December 27, 2023

OpenEEmeter 4.0: Final Model Specification and Results

Lead Contributors

Travis Sikes, Senior Data Scientist

Armin Aligholian, Data Scientist

Jason Chulock, Senior Software Engineer

Adam Scheer, Vice President of Applied Data Science

OpenEEmeter 4.0 Final Model Specification and Testing Results

This document contains three sections:

1. A technical summary of improvements from OpenEEmeter 3.0 to 4.0
2. Results of testing the OpenEEmeter 4.0
3. A detailed OpenEEmeter 4.0 model specification

I. OpenEEmeter 4.0 Daily Model Summary and Specification

Technical Summary of the 4.0 Model and Improvements

While the OpenEEmeter 3.0 model has provided an important start for the open source, meter-based measurement of gas and electric savings from demand side energy programs, it has long been known that further development could improve model performance in key areas. In particular, the 3.0 model exhibits systematic seasonal bias, in particular for populations of gas meters, as well as weekend/weekday bias, observed frequently among populations of commercial electric meters. The OpenEEmeter 3.0 Daily model also carries high computational cost; it can take up to 1 minute on average to fit a meter. This expense can be burdensome for users, especially for large datasets.

The OpenEEmeter 4.0 Daily model has improved these core performance elements.

Starting with computational cost, the 4.0 model makes improvements on two main fronts. First, the determination of balance points was modified from an exhaustive grid search to a global optimization scheme. Second, instead of explicitly fitting multiple models for the various possible combinations of heating, cooling, and temperature-independent load, inspiration was taken from Lasso regression and model coefficients are penalized. This enables use of only a single model fitting in which the lasso regression will favor model simplicity (as opposed to generating all possible candidate model formulations). This idea of a cost-benefit analysis is a common theme in OpenEEmeter 4.0; the cost of additional model complexity is justified by its benefit. Computational time improvements are as high as 100x (60 seconds to 0.5 seconds) in legacy mode (matching OpenEEmeter 3.0 within ~5% on an individual meter level). If OpenEEmeter 3.0-like results are desired, this performance increase

can be fully realized. However, this speed is leveraged in OpenEEmeter 4.0 to fit a better-performing model.

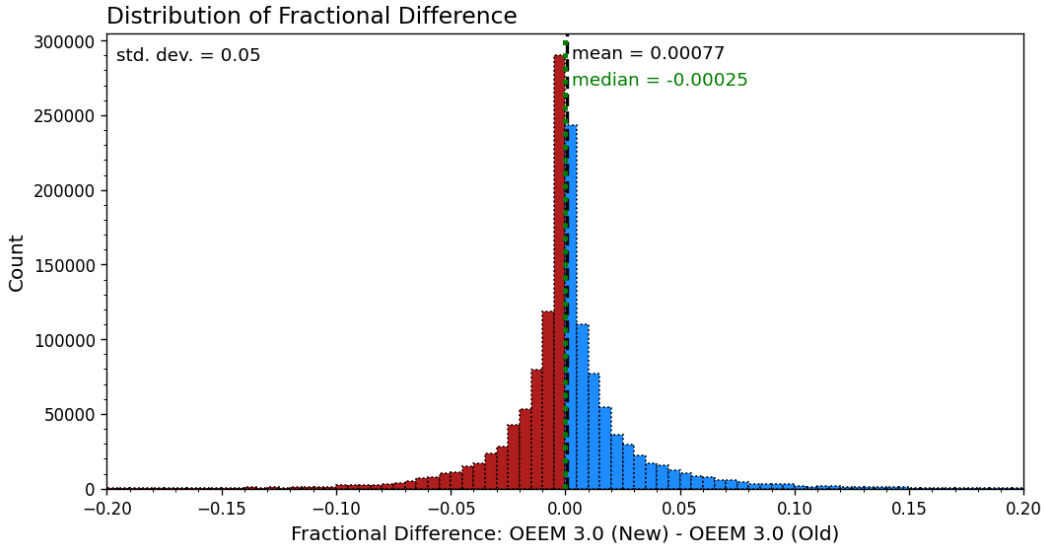
In OpenEEmeter 4.0 seasonal bias has been reduced by 84% and weekday/weekend bias by 95% through a combination of selective splitting into segments based upon season and weekday/weekend designations as well as updating the model formula to a smoothed 3-segment, piecewise linear model. When we detect that the building is behaving fundamentally differently, we enable splitting of time periods and selection of unique sub-models. These additional complexities are only allowed if their benefit overcomes a penalization barrier. To ensure improvements are predictive and the penalization parameters are reasonable, this model was tested extensively using a 10-fold, shuffle split cross validation scheme on ~6,000 meters. Of the 6,000 meters, 4,000 were residential gas, 1,000 electric residential, and 1,000 electric commercial meters. This distribution was chosen because seasonal bias was prominent in residential gas, but improvement here was not to be made at the expense of electric performance. Additional out-of-sample spot checks have shown that the changes to OpenEEmeter are robust and globally relevant meaning that OpenEEmeter 4.0 meets not only its stated goals but is an overall predictive improvement over OpenEEmeter 3.0.

II. Results of Final Model Testing

Bias Summary Table: Comparison between OpenEEmeter 3.0 Daily and OpenEEmeter 4.0 Daily of fractional biases during the noted timeframes for each sample studied:

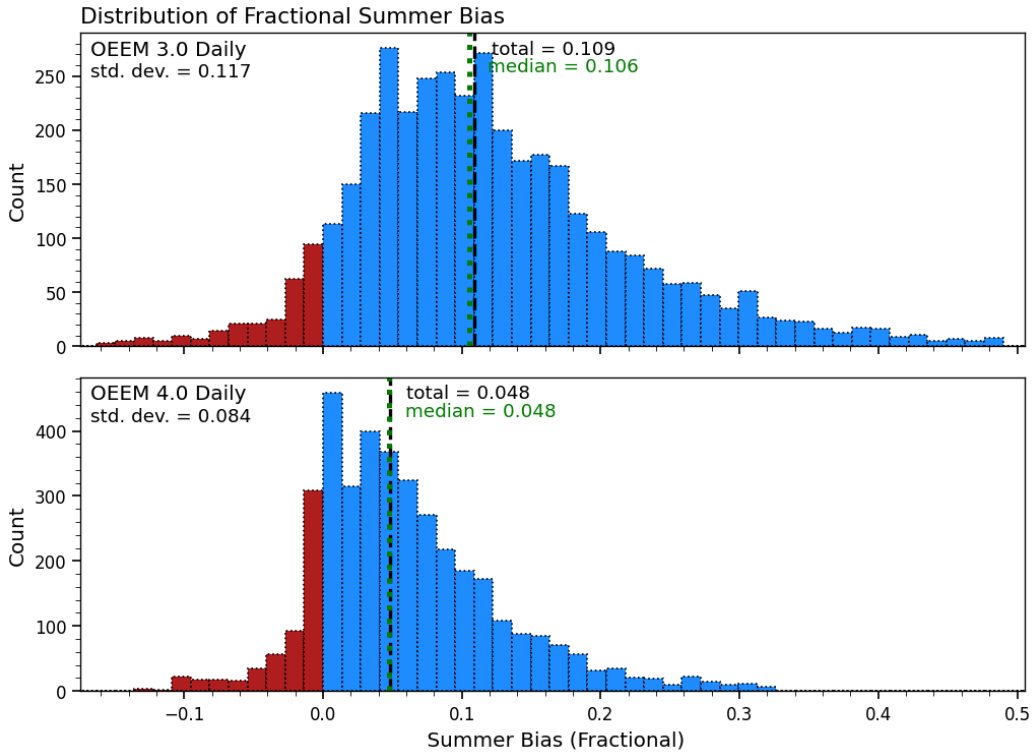
	Baseline OpenEEmeter Model	Comm Elec	Res Elec	Res Gas
% Annual Bias	3.0	0.07%	0.03%	0.16%
	4.0	0.59%	0.25%	0.55%
% Winter Bias	3.0	0.27%	-0.61%	-7.00%
	4.0	1.61%	-0.14%	-0.05%
% Shoulder Bias	3.0	-0.88%	2.21%	5.74%
	4.0	0.10%	1.03%	-1.08%
% Summer Bias	3.0	0.74%	-1.14%	12.21%
	4.0	0.16%	-0.02%	5.07%
% Weekday Bias	3.0	-4.66%	1.41%	1.44%
	4.0	0.59%	0.81%	1.18%
% Weekend Bias	3.0	14.26%	-3.27%	-2.91%
	4.0	0.61%	-1.10%	-0.99%

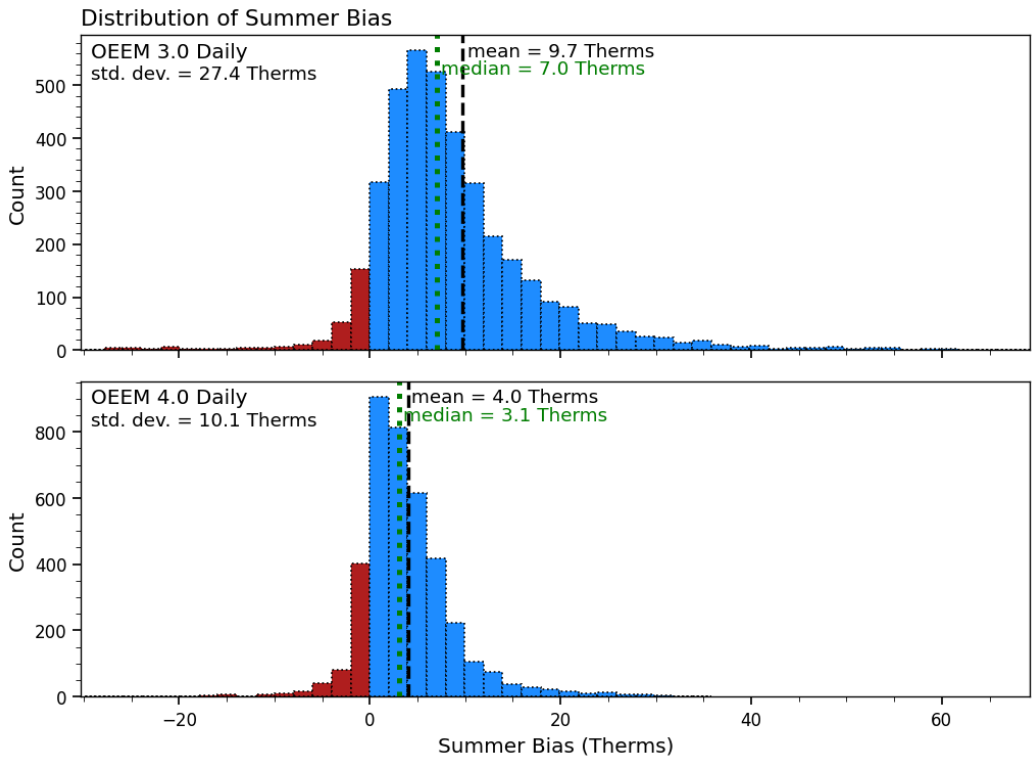
OpenEEmeter 3.0 vs. OpenEEmeter 4.0 in "3.0 Mode": Distribution of differences in baseline model among individual data points (Res Gas sample):

