IE590 Final Report

Brian Baller

4/18/2019

Honor Code

I have obeyed all rules for this exam and have not received any unauthorized aid or advice Signed!

Overview

My basic approach to this exam was the same as the midterm – to run all the applicable models that we have used in class and evaluate their performance. For my analysis, I've chosen *Electricity cooling use (thous Btu)* (*ELCLBTU*) as the response variable, per the email from Dr. Nateghi.

My objective:

The local government in the "Pacific" region of the United States would like you to leverage this data and report to them if you can developed a predictive model of the total electricity used for cooling (quantitative response) and identify the main predictors.

With *ELCLBTU* as the response, the problem becomes a regression problem so I've only chosen those models we've used that are suitable for regression. They are:

- Linear Regression
- Ridge/Lasso Regression
- GAM
- Rpart
- Random Forest/Bagging
- Boosting
- Earth
- BART
- SVM Regression

Data Import and Cleaning

The data file has two quite noteworthy characteristics – more columns than rows and lots of missing values. My EDA dealt with both and is commented in the attached R Markdown File. Several key points on the EDA:

- My assigned area is "Pacific", so I filtered on just "Pacific". It's possible that using training data from other regions would help our Pacific predicitions, but the Pacific climate is milder than many of the other US regions, so I didn't include the other regions in the training data.

- The data included a "Yes/No" question on "Is electricity used for cooling?" About 12% of the buildings responded in the negative. After checking to ensure these building had no electricity usage for cooling, I deleted them from the dataframe.

- The response, *ELCLBTU* is not normally distributed, but closer to exponentially distributed. The log transform was used to bring it in line with a normal distribution.

- One outlier (PUBID 5503) was noted in the response data – one building had usage 25 standard deviations above the mean, while having a square footage only 6 SDs above the mean. This outlier will dominate the results if left in. For instance, if the outlier is in the test data and the training data predict the Y value to be equal to the second highest Y value, then the error will be 87M or 100 times the median value. In comparison, if the second highest Y value is in the test set and the training data predicts it to have the same Y value as the third highest Y value, then the error is 6M or 7 times the median value. I chose to delete this value, but it should be investigated to find out why it's so high. If it is a true reading, the cause should be identified and either corrected or, if possible, a predictor added to explain the variation.
- I removed all the **Z** columns and **FINALWT** columns, as they have no predictive value.
- After removing those columns, only 104 columns remained that were without NAs. As noted in the User's Guide, EIA already tried to impute as many values as they could. They recommend (p. 16) using the imputed data, which I did. I also decided to not impute any more data, as (a) it would be error-prone and (b) EIA has already done what they can. Thus, I deleted all the columns with NAs. I think we lost some possible valuable data ('percent exterior glass' is one example).

- I also deleted 12 very low-variance columns.

- There are 28 energy usage and expenditure columns. I deleted these as (a) I feel like including them to predict electrical cooling usage is cheating and (b) if we are trying to predict electrical usage using other usage data, why didn't we just capture the electrical cooling usage data as well?

- With the usage columns removed, the correlations with *ELCLBTU* are not as strong. The highest correlation with *ELCLBTU* is 0.66. Several predictors were highly correlated with each other (> 0.8) and six of these were removed. Several variables were represented twice (such as "Year Constructed" and "Year Constructed Category"). I elected to keep the most useful one and deleted its twin.
- Several of the "integer" columns included a "category" or "level" to capture the "greater than" information. An example is "995" in "Number of Floors" corresponds to greater

than 25 floors. Because 995 is many times larger than 1-25, this variable shouldn't be treated as an integer. Yet, using it as a factor loses valuable information as *ELCLBTU* is likely linearly related to the number of floors. To try and split the difference, I recoded these "995" numbers to more appropriate values and coded these variables as integers. As confirmation of this technique, step.GAM selected *FLCEILHT* as a linear variable – had I kept the "995" it would have had to fit a curve with a sharp bend.

Note that I've removed the above columns from my dataframe, so my models DO NOT take the whole data file as an input. I've also deleted one big outlier – PUBID 5503. If the models are run with PUBID 5503, the RMSEs will be many times higher.

