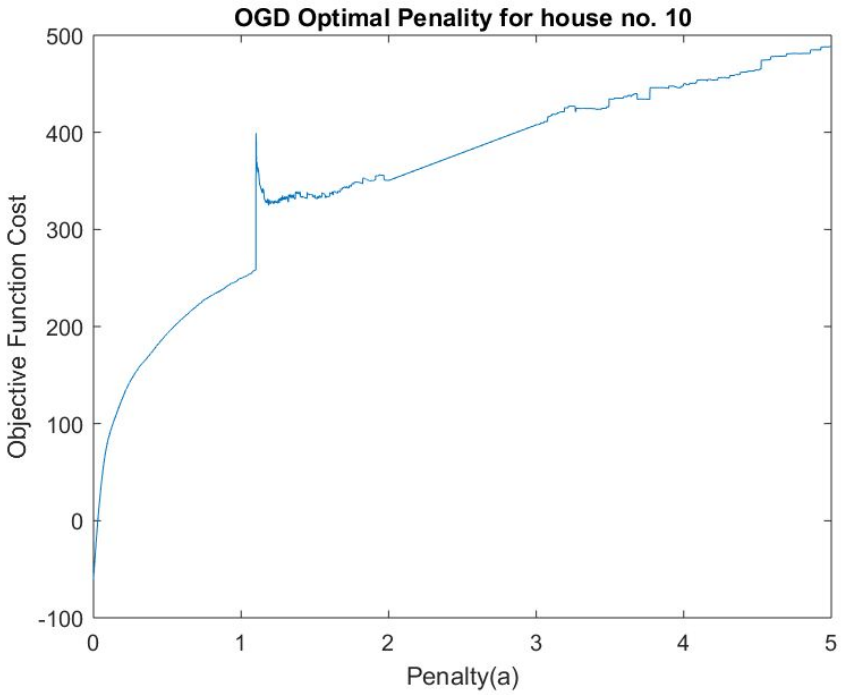
Online Gradient Descent

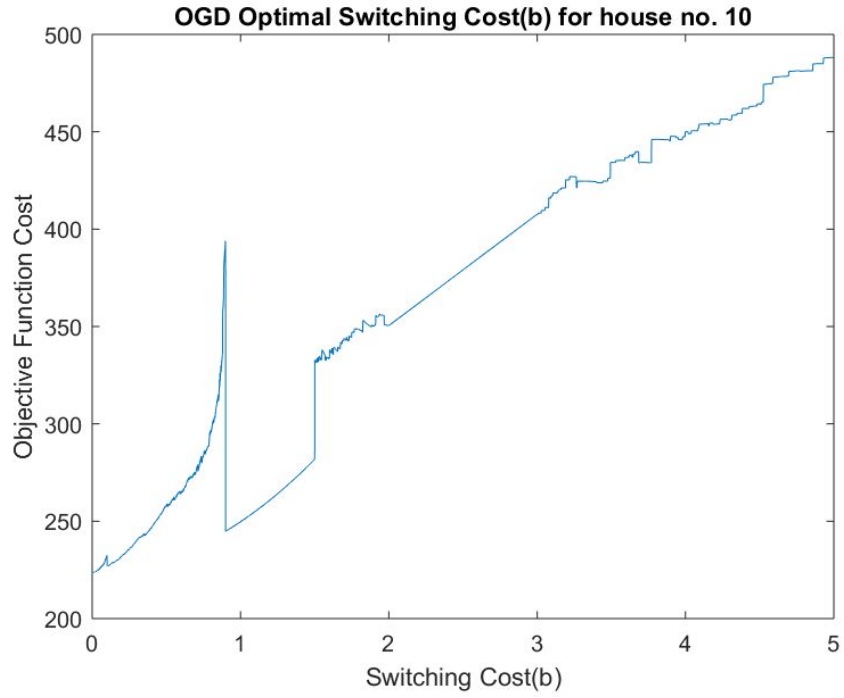

OGD Optimal Penality for house no. 4
Objective Function Cost
Penalty(a)
500
400
300
200
100
0
-100
0
1
2
3
4
5


OGD Optimal Switching Cost(b) for house no. 4
Objective Function Cost
Switching Cost(b)
500
450
400
350
300
250
200
0
1
2
3
4
5