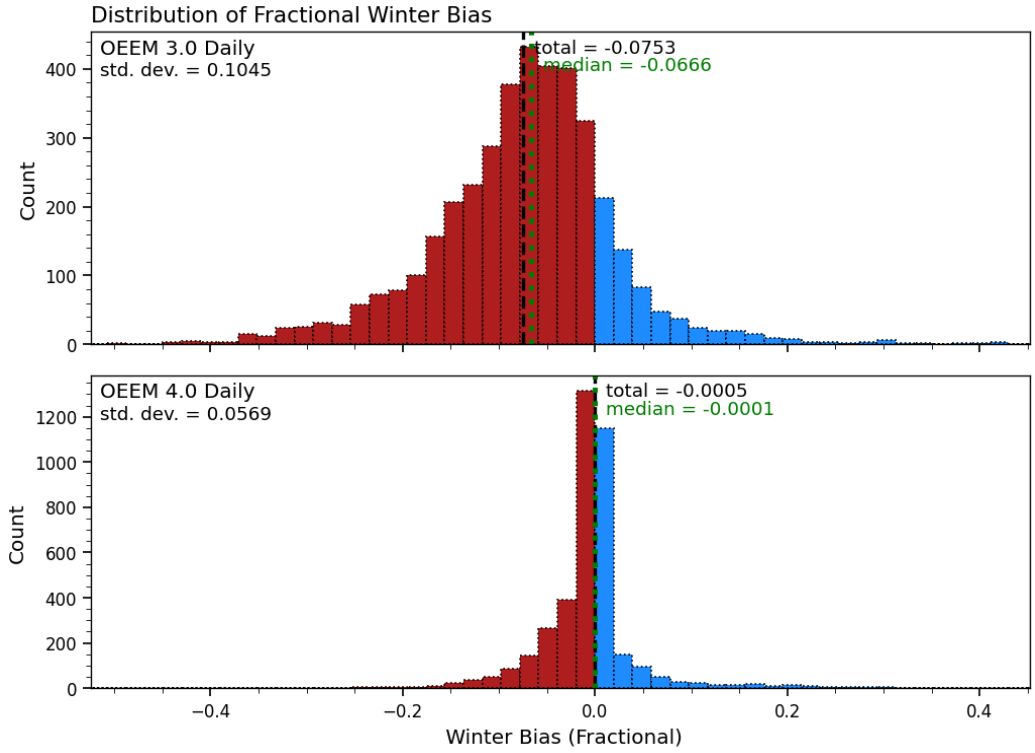


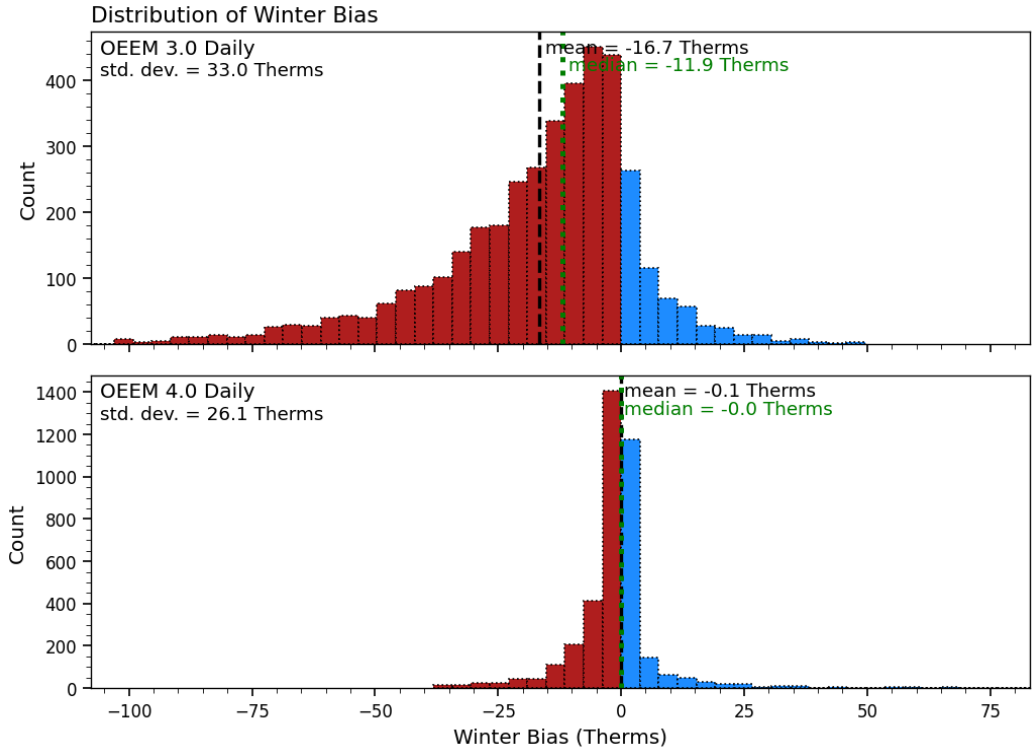
Residential Gas

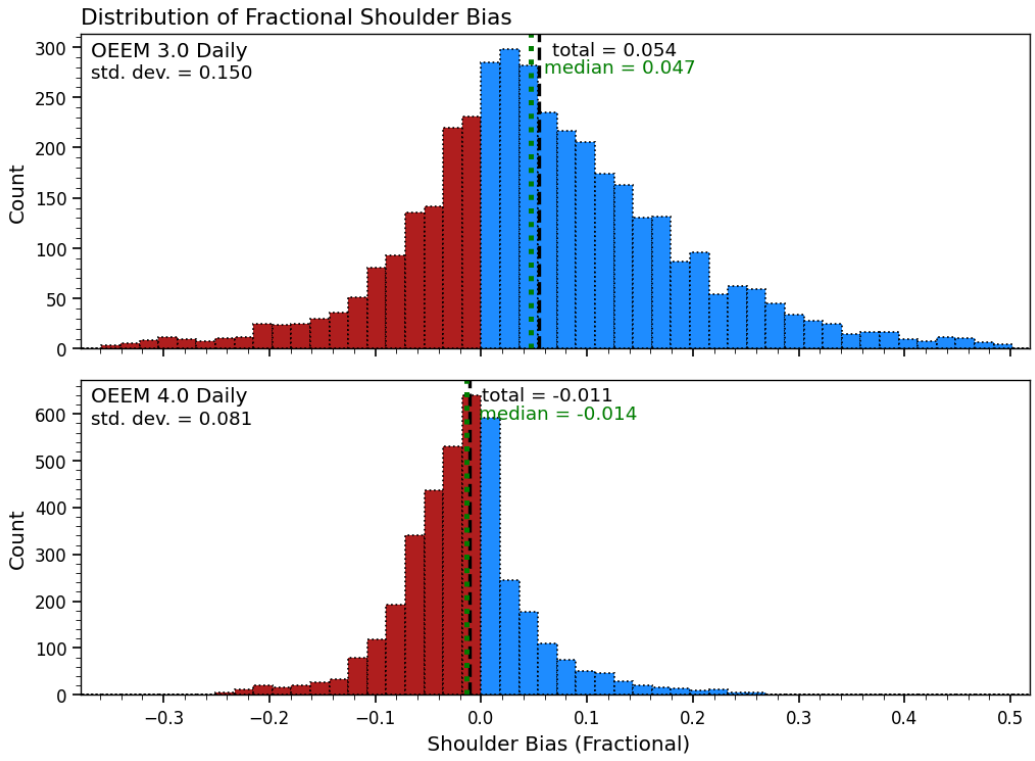
OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Distributions of baseline model seasonal bias:

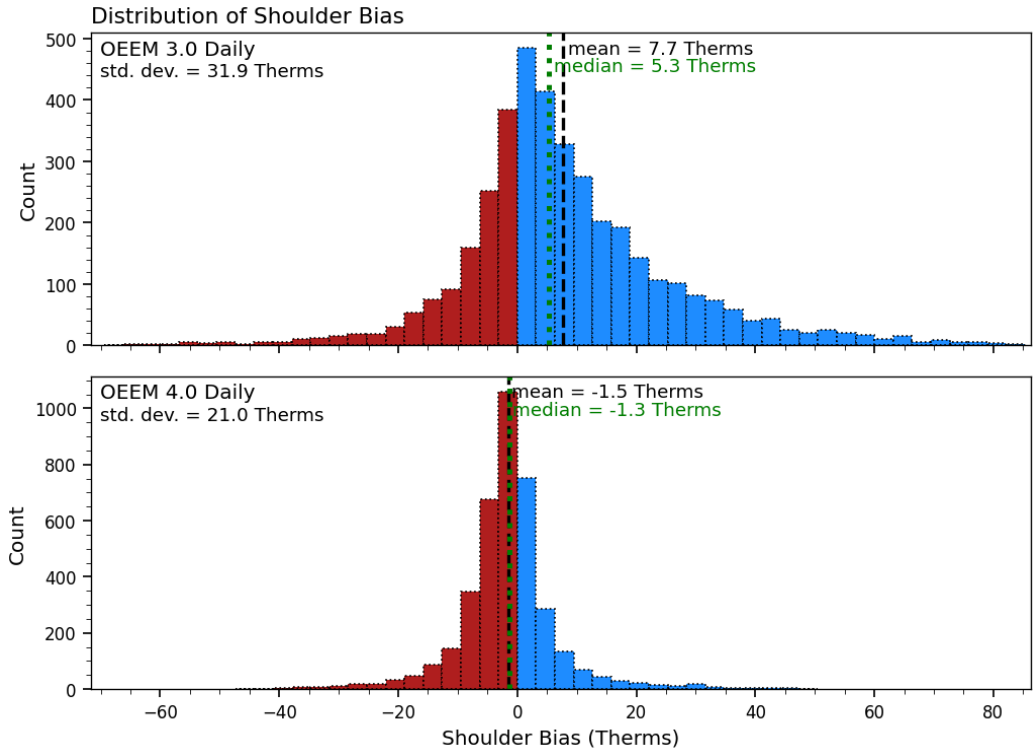




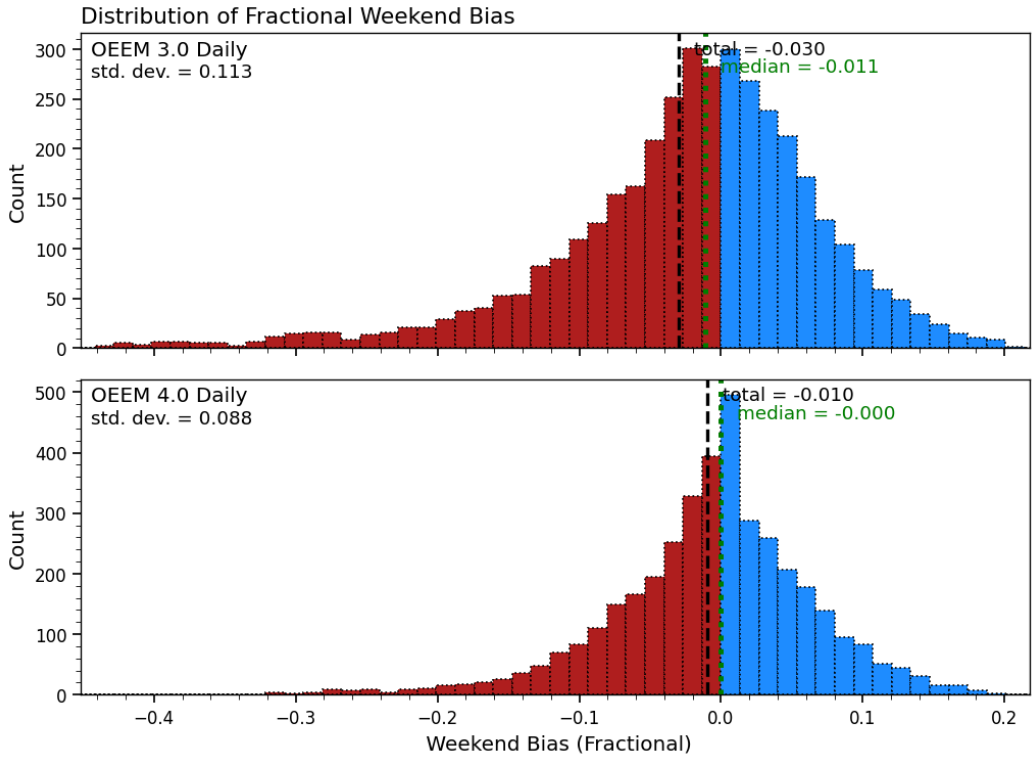

Distribution of Fractional Winter Bias
OEEM 3.0 Daily
std. dev. = 0.1045
total = -0.0753
median = -0.0666
Count
OEEM 4.0 Daily
std. dev. = 0.0569
total = -0.0005
median = -0.0001
Count
Winter Bias (Fractional)

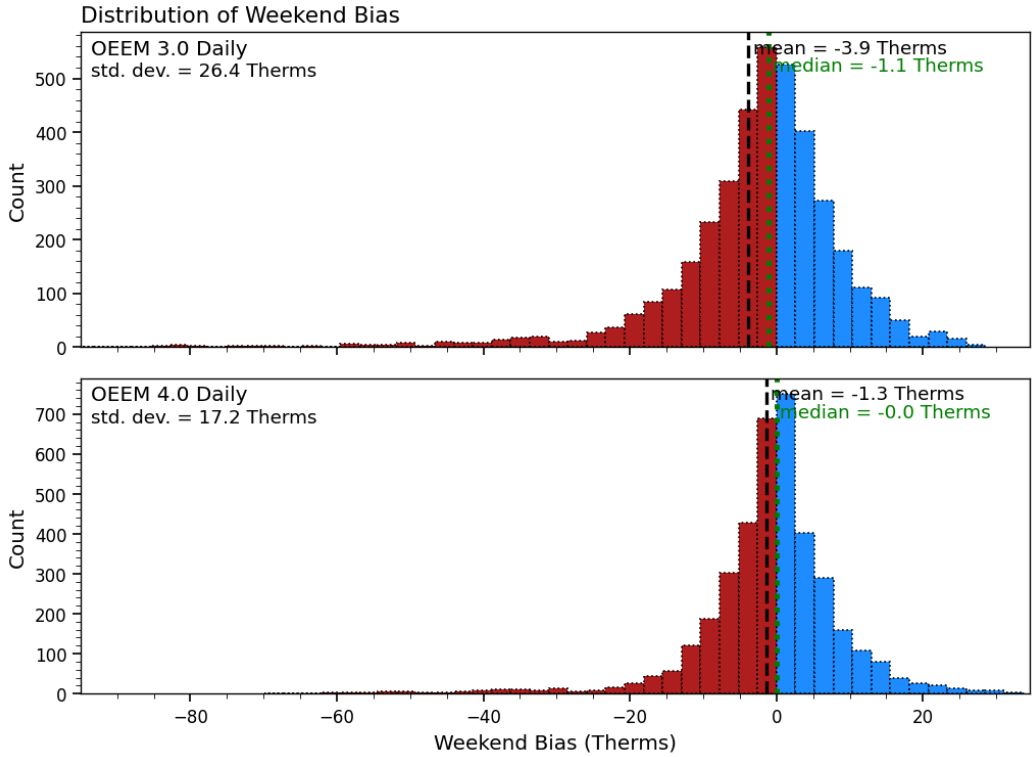

Distribution of Winter Bias
OEEM 3.0 Daily
std. dev. = 33.0 Therms
mean = -16.7 Therms
median = -11.9 Therms
Count
OEEM 4.0 Daily
std. dev. = 26.1 Therms
mean = -0.1 Therms
median = -0.0 Therms
Count
Winter Bias (Therms)


Distribution of Fractional Shoulder Bias
OEEM 3.0 Daily
std. dev. = 0.150
total = 0.054
median = 0.047
OEEM 4.0 Daily
std. dev. = 0.081
total = -0.011
median = -0.014
Count
Shoulder Bias (Fractional)

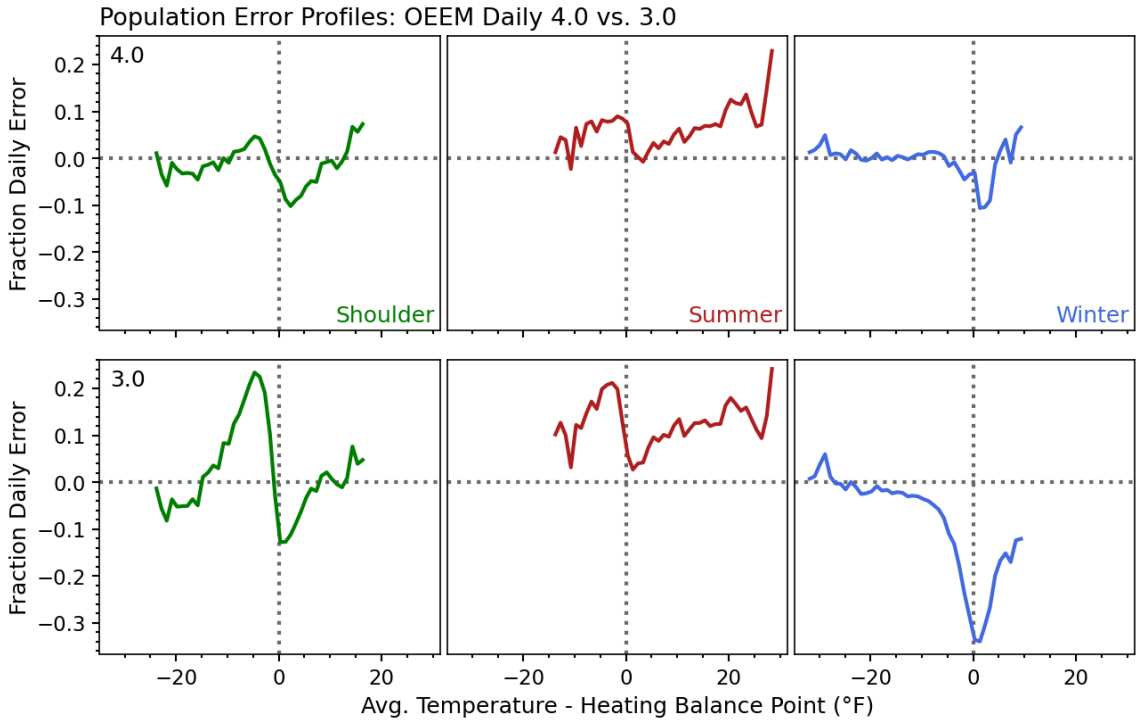

Distribution of Shoulder Bias
OEEM 3.0 Daily
std. dev. = 31.9 Therms
mean = 7.7 Therms
median = 5.3 Therms
OEEM 4.0 Daily
std. dev. = 21.0 Therms
mean = -1.5 Therms
median = -1.3 Therms
Count
Shoulder Bias (Therms)

