

Introduction to nmecr

*Mrinalini Sharma**

David Jump[†]

Devan Johnson[‡]

November 22, 2019

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 ASHARE 1040-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.

Data

nmecr includes two datasets: **temp** and **eload**. These contain three years (03/01/2012 - 02/28/2015) of outside air temperature data and energy use data from a commercial building in North America. Energy conservation measures were installed between 03/01/2013 and 02/28/2014 in this building and so, we will use the first year (03/01/2012 - 02/01/2013) as the pre-implementation dataset and the third year (03/01/2014 - 02/28/2015) as the post-implementation dataset.

*msharma@kw-engineering.com

[†]djump@kw-engineering.com

[‡]johnson@kw-engineering.com

```
# Load data into an R session:
```

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

`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  
#>   <dtm>              <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  
#>   <dtm>              <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 and Performance Period Dataframes 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. If an operating mode dataframe is supplied, `create_dataframe` creates a list with two objects: dataframe, `operating_mode_data`.

```
# 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")
```

```
head(baseline_df$dataframe, 5)  
#> # A tibble: 5 x 5  
#>   time                temp  HDD  CDD  eload  
#>   <dtm>              <dbl> <dbl> <dbl> <dbl>  
#> 1 2012-03-01 00:00:00  38.4  26.6    0 21505.  
#> 2 2012-03-02 00:00:00  39.9  25.1    0 20892.  
#> 3 2012-03-03 00:00:00  43.0  22.0    0 20435.  
#> 4 2012-03-04 00:00:00  49.7  15.3    0 15660.  
#> 5 2012-03-05 00:00:00  48.3  16.7    0 15864.
```

```

# Performance Period Dataframe

performance_df <- create_dataframe(eload_data = eload, temp_data = temp,
                                   start_date = "03/01/2013 00:00",
                                   end_date = "02/28/2014 23:59",
                                   convert_to_data_interval = "Daily")

head(performance_df$dataframe, 5)
#> # A tibble: 5 x 5
#>   time                temp  HDD  CDD  eload
#>   <dtm>              <dbl> <dbl> <dbl> <dbl>
#> 1 2013-03-01 00:00:00  49.3 15.7    0 19858.
#> 2 2013-03-02 00:00:00  55.8  9.18   0 19467.
#> 3 2013-03-03 00:00:00  50.6 14.4    0 17129.
#> 4 2013-03-04 00:00:00  42.9 22.1    0 17736.
#> 5 2013-03-05 00:00:00  44.6 20.4    0 19585.

```

Energy Data Modeling

The baseline and performance period dataframes can then 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