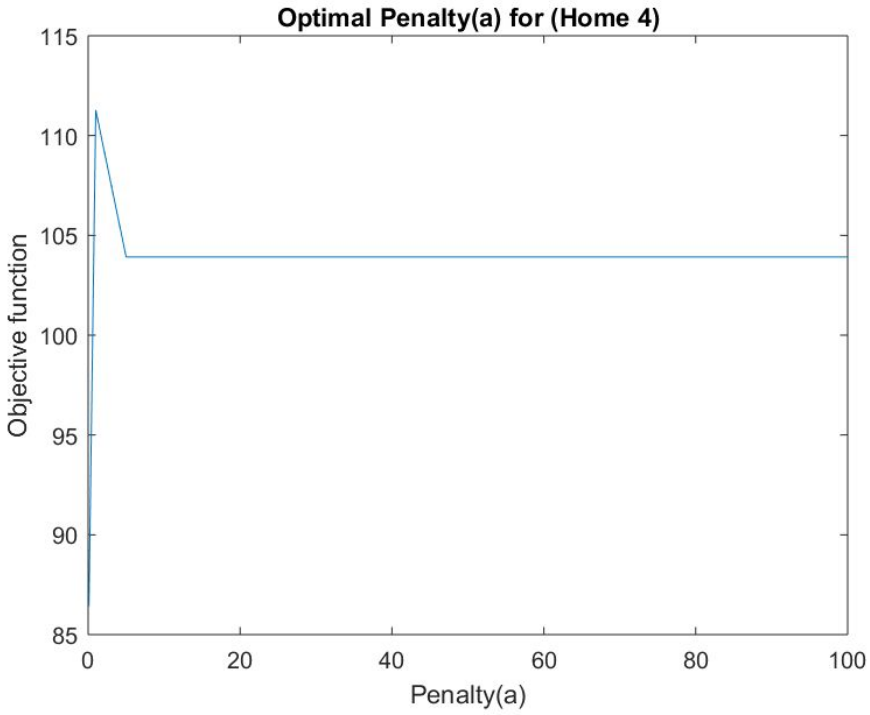

OGD Optimal Penality for house no. 9
Objective Function Cost
Penalty(a)
500
400
300
200
100
0
-100
0
1
2
3
4
5

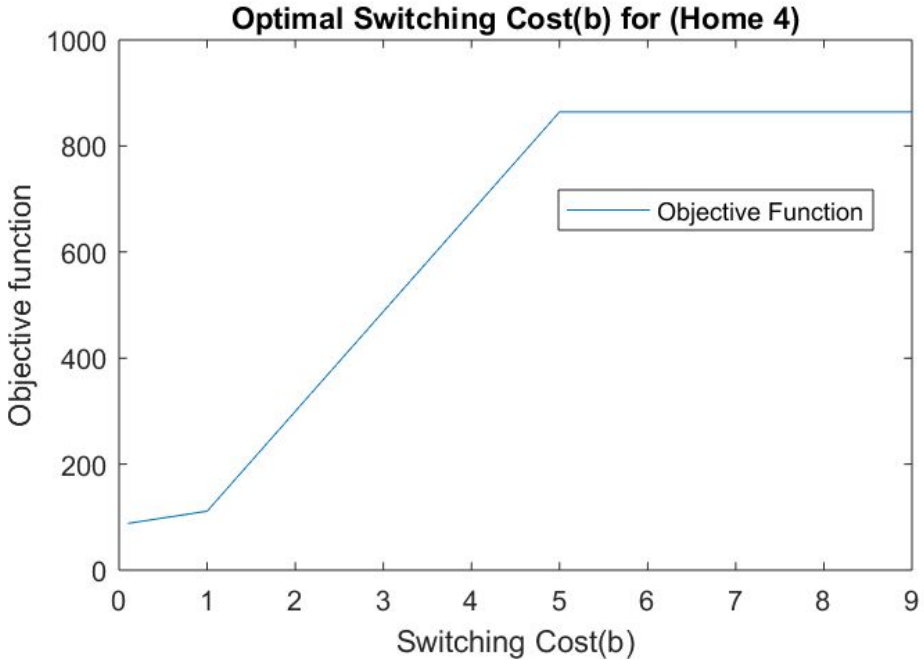

OGD Optimal Switching Cost(b) for house no. 9
Objective Function Cost
Switching Cost(b)
500
450
400
350
300
250
200
0
1
2
3
4
5