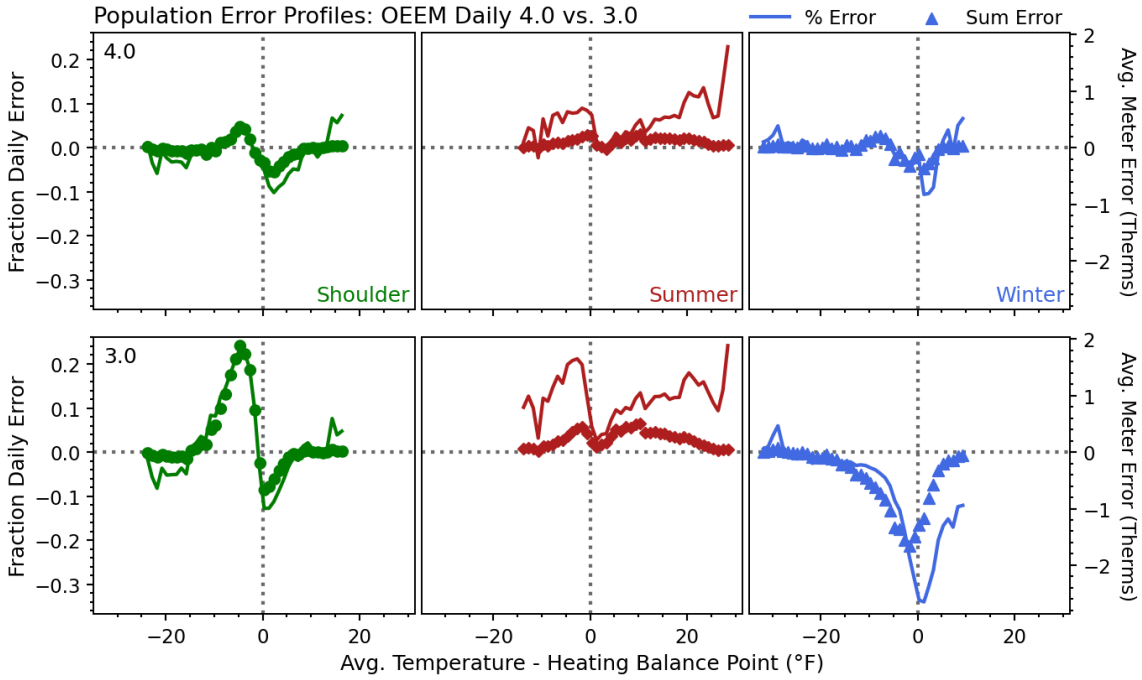
OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Distributions of baseline model weekend bias:



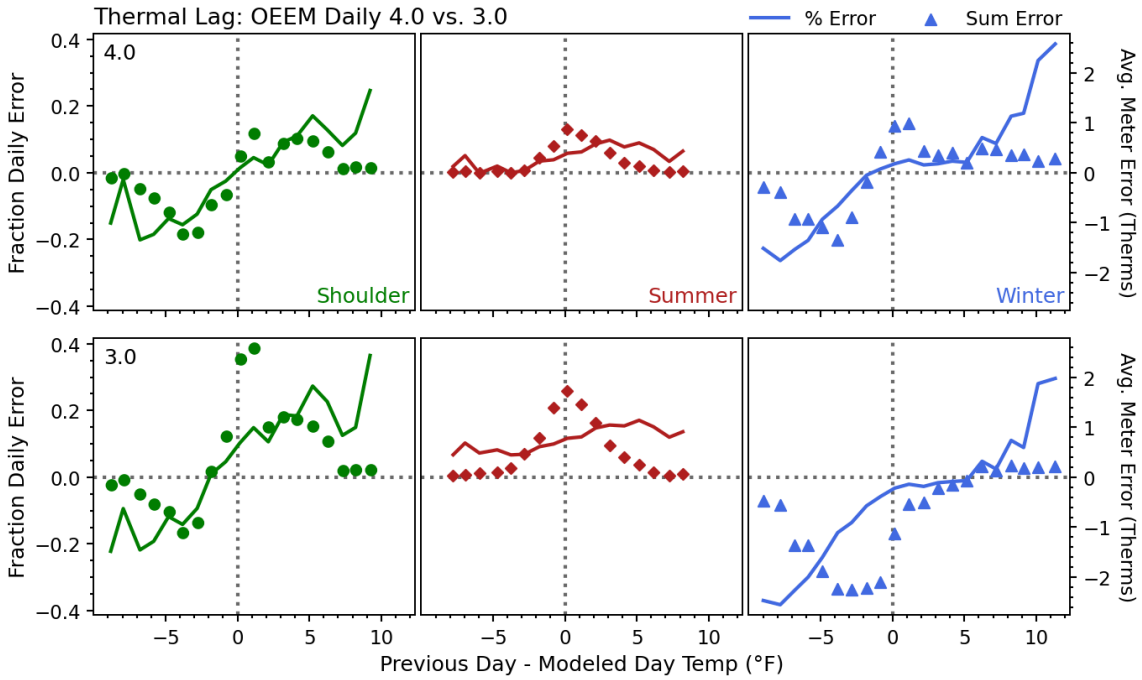


OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Population-level seasonal error profiles in the baseline model as a function of the difference between the average daily temperature and the heating balance point temperature:

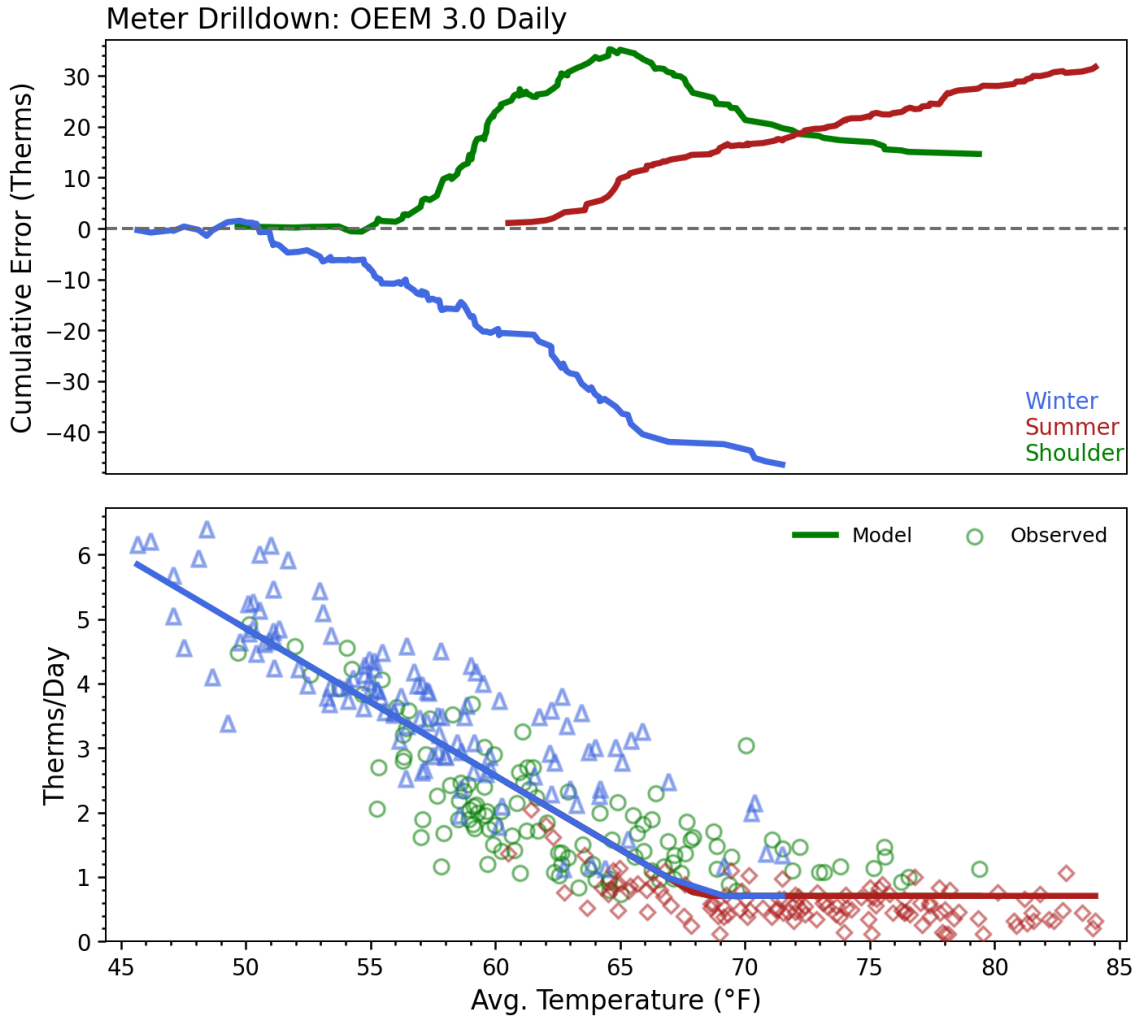


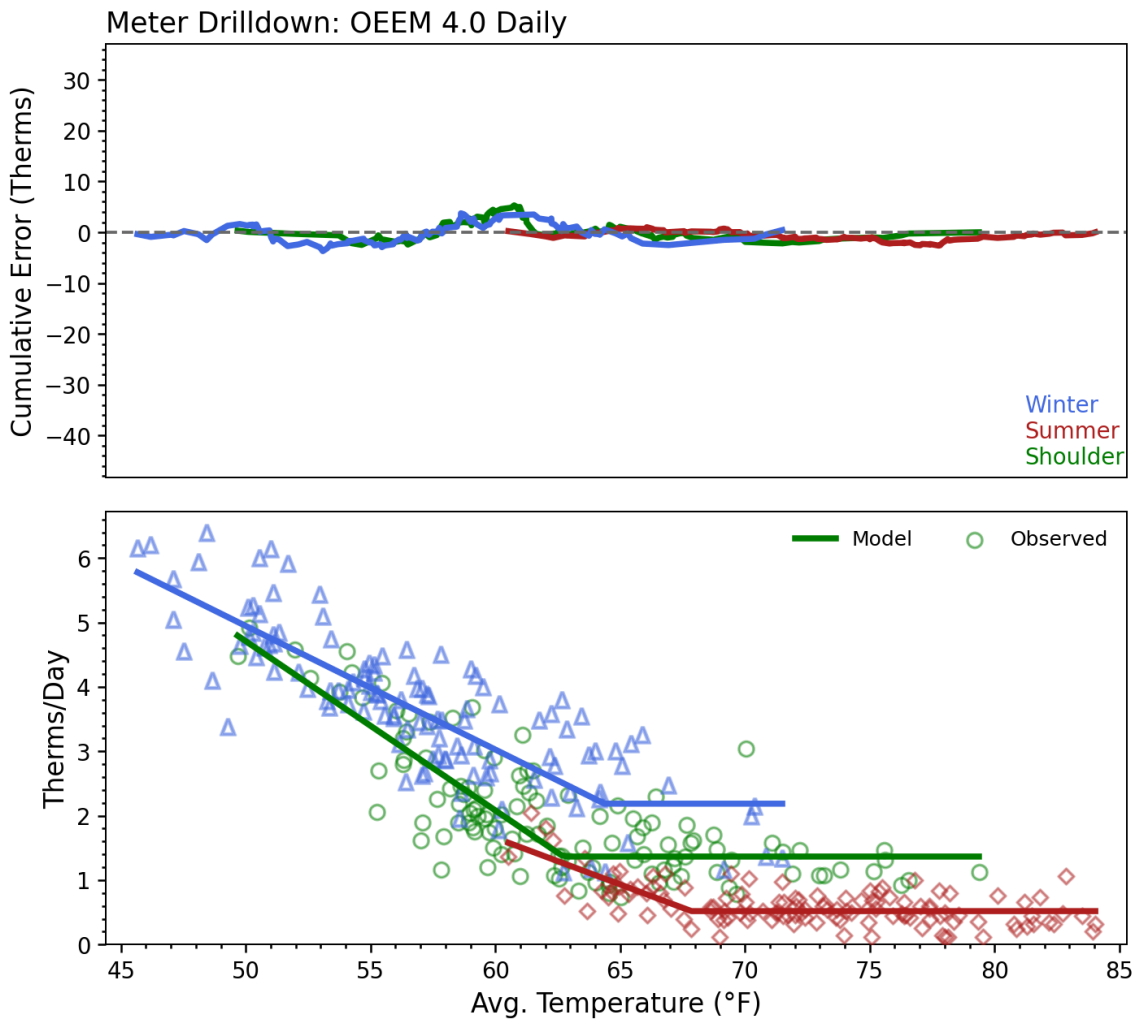


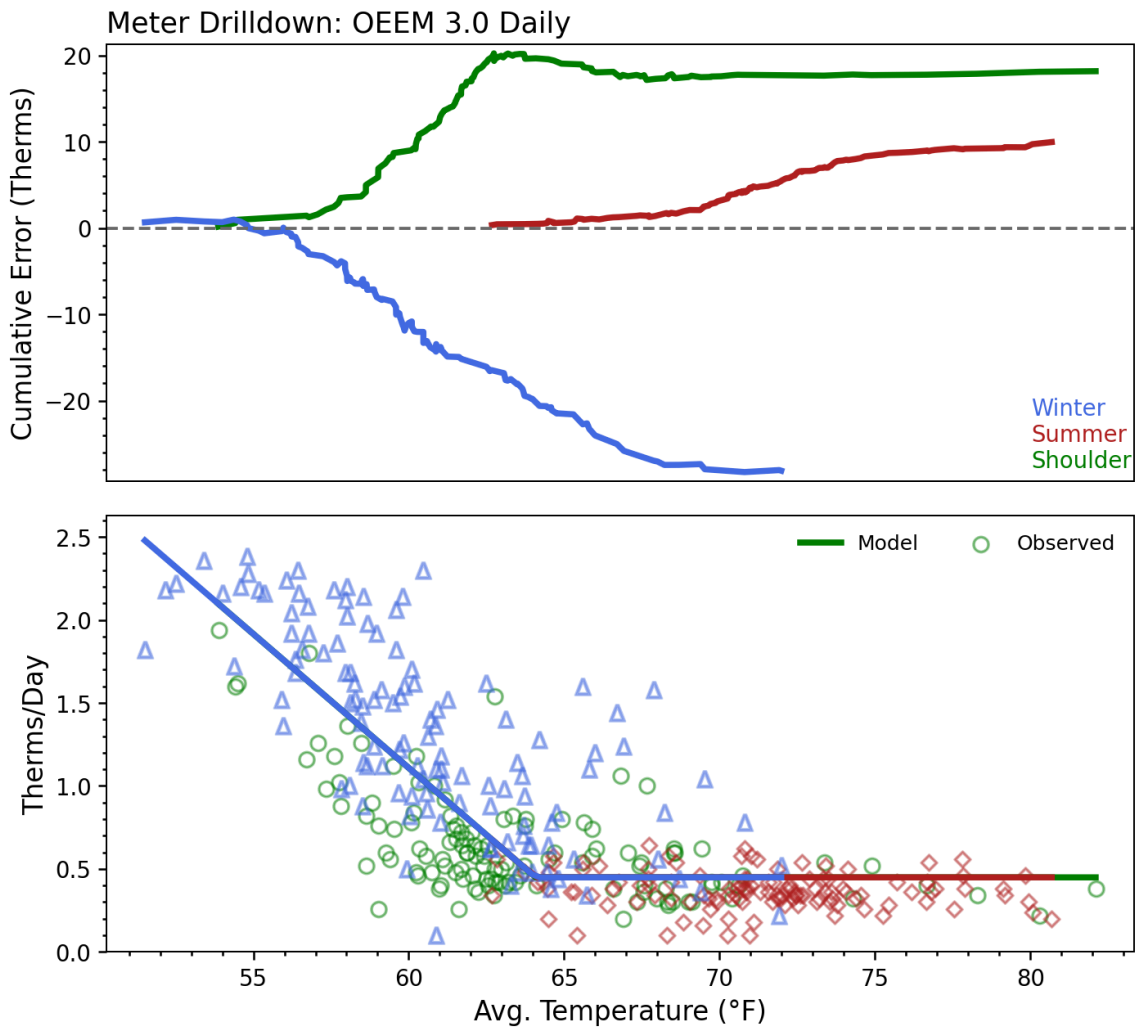
OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Population-level thermal lag error profiles in the baseline model as a function of the difference between the average daily temperature and the previous day's temperature:

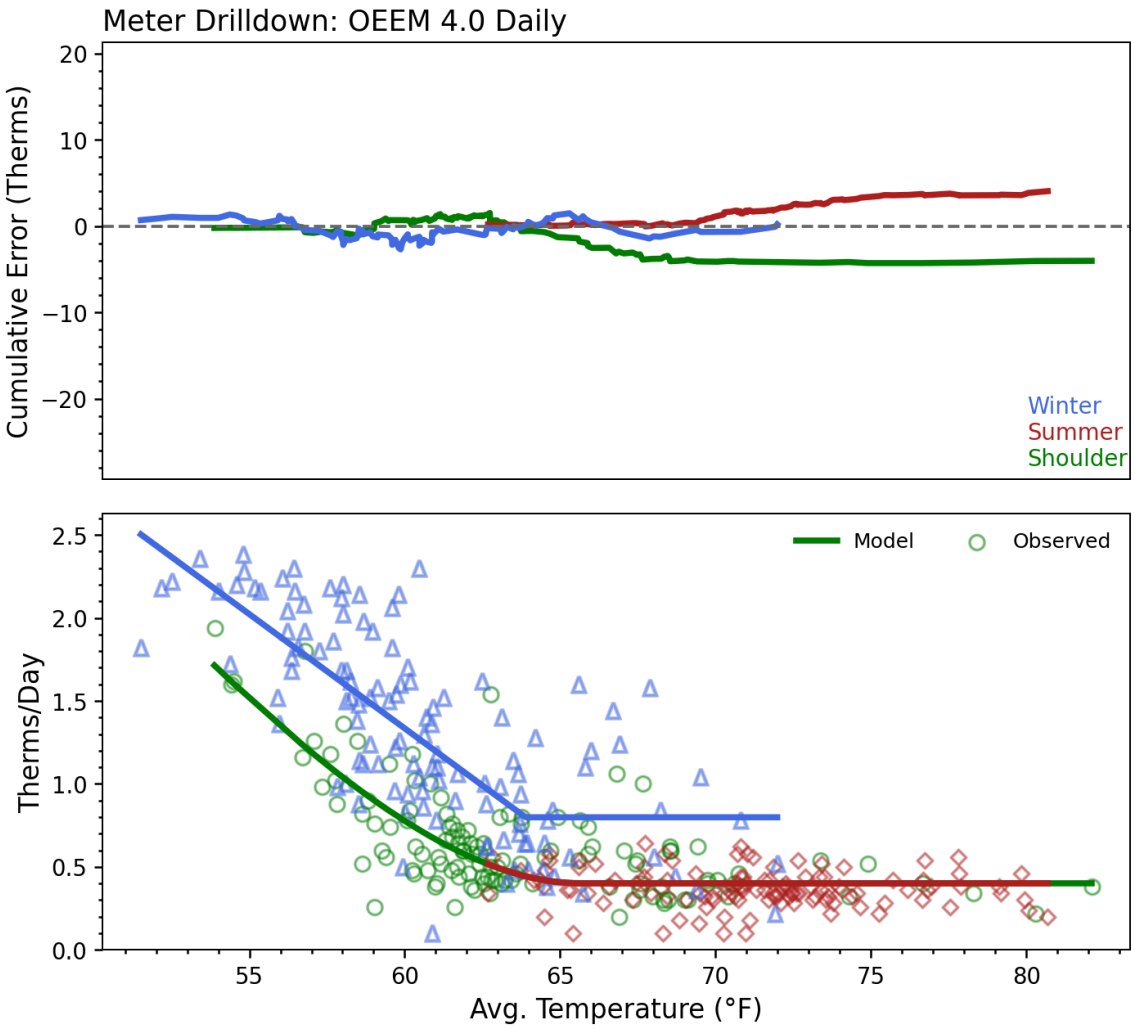


OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Individual meter model examples:



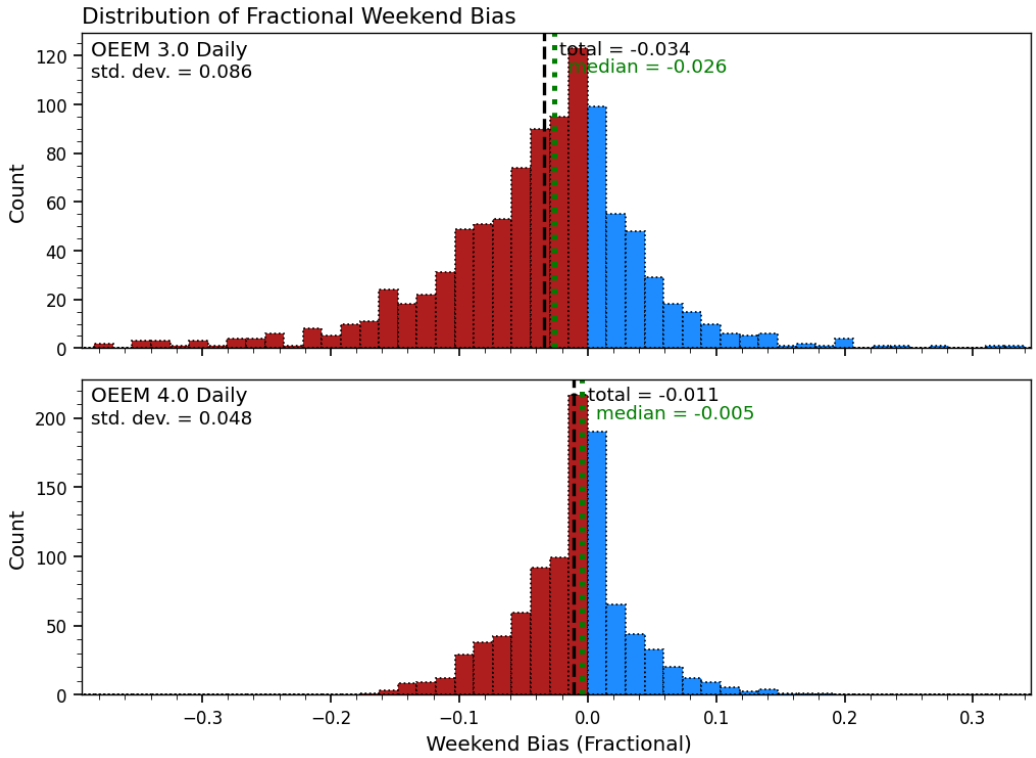

Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (Therms)
30
20
10
0
−10
−20
−30
−40
Winter
Summer
Shoulder
Therms/Day
Model
Observed
0
1
2
3
4
5
6
45
50
55
60
65
70
75
80
85
Avg. Temperature (°F)


Meter Drilldown: OEEM 3.0 Daily
Cumulative Error (Therms)
20
10
0
−10
−20
Winter
Summer
Shoulder
Therms/Day
2.5
2.0
1.5
1.0
0.5
0.0
Model
Observed
55
60
65
70
75
80
Avg. Temperature (°F)


Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (Therms)
20
10
0
−10
−20
Winter
Summer
Shoulder
Therms/Day
2.5
2.0
1.5
1.0
0.5
0.0
Model
Observed
55
60
65
70
75
80
Avg. Temperature (°F)

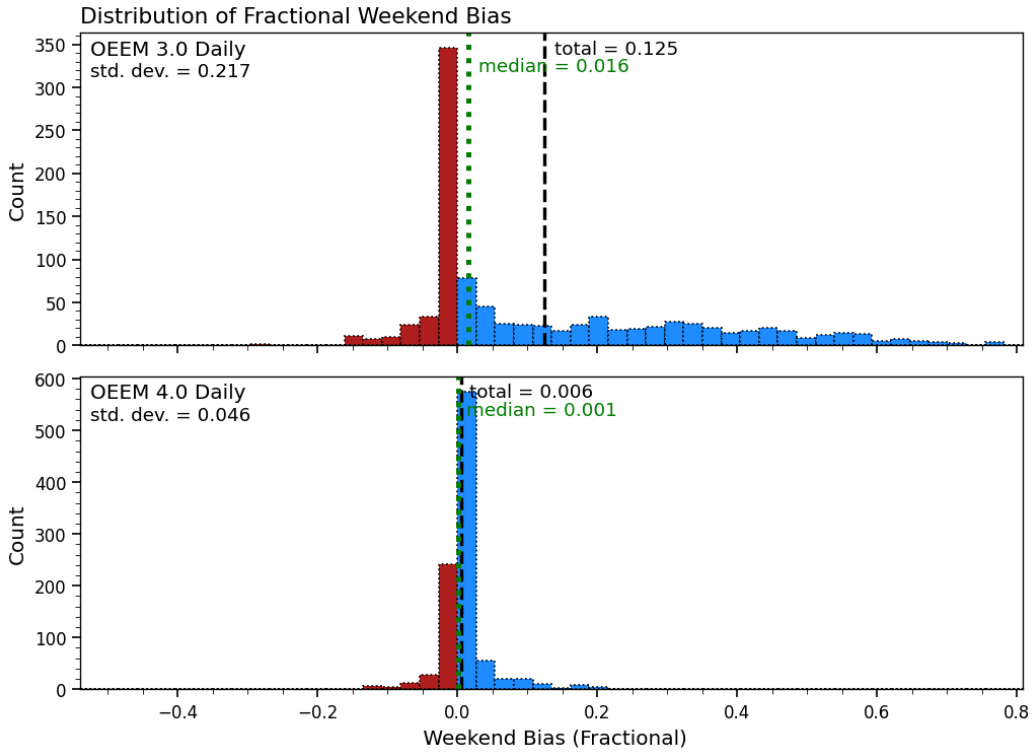
Residential Electric

OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Distributions of baseline model weekend bias:

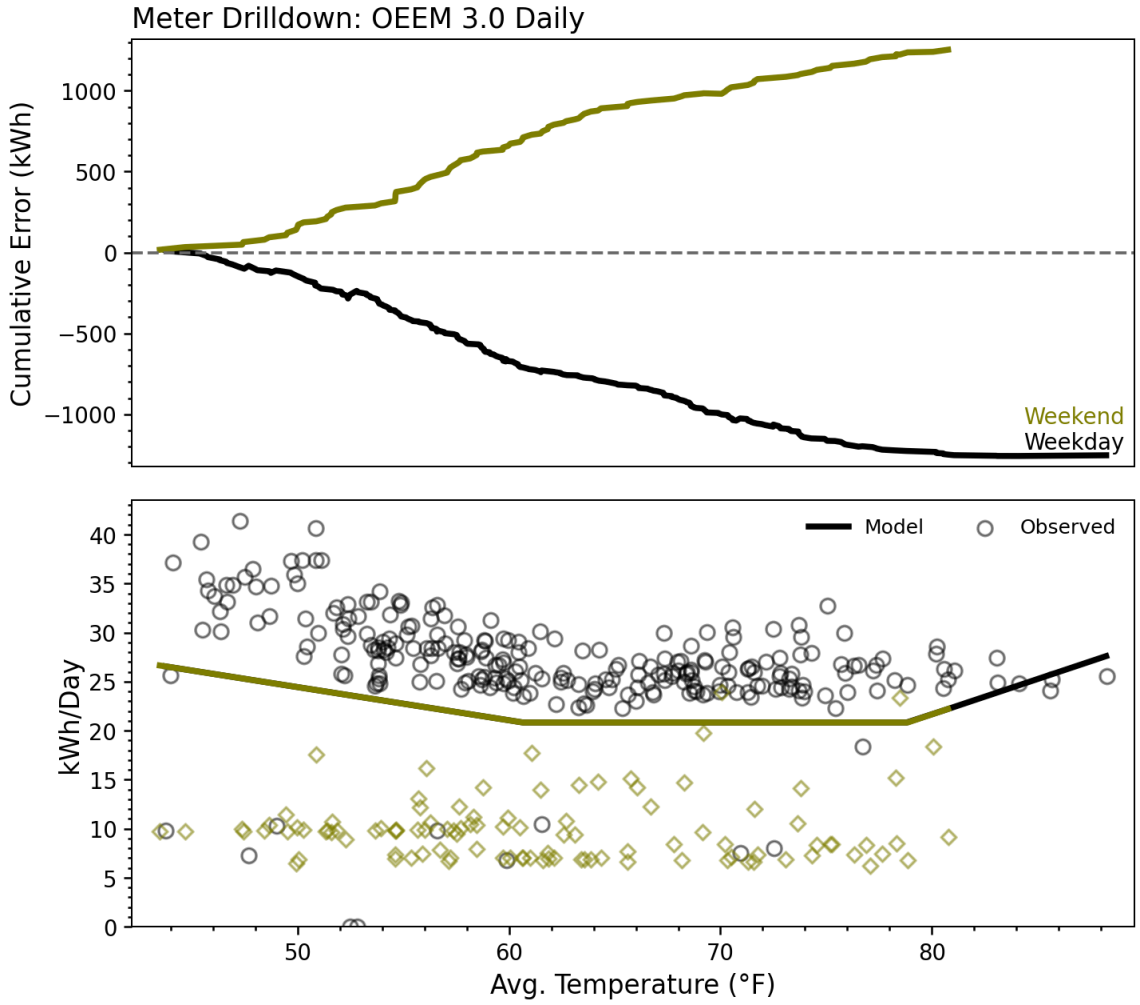


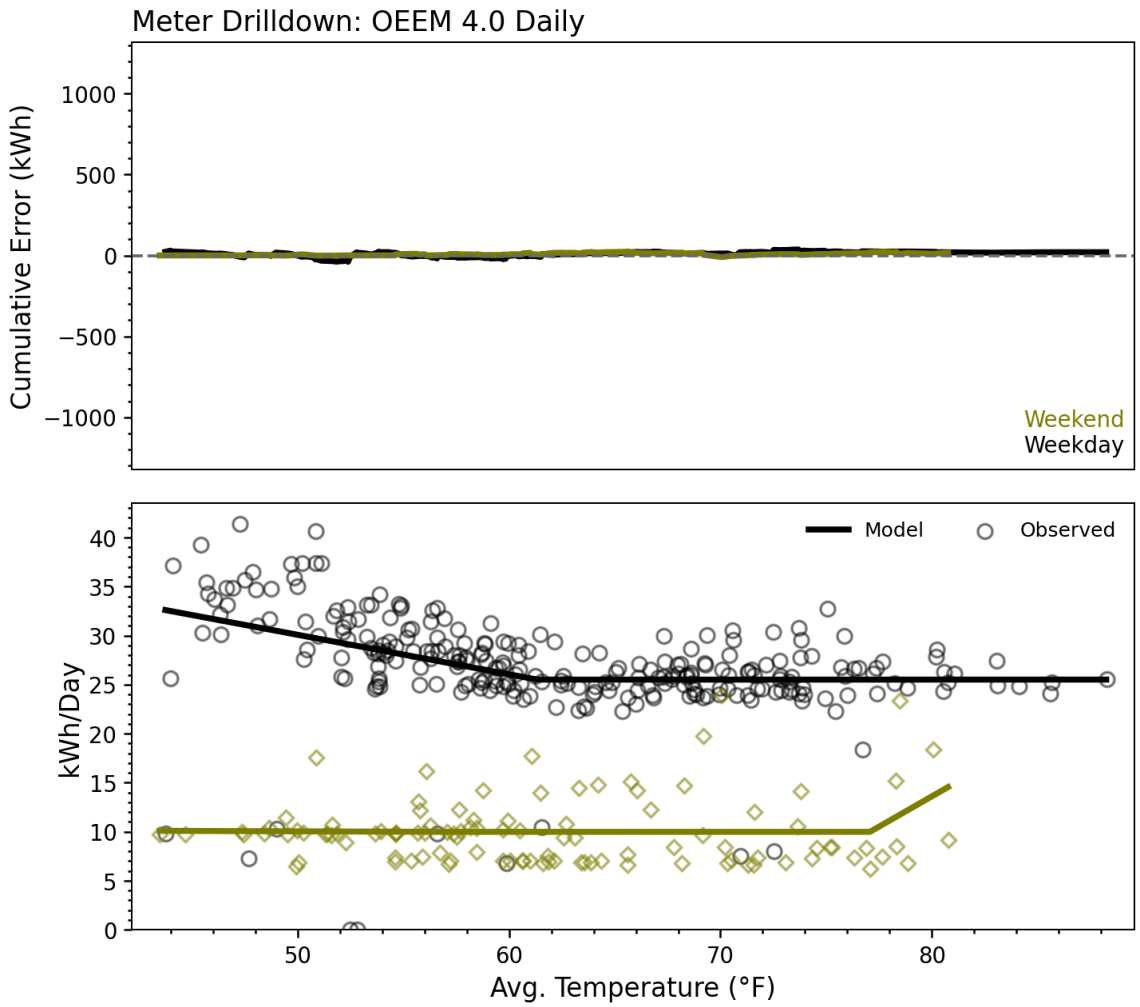
Commercial Electric

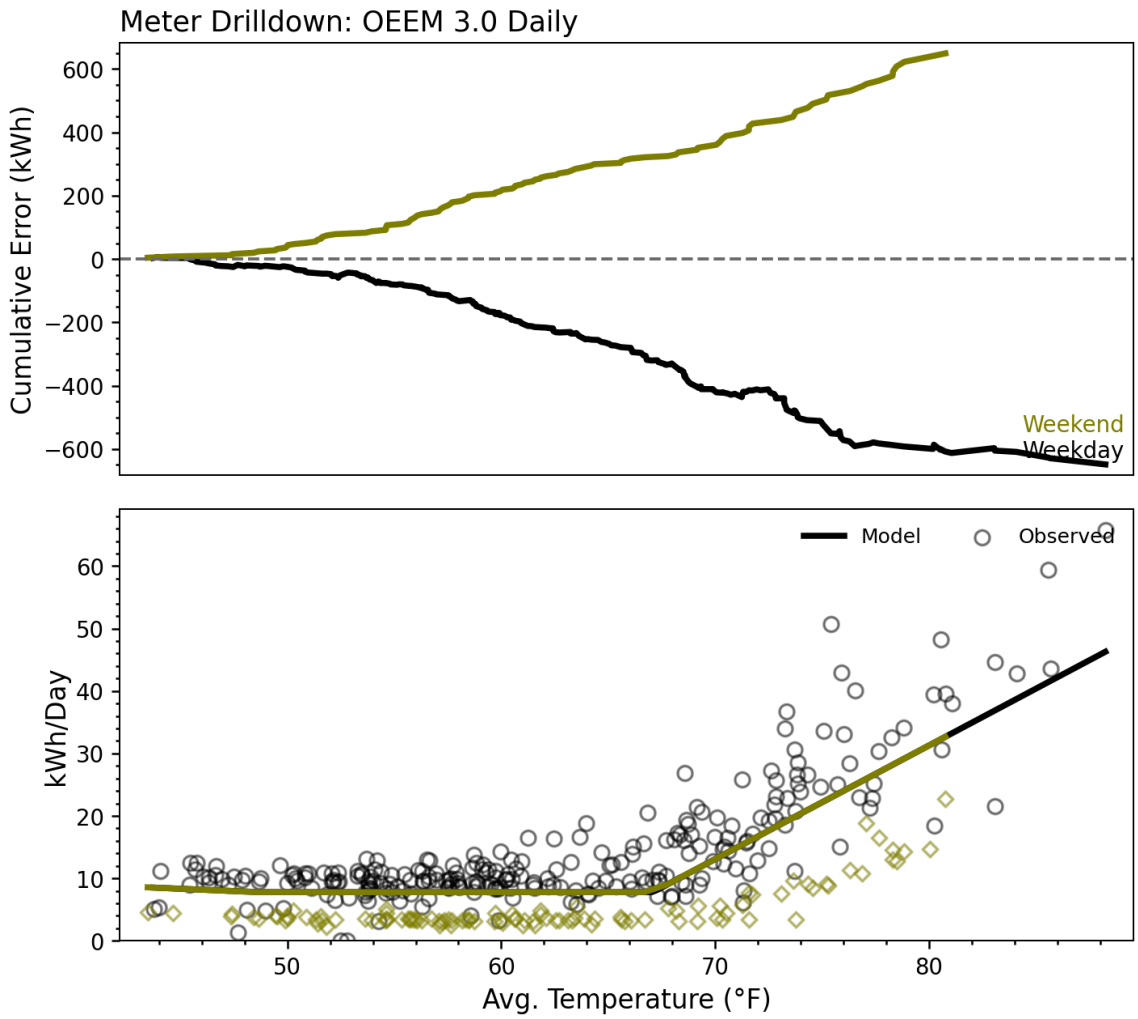
OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Distributions of baseline model weekend bias:

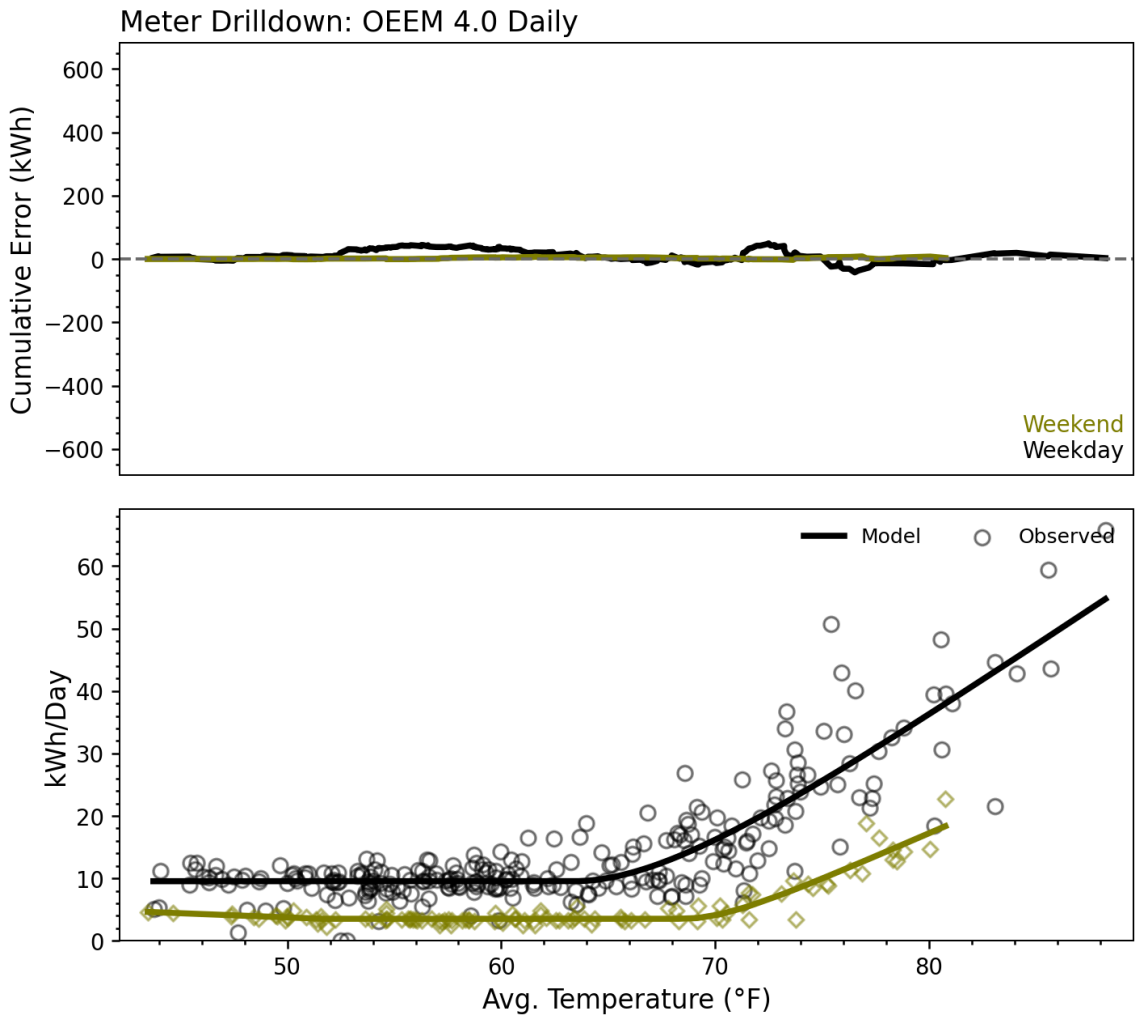


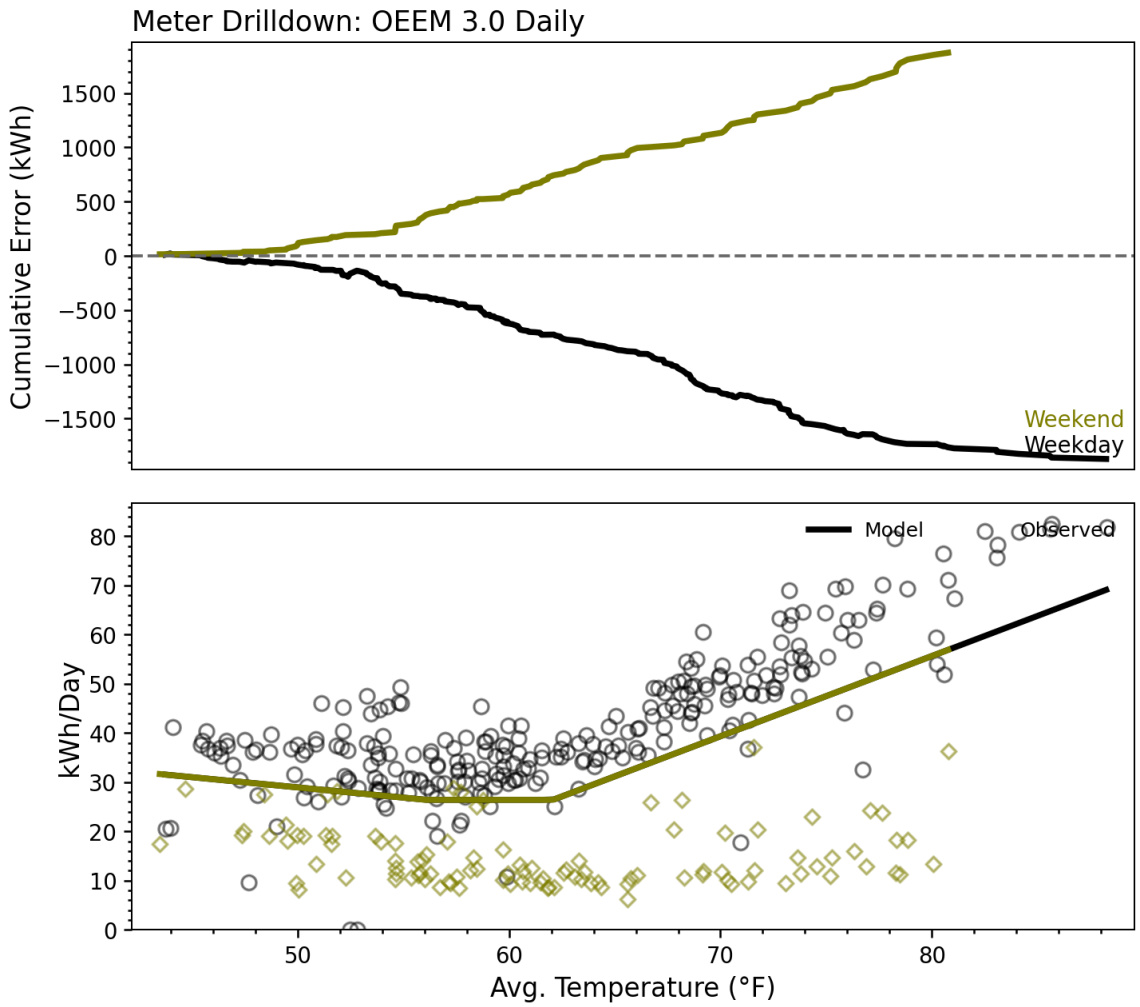
OpenEEmeter 3.0 vs. OpenEEmeter 4.0: Individual meter model examples:

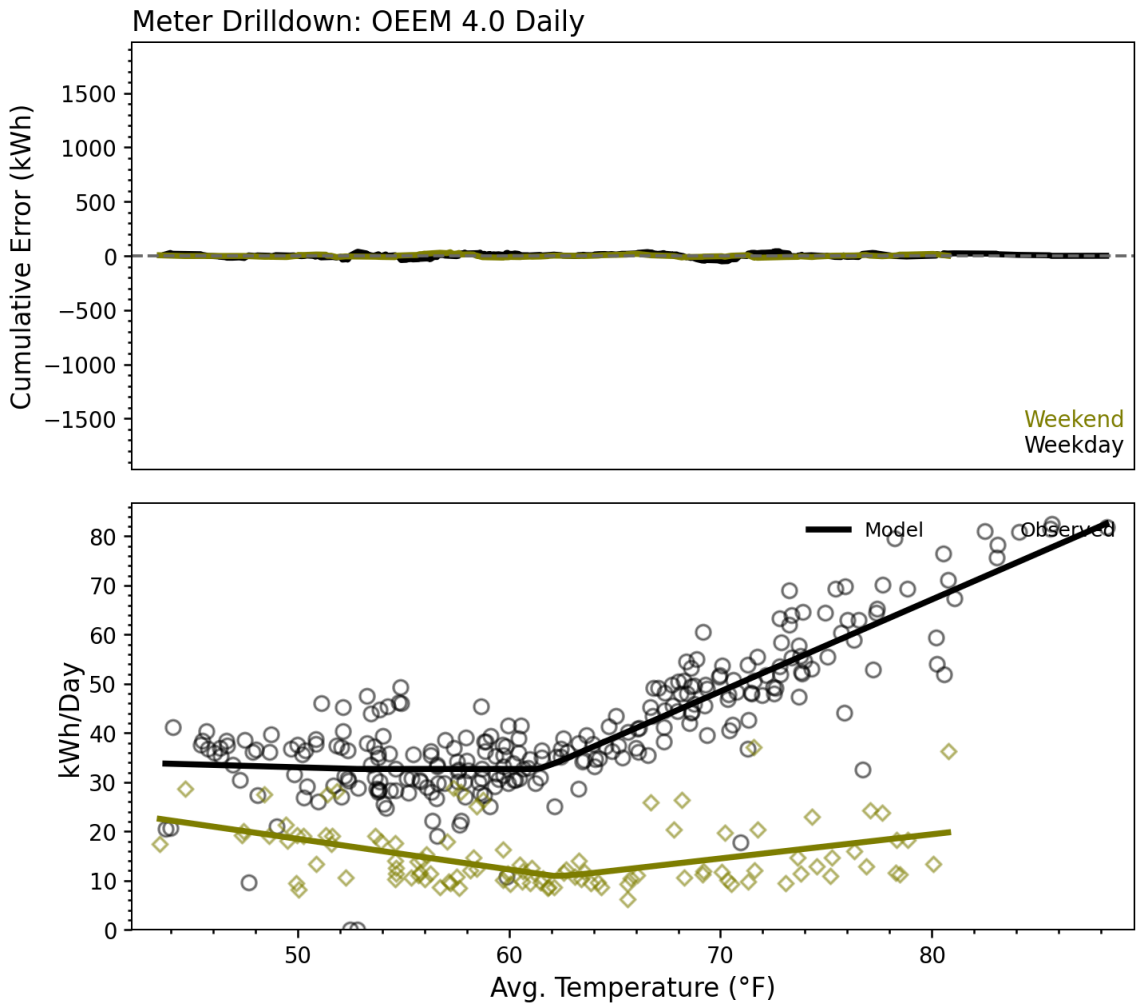


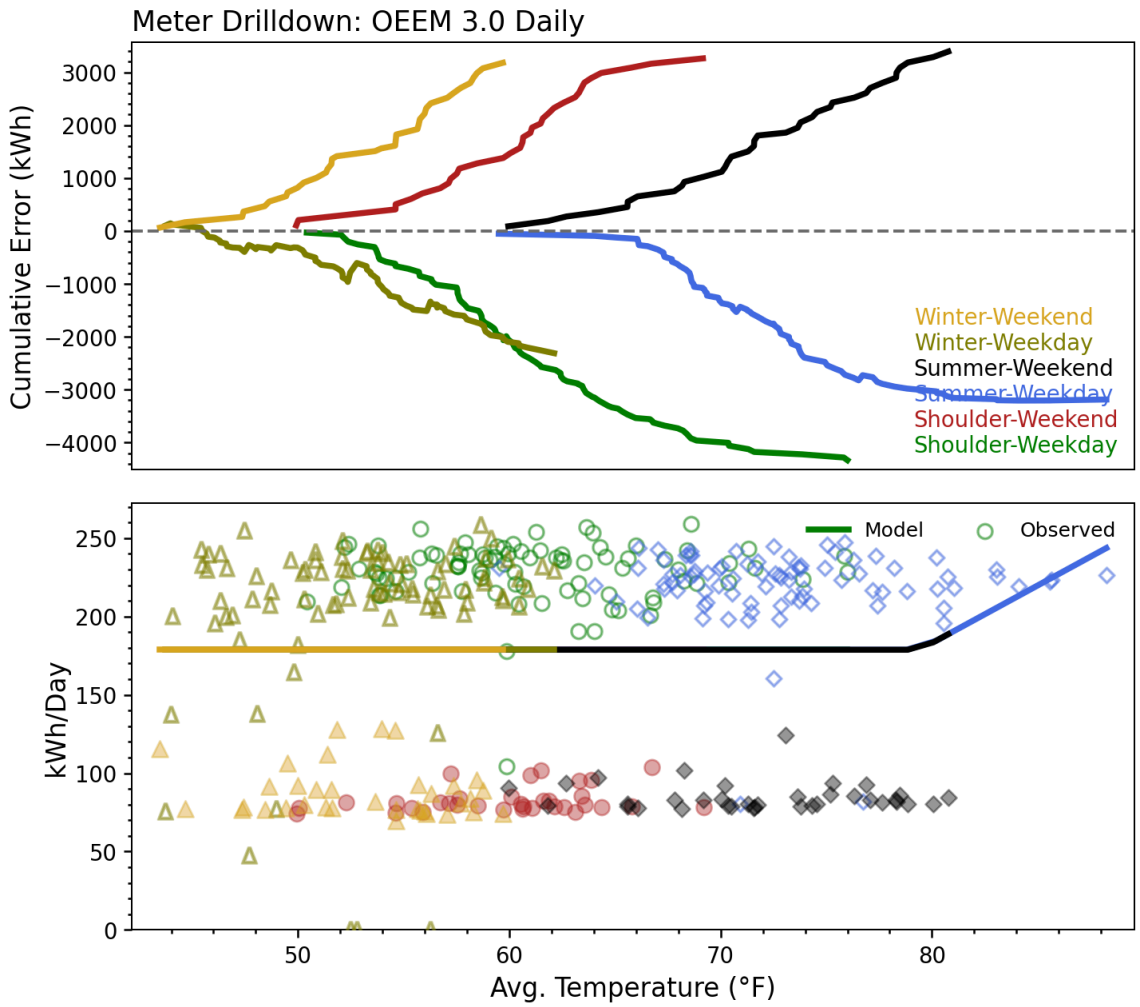

Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
1000
500
0
−500
−1000
Weekend
Weekday
kWh/Day
Model
Observed
40
35
30
25
20
15
10
5
0
50
60
70
80
Avg. Temperature (°F)

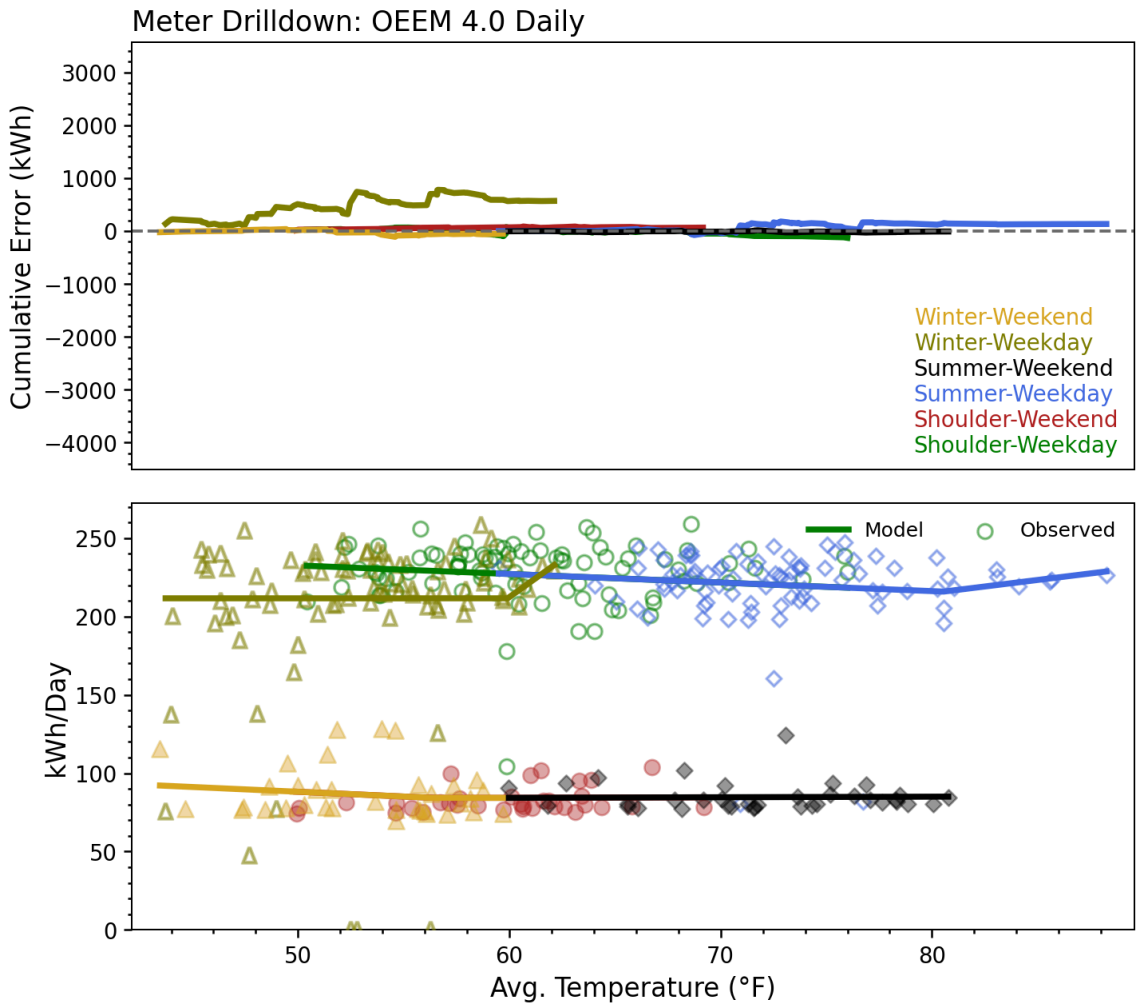

Meter Drilldown: OEEM 3.0 Daily
Cumulative Error (kWh)
600
400
200
0
−200
−400
−600
Weekend
Weekday
Model
Observed
kWh/Day
60
50
40
30
20
10
0
50
60
70
80
Avg. Temperature (°F)