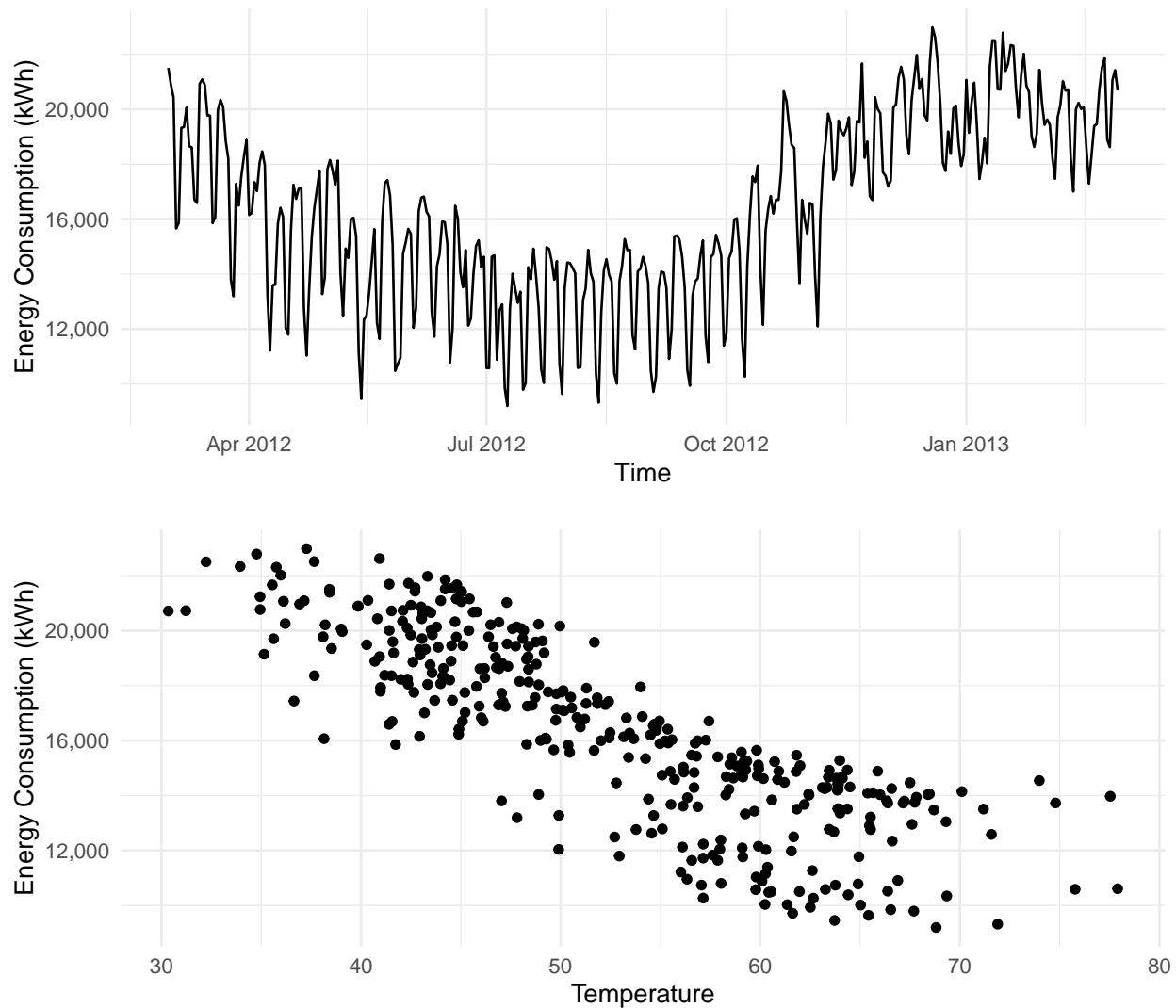
```

model_input_options <- assign_model_inputs(timescale_days = NULL, interval_minutes = 15,
                                           has_temp_knots_defined = FALSE,
                                           equal_temp_segment_points = TRUE,
                                           temp_segments_numeric = 6,
                                           temp_knots_value = c(40, 45, 50, 60, 65, 70),
                                           initial_breakpoints = c(50, 65),
                                           regression_type = "TOWT")

```

Before creating data models, it is a good practice to visualize the energy use profiles:

Baseline Energy Use

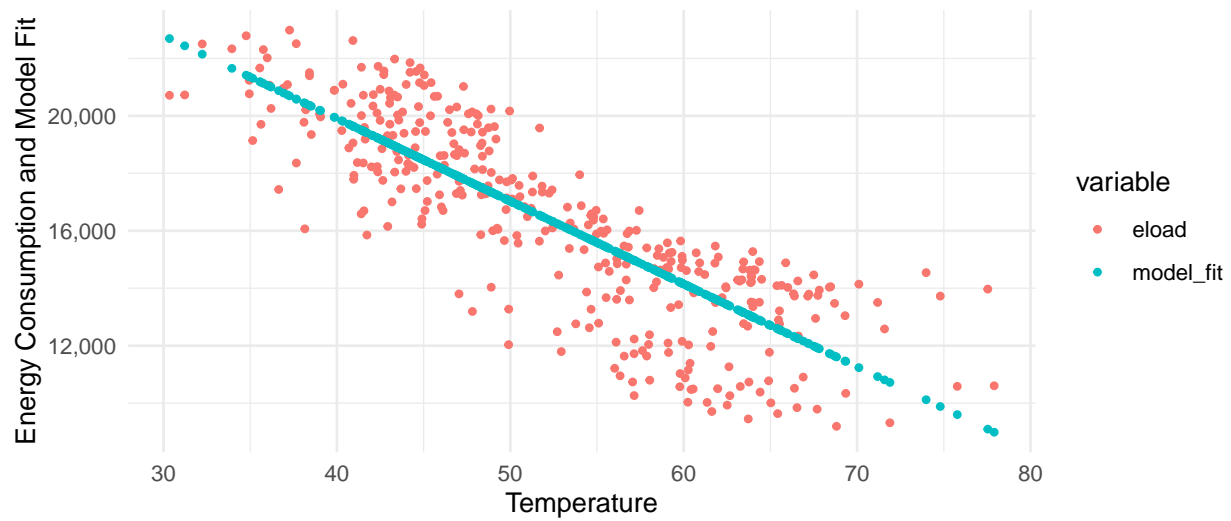
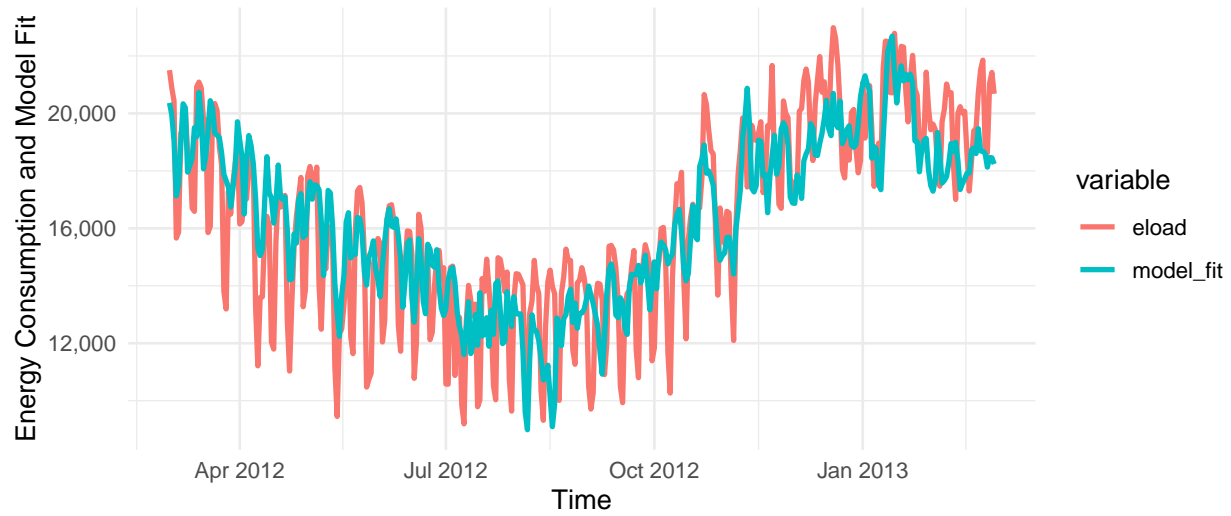


Due to the high dependence on temperature, we can begin the data modeling using Simple Linear Regression and compare that to other modeling algorithms