Models

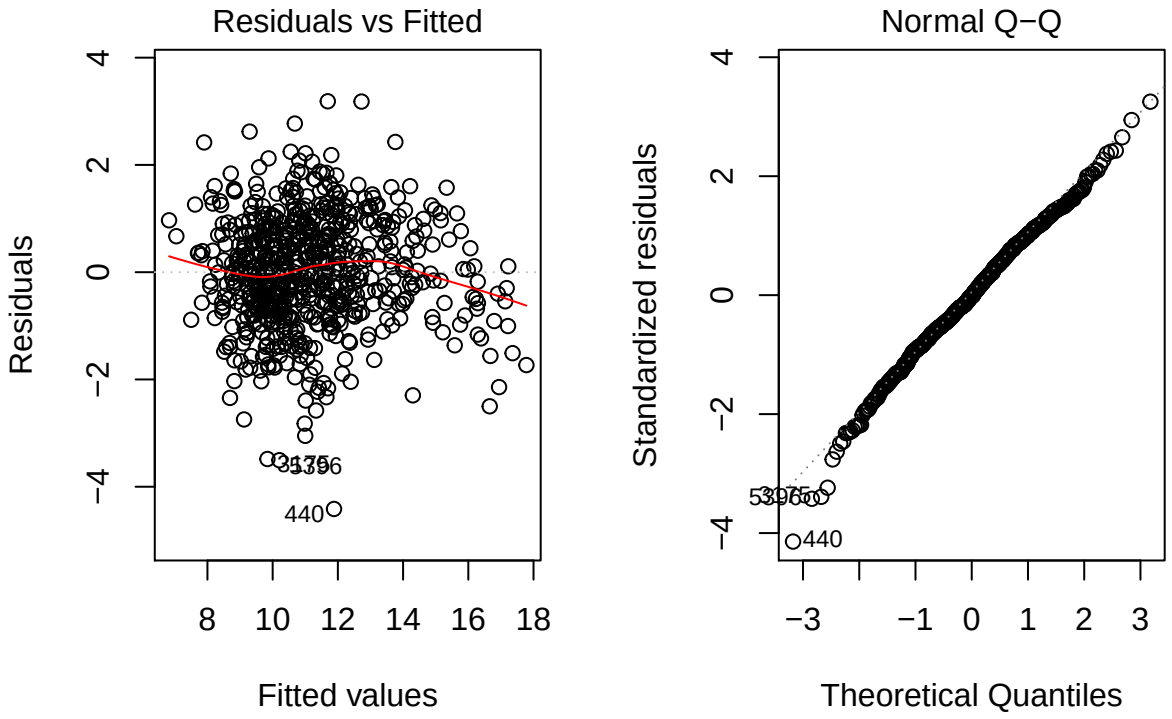
All analysis was done using a training / test set split of 70/30. For the linear models, I one-hot encoded the factor variables. All other models use the non-one-hot encoded dataframe. To pick parameters for model, I did K-fold CV within the training set. The test set was only used to evaluate the final model for each model type. The MARS model proved the best of all models and is my final model. The table of results:

```
##           Training RMSE Test RMSE
## err.earth       1406333   1134383
## err.svr         1179710   1479023
## err.bart        1251918   1493189
## err.boost       1336240   1861454
## err.gam         2015878   2033110
## err.rf          1158706   2149805
## err.lasso       1999294   2379212
## err.rpart       2090896   2480571
## err.lm          2887626   4910952
```

A few notes on the models used:

Linear Models

The linear regression model had an adjusted R^2 of 78%. A LASSO model was also fitted – it has a reduced test RMSE and reduces the number of predictors. The model assumption were also checked and they look good – the residuals show constant variance and have a close to normal Q-Q Plot.



GAM

A GAM model was run with the available predictors. Because GAM requires numerics or ordered factors, the number of variables available to use is smaller than the other models. There are really only 8 integer columns, although I converted several factors to integers by recoding the "995" levels to a appropriate numeric (see above explanation). With the extra variables, GAM was able to perfrom well, although not the best.

SVR

The SVR model was tuned using the *tune()* over a range of ϵ and cost. The best model had $\epsilon = 0$ and a cost of 36.