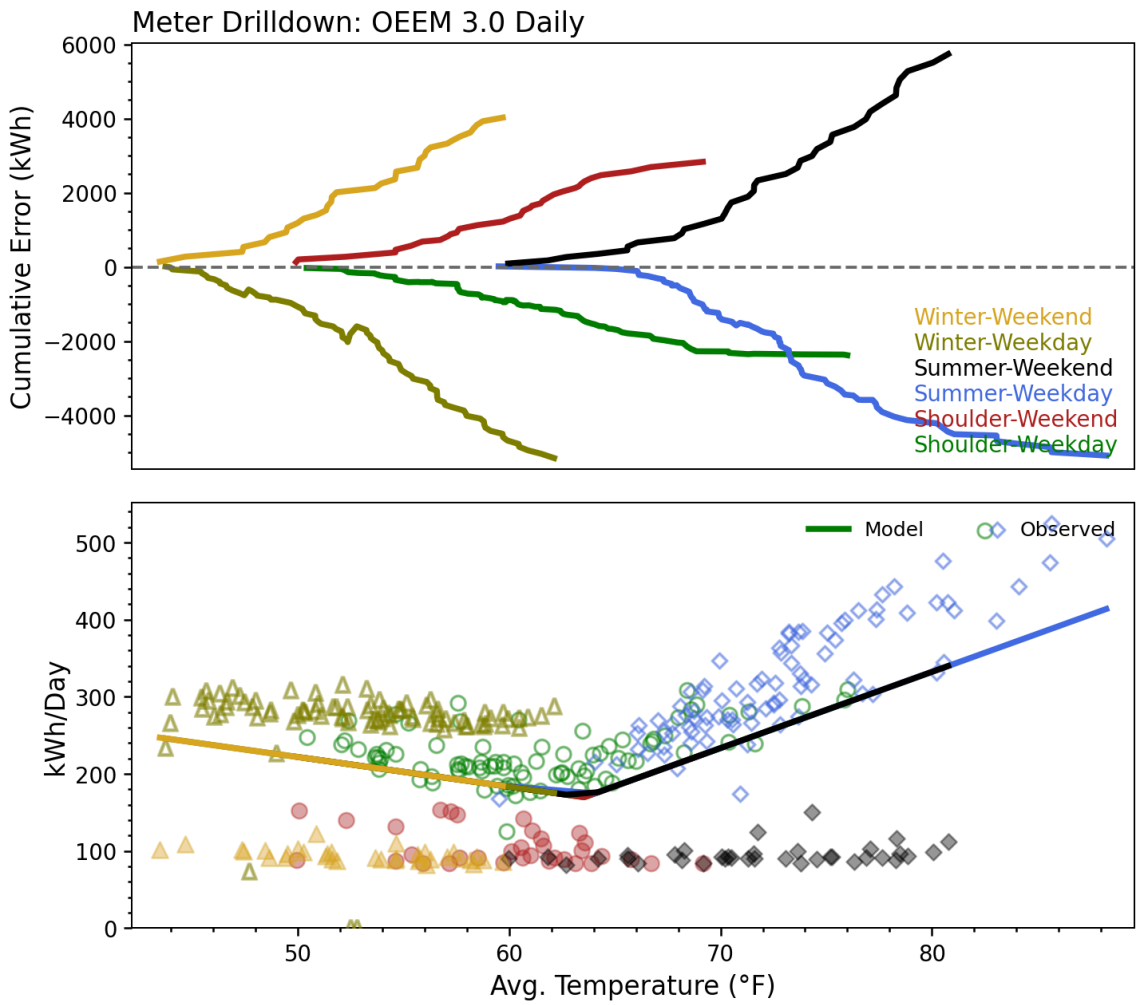

Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
600
400
200
0
−200
−400
−600
Weekend
Weekday
Model
Observed
kWh/Day
60
50
40
30
20
10
0
50
60
70
80
Avg. Temperature (°F)

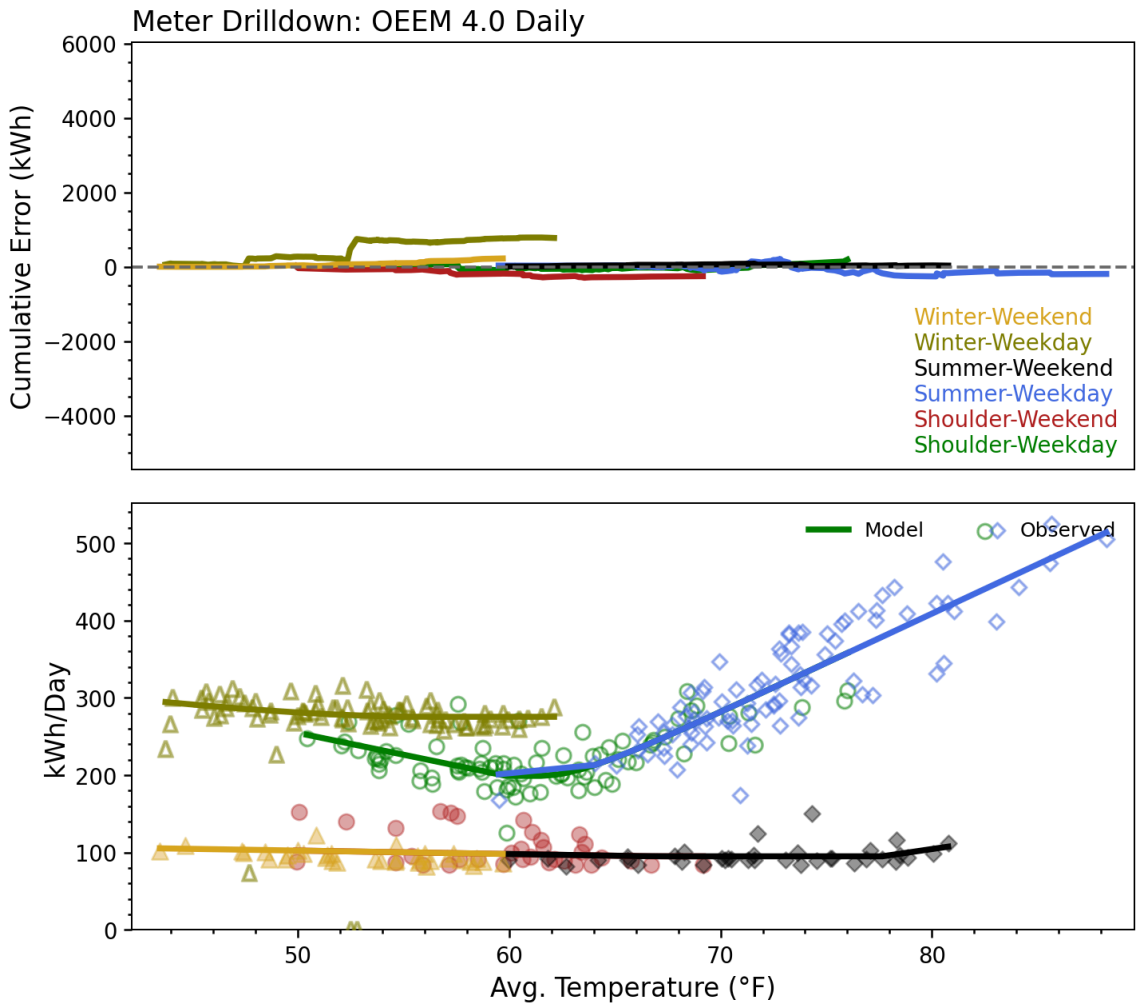

Meter Drilldown: OEEM 3.0 Daily
Cumulative Error (kWh)
1500
1000
500
0
−500
−1000
−1500
Weekend
Weekday
kWh/Day
Model
Observed
80
70
60
50
40
30
20
10
0
50
60
70
80
Avg. Temperature (°F)

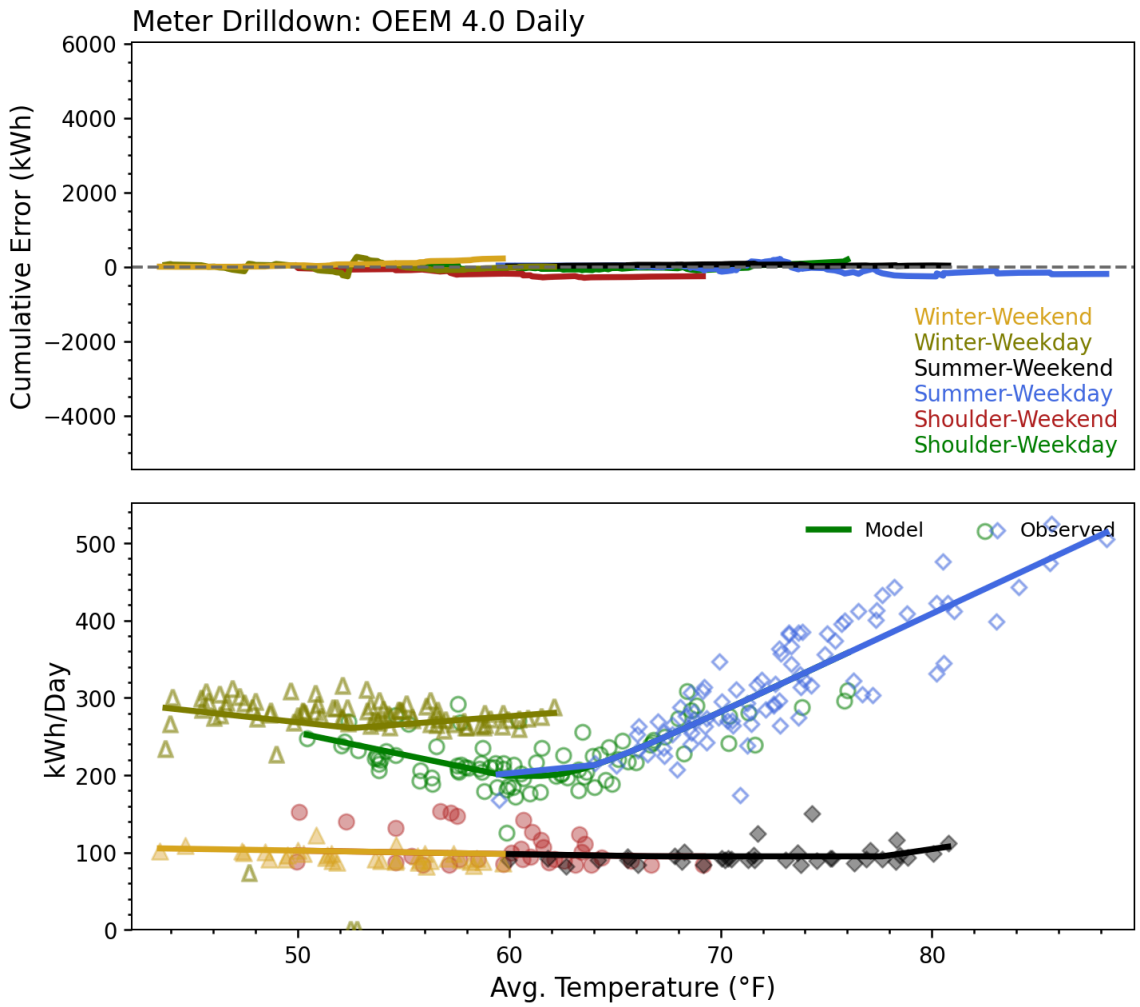

Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
1500
1000
500
0
−500
−1000
−1500
Weekend
Weekday
kWh/Day
Model
Observed
80
70
60
50
40
30
20
10
0
50
60
70
80
Avg. Temperature (°F)


Meter Drilldown: OEEM 3.0 Daily
Cumulative Error (kWh)
3000
2000
1000
0
−1000
−2000
−3000
−4000
Winter-Weekend
Winter-Weekday
Summer-Weekend
Summer-Weekday
Shoulder-Weekend
Shoulder-Weekday
kWh/Day
250
200
150
100
50
0
Model
Observed
50
60
70
80
Avg. Temperature (°F)


Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
3000
2000
1000
0
−1000
−2000
−3000
−4000
Winter-Weekend
Winter-Weekday
Summer-Weekend
Summer-Weekday
Shoulder-Weekend
Shoulder-Weekday
kWh/Day
250
200
150
100
50
0
Model
Observed
50
60
70
80
Avg. Temperature (°F)

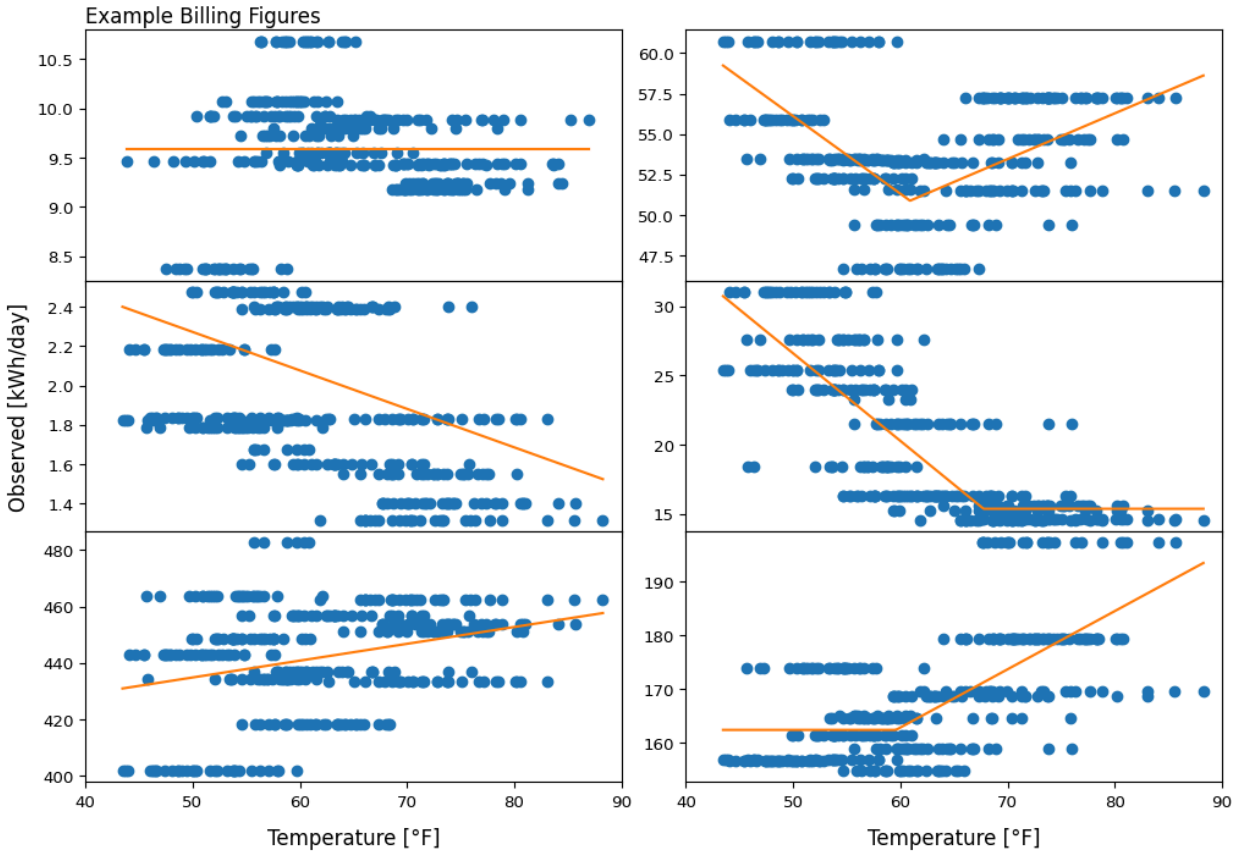

Meter Drilldown: OEEM 3.0 Daily
Cumulative Error (kWh)
6000
4000
2000
0
−2000
−4000
Winter-Weekend
Winter-Weekday
Summer-Weekend
Summer-Weekday
Shoulder-Weekend
Shoulder-Weekday
kWh/Day
500
400
300
200
100
0
Model
Observed
50
60
70
80
Avg. Temperature (°F)


Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
6000
4000
2000
0
−2000
−4000
Winter-Weekend
Winter-Weekday
Summer-Weekend
Summer-Weekday
Shoulder-Weekend
Shoulder-Weekday
kWh/Day
500
400
300
200
100
0
Model
Observed
50
60
70
80
Avg. Temperature (°F)

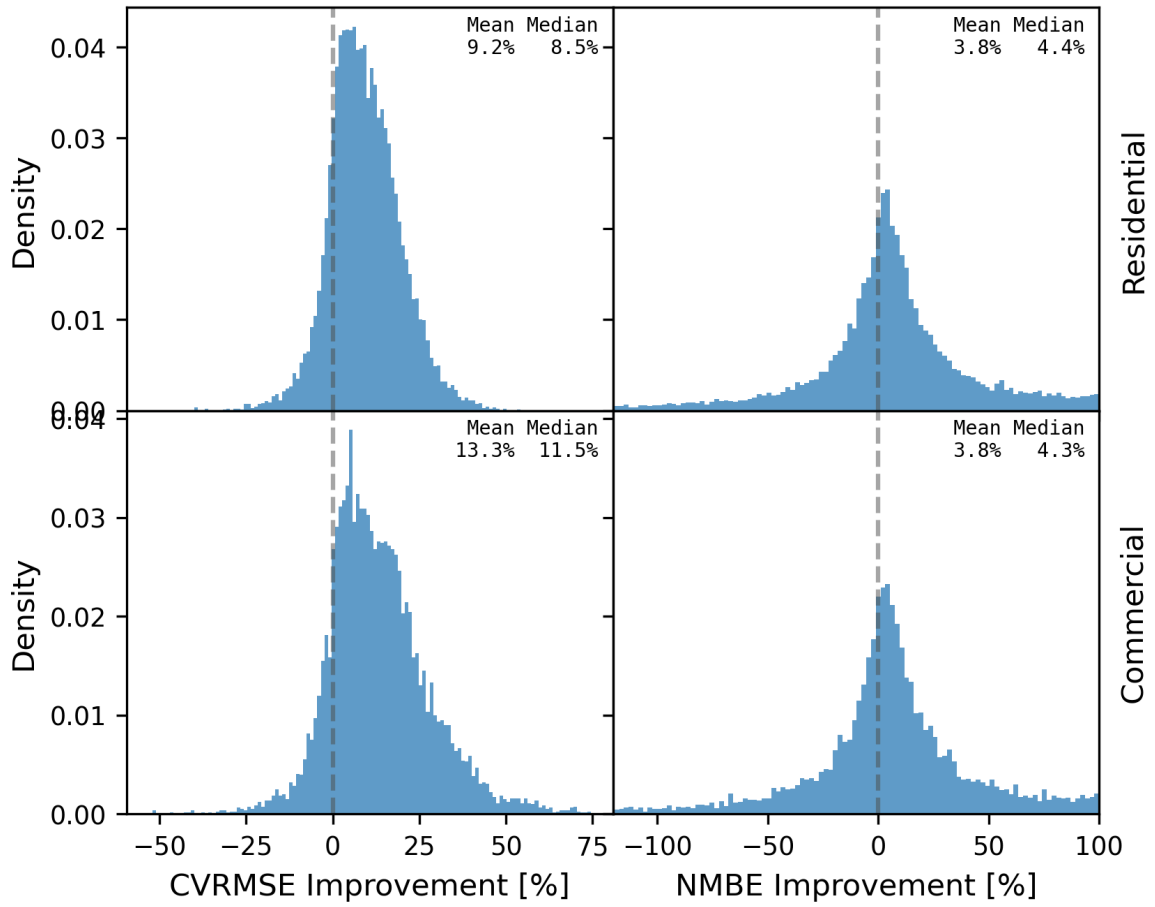

Meter Drilldown: OEEM 4.0 Daily
Cumulative Error (kWh)
6000
4000
2000
0
−2000
−4000
Winter-Weekend
Winter-Weekday
Summer-Weekend
Summer-Weekday
Shoulder-Weekend
Shoulder-Weekday
kWh/Day
500
400
300
200
100
0
Model
Observed
50
60
70
80
Avg. Temperature (°F)

OpenEEmeter 4.0 in "3.0 Mode" on Billing Data:

Example figures: How the model performs on various meters:



Comparison to CalTRACK: Overall model performance on 15270 residential meters and 8056 commercial meters. All meters are non-participant meters with no solar PV generation. The dashed line at 0 indicates the CalTRACK 2.0 billing model and the distribution represents how much the new billing model improves on those metrics. The new model is 87% faster.



III. Model Specifications

1. Model Options:

1.1. Date times are broken into 3 segments: summer, shoulder, and winter days.

1.1.1. Defaults that have been tested in the United States are summer = (June, July, August, September), shoulder = (March, April, May, October), winter = (January, February, November, December).

1.2. Date times should be defined to be either a weekday or weekend.

1.2.1. Note: This nomenclature will be retained throughout this document, but in the model formulation weekdays are not required to refer to Monday - Friday nor weekends to be Saturday and Sunday.

1.2.2. Defaults that have been tested are weekdays = (Monday - Friday), weekends = (Saturday, Sunday)

1.3. An uncertainty significance level should be assigned. The standard is $\alpha = 0.1$.

2. Data

2.1. The following data is required:

2.1.1. Date time

2.1.2. Temperature

2.1.3. Observed meter reading

2.2. Date times should be assigned as a summer (su), shoulder (sh), or winter (wi) day based on prior model options.

2.3. Date times should be assigned as weekday (wd) or weekend (we) based on prior model options.

2.4. Temperature determination should conform with existing OpenEEmeter 3.0 requirements.

3. Elimination of Allowed Splits

Each combination of season and weekday/weekend can potentially be assigned an independent mode (a "split"). The purpose of the split elimination step is to shrink the number of these possible splits so that the overall number of permutations can be reduced, thus limiting computation time.

3.1. Each segment (wd-su, we-su, …, we-wi) are fit with confidence ellipses based on their observed values and temperatures. For a given confidence ellipse, the major axis is multiplied by 1.4 and the minor axis by 0.89. These values were determined by testing.

3.1.1. These confidence ellipses can be further refined by removing outliers using a 2-D median filter and/or other methods.

3.1.1.1. In OpenEEmeter 4.0, a median filter with a size of 5 is used in conjunction with a filter to remove values outside 3 standard deviations of the confidence ellipse

3.1.2. The confidence ellipses are compared against each other.

3.1.2.1. If, for a given season, both the weekday and weekend data overlaps with another season then the first season is not allowed to be an independent split

3.1.2.2. If, within any season, the weekday data overlaps with the weekend data, then a weekday/weekend split is not allowed

4. Model Fitting

The "model" (or "full model") refers to the combination of submodels selected to predict energy consumption for the full year. Sub-models refer to the individual models of a split, whether that be we-su, wd-su, or any combination of wd/we-su-sh-wi. Submodels that do not have a weekday/weekend split are instead referred to as full week (fw) submodels.

4.1. All possible permutations of weekday/weekend and seasonal splits are determined based on the prior allowed splits and internal model configuration settings. A full model must represent a full year of seasons and days and follows OpenEEmeter 3.0 data sufficiency requirements.

4.2. The permutations or potential model splits have components in common that can be fit independently and combined to form the full model. All unique components are identified and receive a preliminarily fit. The actual fitting procedure can be found in section 5.

4.2.1. For example, A full model consisting of [fw-su + we-sh + wd-sh + fw-si] shares common submodels (underlined) with another full model consisting of [wd-su_wi+ wd-sh + we-su_sh_wi].

4.3. All potential full models are compared using a modified Bayesian Information Criterion (BIC) selection criterion given as Eq. 1. The full model with the lowest selection criterion is the model selected for the final fitting step.

$$BIC_{mod} = \underbrace{\ln\left(\frac{x}{N}\right)}_{\text{loss}} + \underbrace{0.24\frac{K}{N}\ln(N)^{2.061}}_{\text{penalty}} \quad (1)$$

where x is the loss value from optimization divided by the loss value of the unsplit model, both of which are functions of the residuals, K is the number of splits + 1, and N is the number of datapoints in the model. The formulation and coefficients were determined through testing.

4.4. Once the best potential model has been identified, its submodels are refined and refit according to section 5.

5. Preliminary Submodel Fitting

The purpose of the initial submodel fitting is not to get the best model possible, but instead to get an inexpensive estimate that serves to narrow down the potential model splits to a final selection that will then be further refined. In this step, submodels are all smoothed piecewise linear functions, with possible smoothing, that follow the following rules.

5.1. Heating slopes are constrained to always be zero or negative.

5.2. The temperature independent region is an intercept-only function.

5.3. Cooling slopes are constrained to always be zero or positive.

5.4. Smoothing is performed using an exponentially decaying function, Eq. 2, for the heating model and Eq. 3 for the cooling model.

$$E_{heat} = |\beta k| \left(\exp\left(\frac{1}{k} \left(T - T_{bp} \right) \right) - 1 \right) - \beta \left(T - T_{bp} \right) + C \tag{2}$$

$$E_{cool} = |\beta k| \left(\exp\left(-\frac{1}{k} \left(T - T_{bp} \right) \right) - 1 \right) + \beta \left(T - T_{bp} \right) + C \tag{3}$$

where β is the slope coefficient, k is the smoothing parameter, T_{bp} is the balance point temperature, and C is the intercept.

5.5. When optimizing the coefficients, k is derived from a 0 - 100% maximum smoothing factor ($k_{\%}$), and T_{bp} is also shifted such that the slope remains in the same temperature location. This pushes the T_{bp} to the right when smoothing is being applied in the heating model and left in the cooling model. These are defined in Eqs. 4 - 6.

$$k = k_{\%} \cdot (T_{CDD\ BP} - T_{HDD\ BP}) \tag{4}$$

$$T_{HDD\ BP\ new} = T_{HDD\ BP} + k_{HDD} \tag{5}$$

$$T_{CDD\ BP\ new} = T_{CDD\ BP} - k_{HDD} \tag{6}$$

where HDD refers to the heating model and CDD the cooling model.

5.5.1. If the sum of $k_{\%}$ is greater than 100%, they are divided by their sum to enforce that the summation is at maximum 100%.

5.6. The initial guesses for the coefficients are determined by using the DIRECT global optimization algorithm. The algorithm changes only the balance points and the rest of the model is set using a 3 segment, non-smoothed piecewise linear function as described previously. The algorithm seeks to minimize the sum of squared error (SSE).

5.7. The initial guesses are used as inputs into the Subplex local optimization algorithm to minimize a Lasso regression-inspired objective function. The objective function is a function of the residuals from the model function and the observed values as well as the coefficient values.

5.7.1. Each coefficient in the model has its own penalty factor that is combined by taking the sum of the absolute values of these penalty factors multiplied by 0.001.

5.7.2. The heating and cooling balance points are penalized by taking the minimum distance between them and the minimum and maximum observed temperature, respectively. An additional penalty is added to each of them that is half the distance between the two balance points.

5.7.2.1. The balance point penalties serve to push the balance points towards each other and to the temperature extremes.

5.7.3. The slopes are penalized by first scaling them by the standard deviation of the temperature divided by the standard deviation of observed meter values from within the region being modeled. They are multiplied by an additional penalty (1E30) if the number of data points is less than minimum required.

5.7.3.1. The slope penalties push the slopes towards zero, heavily in the case of not meeting the minimum number of data points requirement.

5.7.4. The smoothing parameters, computed as 0 - 100%, are normalized in the same manner as 5.5.1 and then multiplied by their associated slopes divided by 2.

5.7.4.1. The effect of the smoothing parameters is to push them towards zero, but more heavily if slopes they are smoothing are larger.

6. Final Submodel Fitting

The final model uses the inputs from the preliminary models to minimize the weighted SSE (wSSE) between the observed values and the model. This utilizes the adaptive loss function and does not use the Lasso regression-inspired penalty.

6.1. Bounds are set so the balance points cannot create distinct modeling regions that contain less than the minimum number of designated points.

6.2. The adaptive loss is a continuous function of a shape parameter, α, and median-standardized residuals.

6.2.1. The shape parameter can make the function replicate a SSE ($\alpha = 2$), smoothed L1 ($\alpha = 1$), Cauchy ($\alpha = 0$), or Welsh loss ($\alpha = \infty$) depending on its value. It is determined each time the function is called through a

single variable optimization and is penalized to prefer SSE, it will choose more outlier resilient values if appropriate.

6.2.2. The median standardized residuals are unique for each section of the function (heating, temperature-independent, and cooling). The raw residuals undergo a simple outlier rejection scheme using the standard 1.5 Inner Quartile Range (IQR) rule. Their medians are then subtracted from and they undergo another 1.5 IQR rule to estimate the locations of outliers among the shifted residuals. The shifted residuals are divided by this value to get the median-standardized residuals.

6.2.3. The adaptive loss function is converted to weights based on the optimum α.

6.3. Each squared residual is multiplied by its weight and then summed to obtain the net wSSE.

7. Use with Monthly Data

When monthly consumption data are the subject of the calculation, there are not enough data points to support model splitting as described above.

7.1. With monthly data, OpenEEmeter 4.0 should be used in legacy mode.