Model Creation

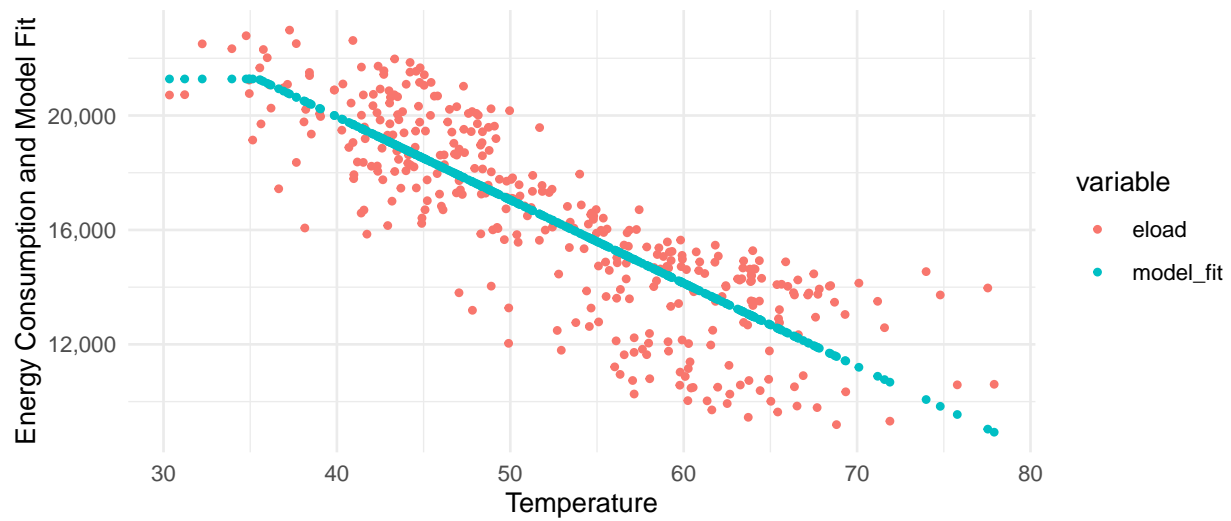
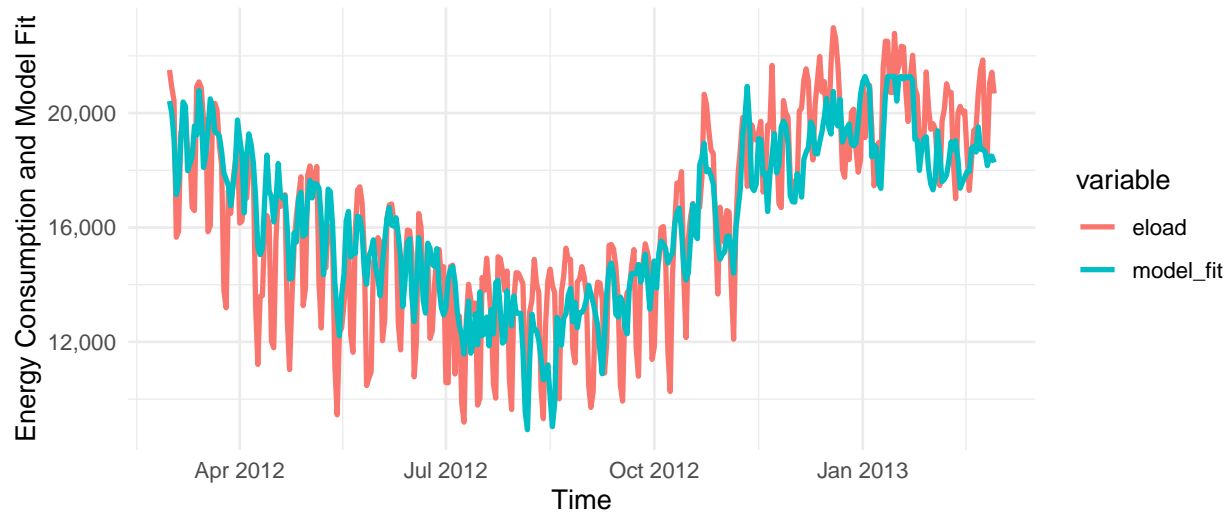
Simple Linear Regression:

```
SLR_model <- model_with_SLR(training_list = baseline_df,  
                             model_input_options =  
                               assign_model_inputs(regression_type = "SLR"))
```