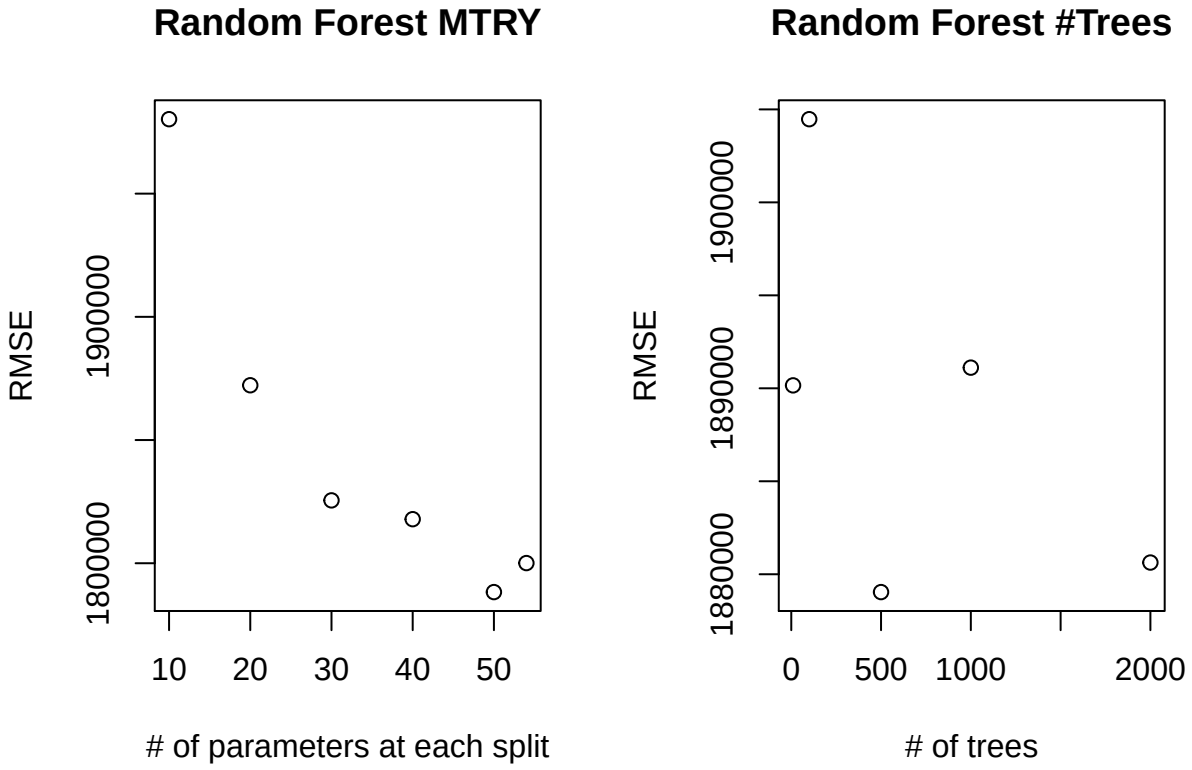
Rpart

An rpart model was fit and optimized using *cp*. The optimal model had eleven terminal nodes.

