# Calculating Normalized Savings using nmecr

Mrinalini Sharma* David Jump† Devan Johnson‡

August 7, 2020

The `nmecr` package was written to integrate the past efforts of the energy efficiency industry to streamline meter-based Measurement & Verification of EE projects. It brings together simple linear regression with outside air temperature, the change point models (from ASHRAE 1050-RP), and the Time-of-Week and Temperature model (developed by the Lawrence Berkeley National Laboratory) and builds enhancements upon these by accurately predicting the energy use profiles of facilities with multiple operating modes.

## Normalized Metered Energy Consumption (NMEC)

Meter-based energy efficiency programs are proliferating across the globe. This proliferation is influenced by the widespread availability of high frequency energy use measurements from new metering technology as well as advancements in statistical regression and other empirical energy data modeling methodologies. Program administrators may report savings as the overall reduction in normalized metered energy consumption (NMEC). This method to determine savings is based on analysis of meter data collected prior to and after energy efficiency and conservation measures have been installed. Referred to as advanced measurement and verification (M&V) by the industry, this time-granular data and updated modeling methods provide several advantages over other methods used to quantify the benefits of energy efficiency:

- It reliably determines the actual savings achieved at the meter
- It provides fast feedback on the facility's energy performance and savings progress
- It enables identification and troubleshooting of issues that prevent savings realization

The `nmecr` package streamlines the application of the NMEC approach by enabling the:

- Management of high frequency, high volume data.
- Execution of advanced, state-of-the-art time-series data modeling algorithms.
- Comprehensive assessment of the validity of energy data models in specific applications.
- Quantification of uncertainties and risks associated with the energy savings projections in accordance with ASHRAE Guideline 14.

For an overview of nmcer, please refer nmecr_overview.pdf

*msharma@kw-engineering.com
†djump@kw-engineering.com
‡johnson@kw-engineering.com

## Data

`nmecr` includes two datasets for this vignette: `temp` and `eload`. These contain three years (03/01/2012 - 02/28/2015) of energy use and outside temperature data from a commercial building in North America. We will use the first year (03/01/2012 - 02/28/2013) of energy use and temperature data for this predictability analysis, as at the start of any project, we generally can collect a year of data. Additionally, a TMY3 dataset is provided as `TMY3`.

```
# Load data into an R session:

data(eload)
data(temp)
data(TMY3)
```

`eload` and `temp` are data frames with the two variables each. When using `nmecr` functions, ensure that the column headers of your datasets are the same as those shown below.

Eload:

```
#> # A tibble: 5 x 2
#>   time                 eload
#>   <dttm>               <dbl>
#> 1 2012-03-01 00:00:00 21505.
#> 2 2012-03-02 00:00:00 20892.
#> 3 2012-03-03 00:00:00 20435.
#> 4 2012-03-04 00:00:00 15660.
#> 5 2012-03-05 00:00:00 15864.
```

Temp:

```
#> # A tibble: 5 x 2
#>   time                 temp
#>   <dttm>              <dbl>
#> 1 2012-03-01 00:00:00  38.4
#> 2 2012-03-02 00:00:00  39.9
#> 3 2012-03-03 00:00:00  43.0
#> 4 2012-03-04 00:00:00  49.7
#> 5 2012-03-05 00:00:00  48.3
```