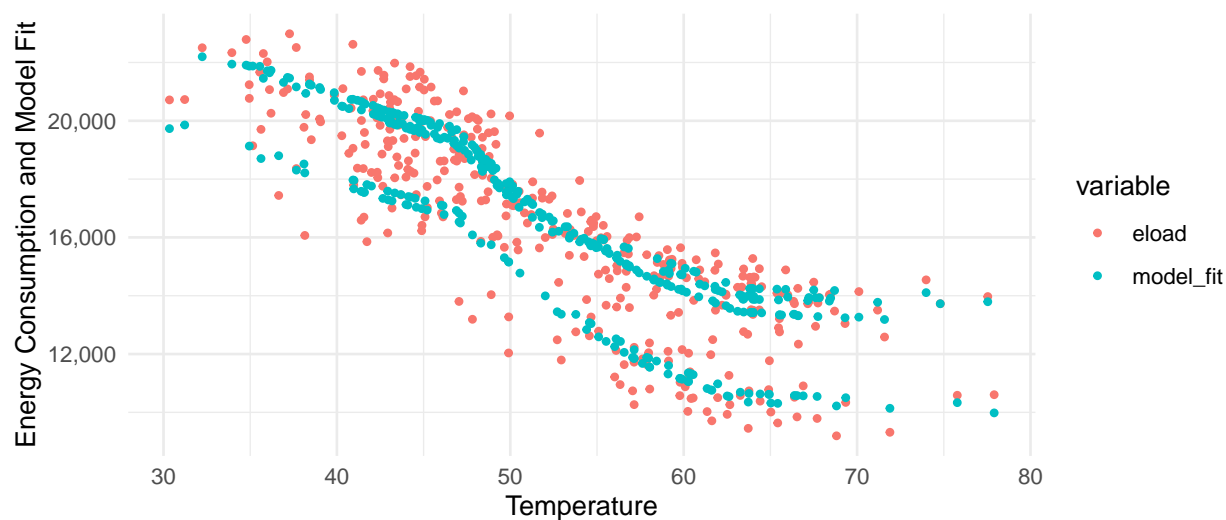
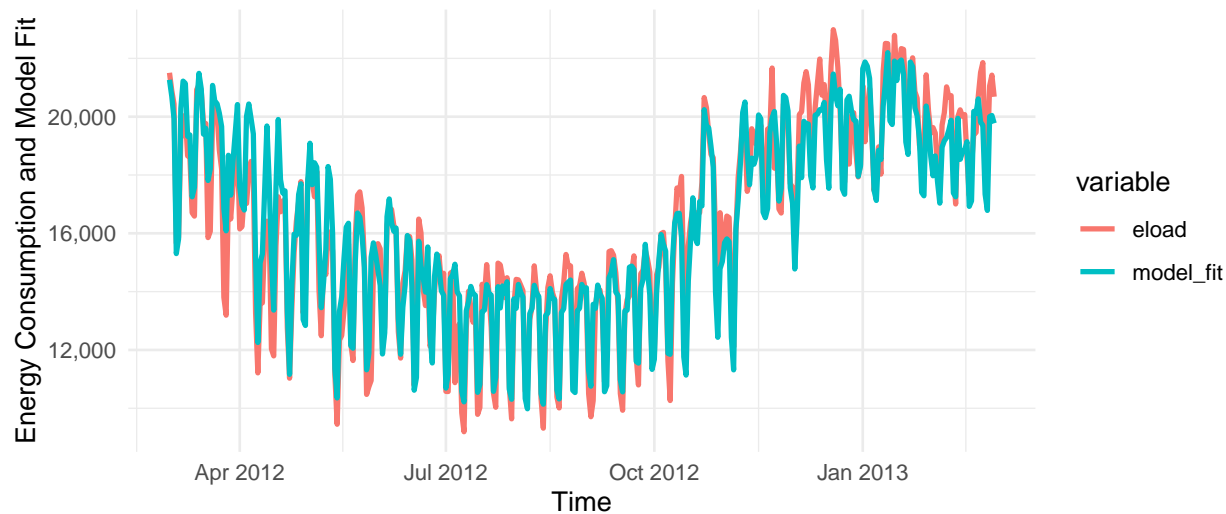
Three Parameter Heating:

```
Three_PH_model <- model_with_CP(training_list = baseline_df,
                                model_input_options =
                                  assign_model_inputs(regression_type =
                                                        "Three Parameter Heating"))
```



Time of Week and Temperature:

```
TOWT_model <- model_with_TOWT(training_list = baseline_df,
                               model_input_options =
                                 assign_model_inputs(regression_type = "TOWT"))
```



The plots show that the Time-of-Week and Temperature algorithm models the energy use profile better than the other two.