Random Forest

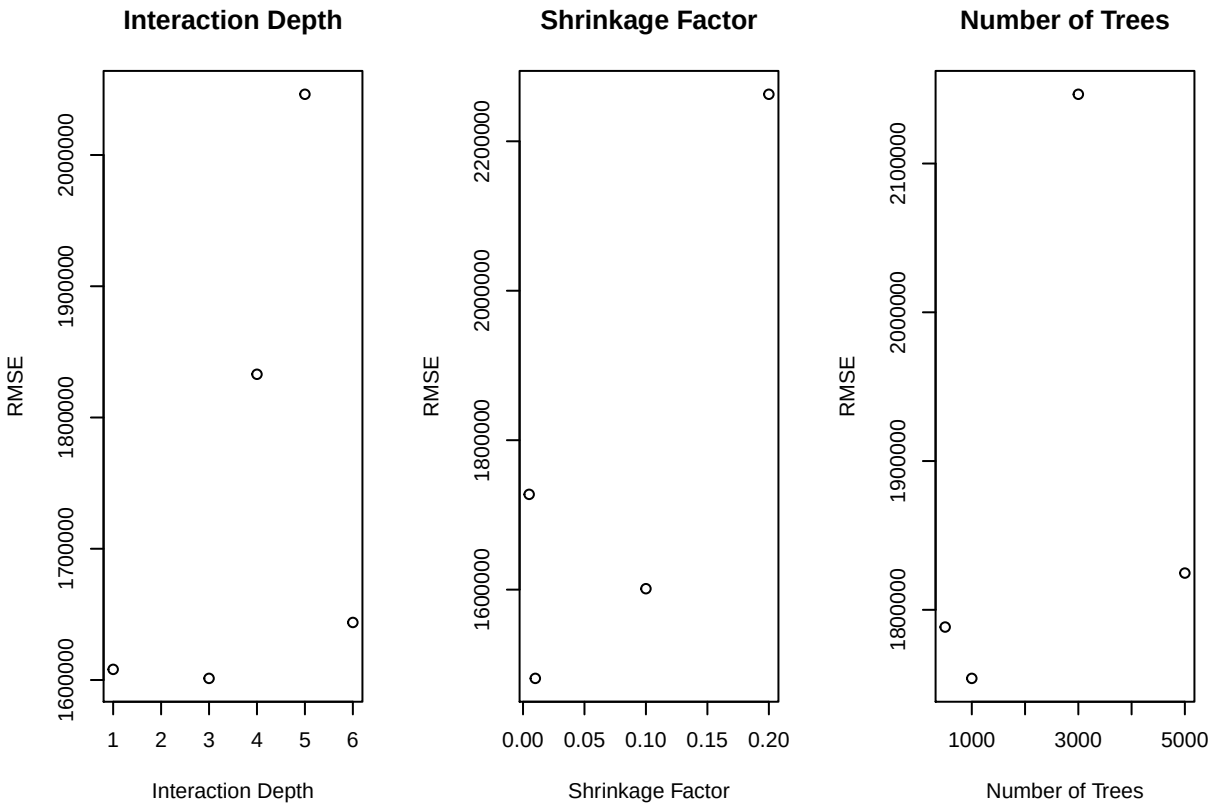
The random forest model was optimized by altering the number of parameters considered at each decision from 10 to all the parameters (bagging) and changing the number of trees. A

5-fold cross-validation was performed using only the training data set. The optimal value of *mtry* was found to be 50, well over the default of $\frac{p}{3}$. The number of trees with the lowest CV RMSE was 500.