## Baseline Dataframe for Modeling

`create_dataframe()` combines the `eload` and `temp` dataframes into one, filters by the specified start and end dates, and aggregates to an hourly, daily, or a monthly data interval. It lines up all data such that each timestamp represents attributes up until that point, e.g. an eload value corresponding to 03/02/2012 represents the energy consumption up until then - the energy consumption of 03/01/2012.. The `timestamps` parameter is optional, with permissible values: 'start' and 'end', indicating whether the timestamps mark the start or the end of the period. If operating mode data is supplied, this information is added to the dataframe created by `create_dataframe()`.

```
# Baseline Dataframe

baseline_df <- create_dataframe(eload_data = eload, temp_data = temp,
```

```
                                start_date = "03/01/2012 00:00",
                                end_date = "02/28/2013 23:59",
                                convert_to_data_interval = "Daily",
                                timestamps = "start")

head(baseline_df, 5)
#> # A tibble: 5 x 5
#>   time                eload  temp   HDD   CDD
#>   <dttm>              <dbl> <dbl> <dbl> <dbl>
#> 1 2012-03-02 00:00:00 21505.  38.4  26.6     0
#> 2 2012-03-03 00:00:00 20892.  39.9  25.1     0
#> 3 2012-03-04 00:00:00 20435.  43.0  22.0     0
#> 4 2012-03-05 00:00:00 15660.  49.7  15.3     0
#> 5 2012-03-06 00:00:00 15864.  48.3  16.7     0
```

```
# Performance Period Dataframe

performance_df <- create_dataframe(eload_data = eload, temp_data = temp,
                                   start_date = "03/01/2014 00:00",
                                   end_date = "02/28/2015 23:59",
                                   convert_to_data_interval = "Daily",
                                   timestamps = "start")

head(performance_df, 5)
#> # A tibble: 5 x 5
#>   time                eload  temp   HDD   CDD
#>   <dttm>              <dbl> <dbl> <dbl> <dbl>
#> 1 2014-03-02 00:00:00 15989.  50.0  15.0     0
#> 2 2014-03-03 00:00:00 16033.  43.5  21.5     0
#> 3 2014-03-04 00:00:00 16938   41.8  23.2     0
#> 4 2014-03-05 00:00:00 17325.  51.8  13.2     0
#> 5 2014-03-06 00:00:00 17408.  51.0  14.0     0
```

## Energy Data Modeling

The baseline period dataframe can be used for energy data modeling using one of the four modeling algorithms available in nmecr:

- `model_with_TOWT()*`: Time-of-Week & Temperature and Time-Only algorithms
- `model_with_CP()`: 3-Parameter Heating, 3-Parameter Cooling, 4-Parameter, and 5-Parameter algorithms
- `model_with_SLR()`: Simple Linear Regression algorithm
- `model_with_HDD_CDD()`: Heating Degree Day only, Cooling Degree Day only, and a combination of Heating Degree Day and Cooling Degree Day algorithms

The common arguments for these four algorithms are:

1. training_list (output from `create_dataframe()`)
2. model_input_options (see below)

`*model_with_TOWT()` has an additional argument (prediction_list) for creating data models and predictions together.

### model_input_options()

The four modeling algorithms have many specifications in common that can be specified using the `assign_model_inputs()` function. The following code chunk shows the default values for these model inputs

```
model_input_options <- assign_model_inputs(timescale_days = NULL,
                                           has_temp_knots_defined = FALSE,
                                           equal_temp_segment_points = TRUE,
                                           temp_segments_numeric = 6,
                                           temp_knots_value = c(40, 45, 50, 60, 65, 90),
                                           initial_breakpoints = c(50,65),
                                           regression_type = "TOWT",
                                           occupancy_threshold = 0.65)
```

As noted in the nmecr_predictability.pdf vignette, the Time-of-Week and Temperature model is best suited for this facility.

**Time of Week and Temperature:**

```
TOWT_baseline_model <- model_with_TOWT(training_data = baseline_df,
                                model_input_options =
                                  assign_model_inputs(regression_type = "TOWT"))
```


Figure 1: Baseline Energy Use modeled with Time-of-week and Temperature over Time

```
TOWT_performance_model <- model_with_TOWT(training_data = performance_df,
                                model_input_options =
                                  assign_model_inputs(regression_type = "TOWT"))
```