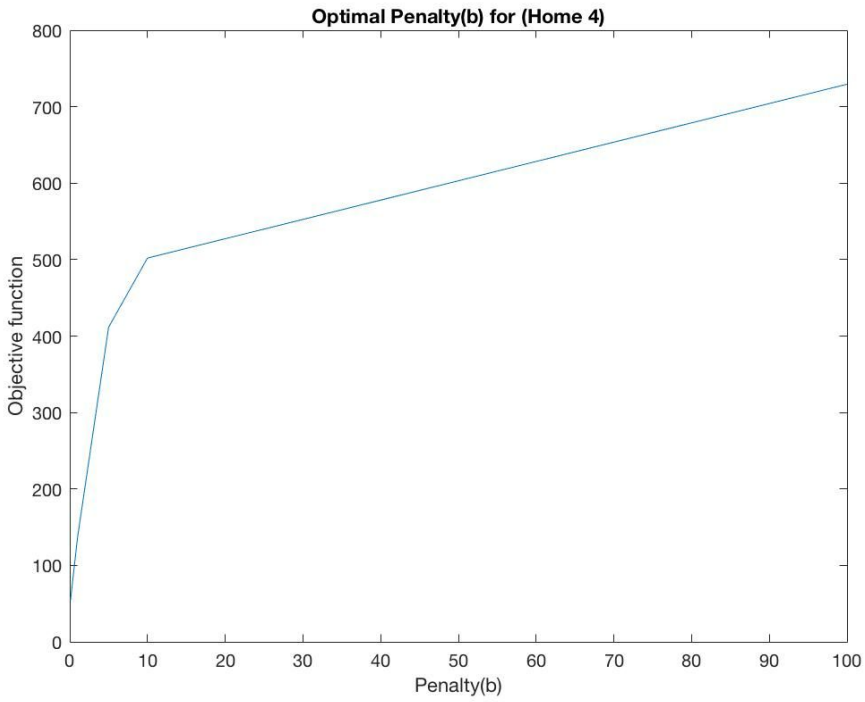
Receding Horizon Control





Commitment Horizon Control

The plot on varying the switching coefficient(b) in CHC algorithm.



5. ALGORITHM SELECTION

Deterministic Algorithm Selection

In order to select the best algorithm out of multiple algorithms for online convex optimization, we need to fix the performance criteria across the algorithms. We could have used
i) Competitive Ratio ii) Competitive Difference iii) Regret factor

We have decided to use the regret factor as our evaluation criteria. For each of the 3 algorithms - OGD, RHC, CHC, we minimize the objective equation

sum_t p(t)*x(t) + a*max{0,y(t)-x(t)} + b*(x(t)-x(t-1)

Algorithm

i) Choose a window size. Ex. w = 4different time slots t1 & t2 at step size 4
ii) Iterate over the time horizon (T=672) with step size = window size
iii) In each window, evaluate the objective function value for each algorithm
iv) The algorithm which gives the value closest to the offline algorithm wins that round/window size
v) We then measured the number of times OGD, RHC and CHC won respectively

vi) The algorithm which won the maximum number of times is the winner

Implementation

We performed the above algorithm on all 3 houses (House 4, 9, 10).
We used a window size of 4 and evaluated each value.

Results
We observed that CHC won most of the times, providing the lowest optimal value as compared to the offline algorithm. Our observed order was CHC > OGD > RHC

Randomized Algorithm Selection

Algorithm

i) Assign random weights to each algorithm. Ex. w1, w2, w3
ii) Pick a window size. Say 4
iii) Run all 3 algorithms on each window
iv) After each iteration

a) Rank the algorithms on the basis of performance
b) Update the weights using any heuristic. We incremented the weight of winner by 0.1 and decreased the weight of loser by 0.1.

v) After all iterations, the final values of weights gives optimal value

Implementation

We performed the above algorithm on all 3 houses (House 4, 9, 10).
We used a window size of 4 and evaluated each value.

Results
Our final weights were 0.2*OGD + 0.5*CHC + 0.3* RHC (Sum of weights will be 1).