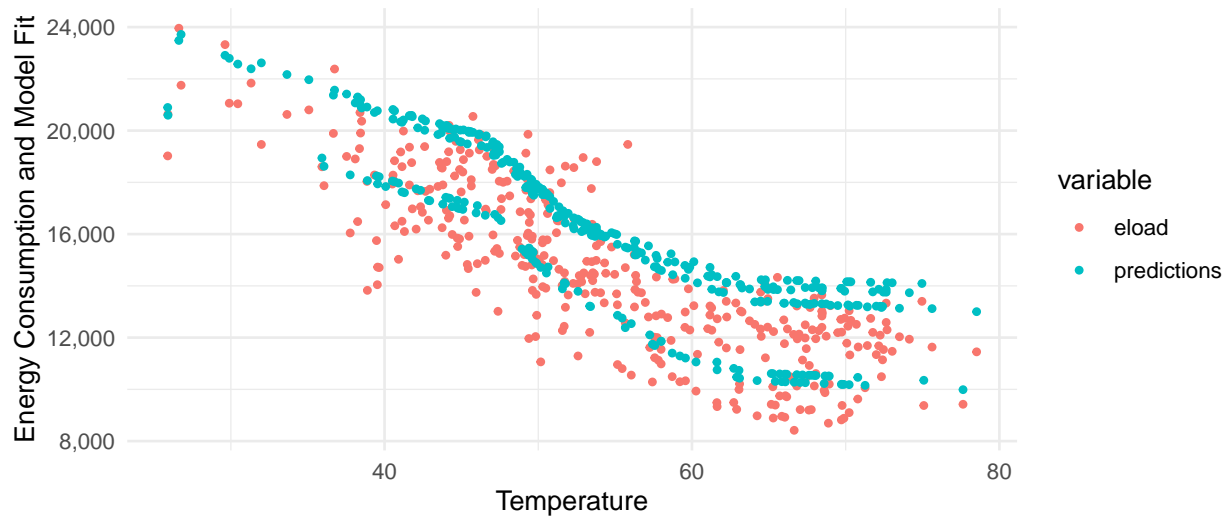
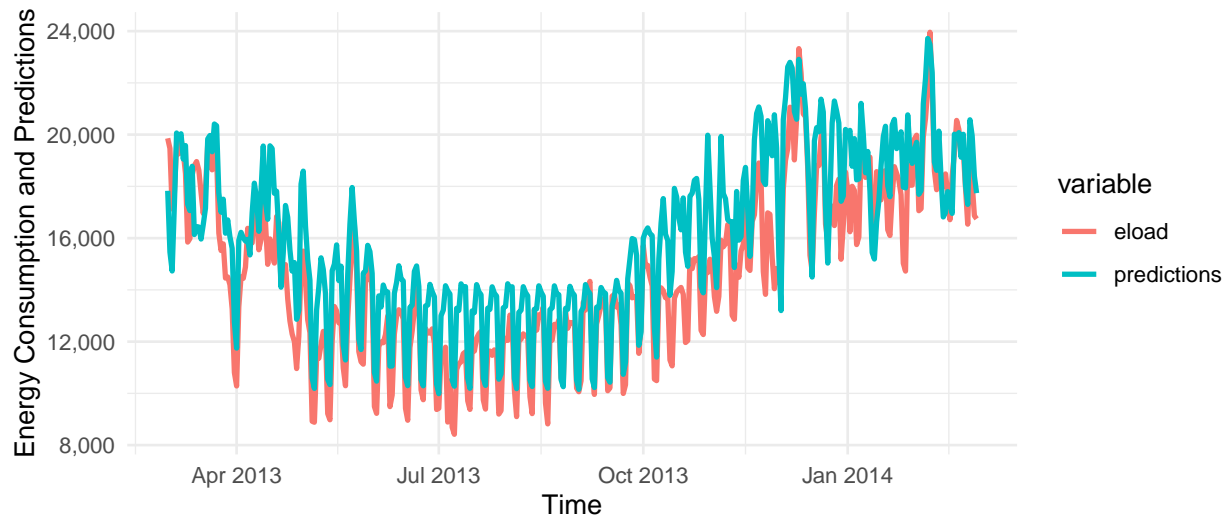
This insight can be confirmed through model statistics by using the `calculate_summary_statistics()` function. The following table summarizes the results from this function for each of the models assessed above:

```
#> # A tibble: 3 x 6
#>   `Model Name`      R_squared CVRMSE      NDBE      MBE `#Parameters`
#>   <chr>          <dbl>  <dbl>    <dbl>    <dbl>      <int>
#> 1 SLR           0.692  0.115  -2.20e-17 -3.59e-13      2
#> 2 Three_Parameter_Heat~ 0.693  0.115   1.64e-16  2.67e-12      3
#> 3 TOWT          0.897  0.0678 -7.95e-16 -1.30e-11     19
```

Energy Load Prediction

The energy use models can be used to predict energy use during a different time period. This functionality is central to calculating the Adjusted Baseline Energy Use, which can be further utilized to calculate Avoided Energy Use (or Energy Savings)

```
TOWT_predictions <- calculate_model_predictions(training_list = baseline_df,
                                              prediction_list = performance_df,
                                              modeled_object = TOWT_model)
```



The California Public Utilities Commission requires savings to be detectable above model variations. **nmecr** interprets this using ASHRAE Guideline 14 - 2014'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. It 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.

```
TOWT_savings <- calculate_savings_and_uncertainty(prediction_results_list =
                                              TOWT_predictions,
                                              modeled_object = TOWT_model,
                                              model_summary_statistics = TOWT_stats,
                                              confidence_level = 90)
```



```
TOWT_savings$savings_summary_df
#>   performance_period_use adjusted_baseline_use savings savings_pct
#> 1           5336163           5829068 492904.4           0.08
#>   savings_uncertainty savings_pct_for_50pct_uncertainty confidence_level
#> 1           0.1312905           0.02100648           90
```

Savings achieved at the meter for this project amount to ~8% of the Adjusted Baseline Use with an associated savings uncertainty of 13%. ASHRAE requires that the savings uncertainty for a project be below 50% at the 90% confidence level. The 13% uncertainty is well below this threshold.

The savings percentage required to meet this threshold would be 2.1% as shown by the output above (see: savings_pct_for_50pct_uncertainty).