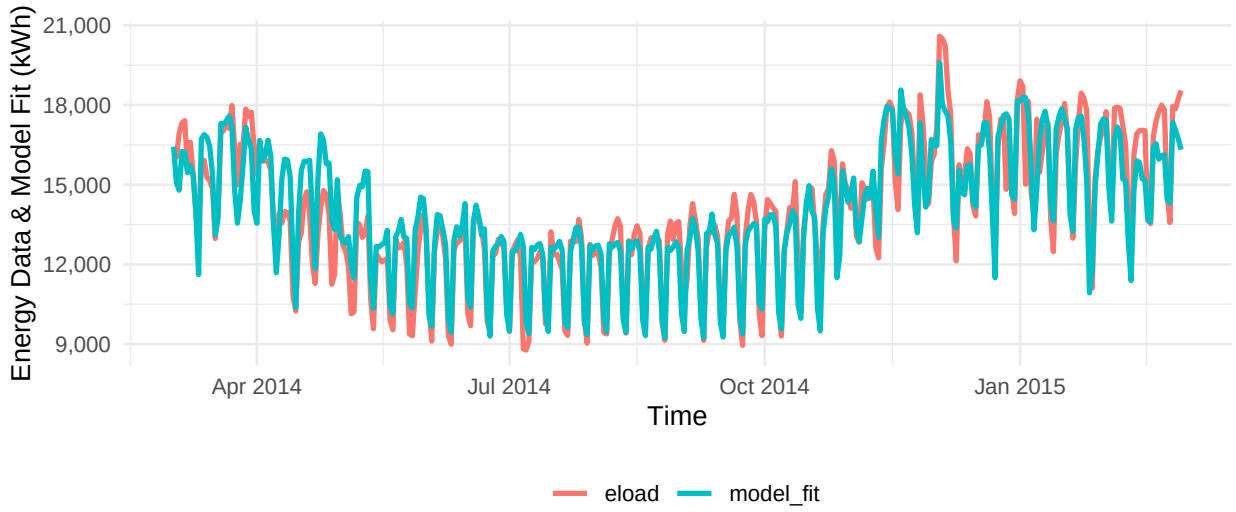


Figure 2: Performance Period Energy Use modeled with Time-of-week and Temperature over Time

**Model Statistics**

To ensure that the models meet goodness-of-fit criteria, we can run the `calculate_summary_statistics()` function:

```
TOWT_baseline_stats <- calculate_summary_statistics(TOWT_baseline_model)

TOWT_performance_stats <- calculate_summary_statistics(TOWT_performance_model)

all_stats <- bind_rows(TOWT_baseline_stats, TOWT_performance_stats)

model_names <- c("TOWT Baseline", "TOWT Performance")

all_stats <- bind_cols("Model Name" = model_names, all_stats)

all_stats
#>          Model Name R_squared CVRMSE %        NDBE %        NMBE % #Parameters
#> 1     TOWT Baseline      0.90     6.79 -7.521974e-14 -7.936227e-14          19
#> 2 TOWT Performance      0.85     7.24  4.407054e-14  4.649761e-14          19
```

- CVRMSE: Coefficient of Variation of Root Mean Squared Error
- NDBE: Net Determination Bias Error
- MBE: Mean Bias Error
- R_squared: Coefficient of Determination

## Assessing project fit for NMEC

Using the models' statistics and the uncertainity for 10% savings, we can determine the validity of the NMEC approach for a certain project. For the building decribed here, following are the key values to be used for this assessment

1. CVRMSE (should be $< 25\%$):

- Baseline Model: 6.8%

- Performance Period Model: 7.2%

2. NDBE: should be $< 0.005\%$

- Baseline Model: 0.0000
- Performance Period Model: 0.000

The California Public Utilities Commission requires savings to be detectable above model variations. `nmecr` interprets this using ASHRAE Guideline 14 - 2002's formulation for savings uncertainty, which relates the savings uncertainty to the model goodness of fit metric CV(RMSE), the confidence level, the amount of savings, the amount of data used to develop the model, and the amount of data required to report savings. The formulation includes the correction polynomial developed by Sun and Balthazar[1] for daily and monthly models. It also includes a correction when autocorrelation is present (which occurs mainly in models developed from daily and hourly data). LBNL has shown this uncertainty formulation with correction for autocorrelation underestimates the savings uncertainty. More work on this issue is needed. Until a better formulation is available, `nmecr` uses ASHRAE's method only as an estimation.

3. Savings Uncertainty for 10% savings (at 90% confidence level) (should be $< 50\%$):

```
TOWT_baseline_savings_10 <- calculate_savings_and_uncertainty(prediction_df = NULL,
                            savings_fraction = 0.1,
                            modeled_object = TOWT_baseline_model,
                            model_summary_statistics = TOWT_baseline_stats,
                            confidence_level = 90)

TOWT_baseline_savings_10$savings_summary_df
#>   savings_fraction savings_uncertainty savings_frac_for_50pct_uncertainty
#> 1              0.1           0.1674086                          0.03348172
#>   confidence_level
#> 1               90
```

- Baseline Model: 16.7%

```
TOWT_performance_savings_10 <- calculate_savings_and_uncertainty(prediction_df = NULL,
                               savings_fraction = 0.1,
                               modeled_object = TOWT_performance_model,
                               model_summary_statistics = TOWT_performance_stats,
                               confidence_level = 90)

TOWT_performance_savings_10$savings_summary_df
#>   savings_fraction savings_uncertainty savings_frac_for_50pct_uncertainty
#> 1              0.1               0.185                          0.03700001
#>   confidence_level
#> 1               90
```

- Performance Period Model: 18.5%

In addition to evaluating the model metrics, it is essential to ensure that the modeled profile follows the actual energy use closely (see Figure 8 above), and that the four assumptions of linear regression are met by the model. For a step-by-step process of evaluating the model against the four assumptions of regression, read nmecr_predictability.pdf

[1] Sun, Y. and Baltazar, J.C., "Analysis and Improvement of the Estimation of Building Energy Savings Uncertainty," ASHRAE Transactions, vol. 119, May 2013

## Normalized Savings and Uncertainty

Normalized Savings represent the energy savings that measures would achieve under a 'normal' set of conditions which, in this case, TMY3 weather data.

Inspecting the TMY3 data, we see that it has hourly data intervals:

```
head(TMY3, n = 5)
#> # A tibble: 5 x 2
#>   time                 temp
#>   <dttm>              <dbl>
#> 1 1987-01-01 01:00:00  41
#> 2 1987-01-01 02:00:00  41
#> 3 1987-01-01 03:00:00  41
#> 4 1987-01-01 04:00:00  42.1
#> 5 1987-01-01 05:00:00  43.0
```

Since we have our models in the daily time intervals, it will be best to aggregate the TMY3 data to the daily level as well:

```
TMY3_daily <- aggregate(eload_data = NULL, temp_data = TMY3,
                        convert_to_data_interval = "Daily", temp_balancepoint = 65)
```

Next, we calculate the baseline and the performance period energy use normalized to TMY3 condotions.

```
TOWT_baseline_normalized <- calculate_model_predictions(training_data = baseline_df,
                            prediction_data = TMY3_daily,
                            modeled_object = TOWT_baseline_model)

TOWT_performance_normalized <- calculate_model_predictions(training_data = performance_df,
                               prediction_data = TMY3_daily,
                               modeled_object = TOWT_performance_model)
```

The normalized energy savings are calculated as:

```
TOWT_normalized_savings <- data.frame("Savings" =
                                        TOWT_baseline_normalized$predictions -
                                        TOWT_performance_normalized$predictions)

TOWT_normalized_savings_fraction <- sum(TOWT_normalized_savings, na.rm = T) /
                                    sum(TOWT_baseline_normalized$predictions, na.rm = T)
```

```
#>   Normalized Savings Normalized Savings Fraction
#> 1          540499.2                   0.0905966
```

Normalized savings uncertainty is calculated as the square-root of the sum of the baseline and performance period models' squared uncertainties. These two values can be calculated using the `calculate_savings_and_uncertainty()` function:

```
TOWT_baseline_normalized_uncertainty <- calculate_savings_and_uncertainty(prediction_df = NULL,
                                        savings_fraction = TOWT_normalized_savings_fraction,
                                        modeled_object = TOWT_baseline_model,
                                        model_summary_statistics = TOWT_baseline_stats,
                                        confidence_level = 90)

TOWT_performance_normalized_uncertainty <- calculate_savings_and_uncertainty(prediction_df = NULL,
                                           savings_fraction = TOWT_normalized_savings_fraction,
                                           modeled_object = TOWT_performance_model,
                                           model_summary_statistics = TOWT_performance_stats,
                                           confidence_level = 90)


TOWT_baseline_normalized_uncertainty_value <-

TOWT_baseline_normalized_uncertainty$savings_summary_df$savings_uncertainty

TOWT_performance_normalized_uncertainty_value <-

TOWT_performance_normalized_uncertainty$savings_summary_df$savings_uncertainty

TOWT_normalized_savings_uncertainty <- sqrt(TOWT_baseline_normalized_uncertainty_value^2 +
                                       TOWT_performance_normalized_uncertainty_value^2)

TOWT_normalized_savings_uncertainty
#> [1] 0.2753975
```

The Normalized Savings are found to be 9.1% with an associated uncertainty of 27.5%.