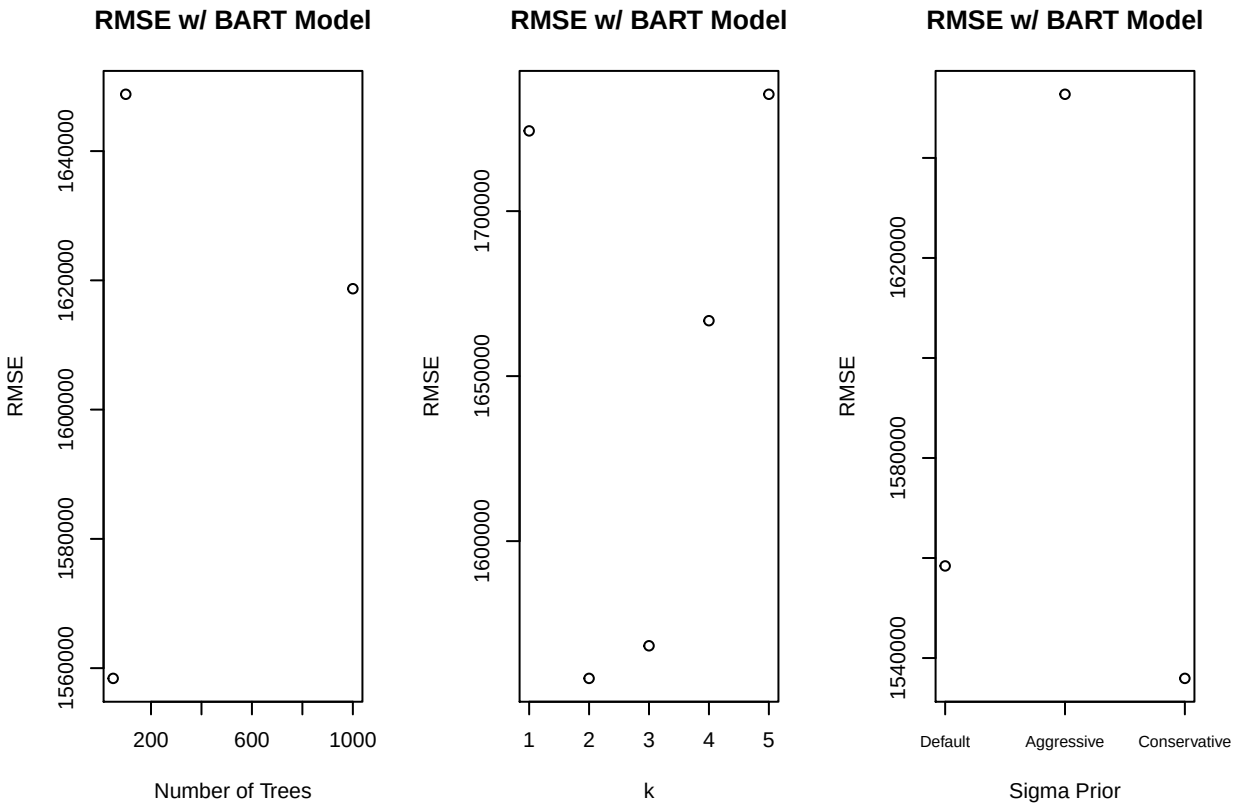
Boosting

The Boosting model was optimized by varying the number of trees, interaction depth and shrinkage and checking cross-validation error on the training data set. The optimum number of trees was small (1000); the optimal interaction depth was 3 and the shrinkage factor was 0.01.



BART

The BART Model was tuned manually using a 5-fold CV function. The number of trees and the parameters k, q and ν were all optimized. The parameters q and ν were tuned together per the 'default', 'aggresive' and 'conservative' setting listed in the documentation. Again, a small number of trees were selected by the CV function.



MARS – Best Predictive Model

(a) The MARS model was fit using the *earth()* function. As described in the above EDA section, a log transform was used on the response variable. The default pruning method ("backward") was used, although all pruning method were tried. The default method had the best training error and was selected for use.

(b) MARS uses hinge functions (see Figure 1 from the ESL book) or products of pairs of hinge functions as basis functions. A hinge function takes form of zero on one side of a "knot" (0.5 in the Figure 1) and is linear on the opposite side. Knots are at each observation.

The model has the form:

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X)$$

where $h_m(X)$ is a pair of hinge functions or their products. The final model specified numerically is:

```
##   60.83015
##   +     1.317876 * PBA13
##   +    0.6704618 * PBA16
```

```
##   +    0.8863036 * PBA26
##   +     1.436075 * PBA4
##   -     1.501465 * PBAPLUS18
##   -    0.7545645 * PBAPLUS25
##   +    0.9414023 * PBAPLUS32
##   +    0.6973177 * PBAPLUS33
##   -    0.8267593 * PBAPLUS46
##   -    0.1537053 * FACIL2
##   +     0.377209 * HT12
##   -    0.3761607 * FKUSED2
##   +    0.3256835 * PRUSED2
##   -      1.46737 * OTCLEQ2
##   -    0.5263408 * RDCLNF2
##   -     1.755727 * PUBCLIM7
##   -  0.000106287 * h(SQFT-8800)
##   - 0.0001445472 * h(66000-SQFT)
##   + 0.0001090644 * h(SQFT-66000)
##   +    0.0253252 * h(FLCEILHT-28)
##   +     1.411377 * h(2-NOCCAT)
##   -    0.1685639 * h(9-MONUSE)
##   -  0.003320173 * h(100-WKHRS)
##   -   0.07777523 * h(NWKER-8)
##   -   0.07991691 * h(500-NWKER)
##   +   0.07764775 * h(NWKER-500)
##   -   0.06953244 * h(25-COOLP)
##   +   0.01441096 * h(COOLP-25)
##   -     1.972984 * MAINCL4
##   + 0.0001165952 * h(3975-HDD65)
##   -  0.009804563 * h(CDD65-164)
##   -   0.01259119 * h(414-CDD65)
##   +   0.01029136 * h(CDD65-414)
##   - 0.0004769558 * h(CDD65-2099)
##
```

Plots of these relationships are available as well (Figure 2 below).

Notice that in the equation *CDD65* has four 'knots', which are visible in the plot.

(c) MARS was selected because of its out-of-sample performance and the (relative) ease of inference (see d). Its runtime is great as well.

(d) The most important predictors have been identified by the *earth()* model. See table:

```
##           nsubsets   gcv    rss
## SQFT            34 100.0  100.0
## COOLP           33  62.1   63.4
## CDD65           32  51.5   53.2
```

```
## NWKER          31  45.4   47.4
## MAINCL4        26  29.4   32.2
## PBA13          25  27.1   30.0
## PBA16          23  22.6   26.0
## PBA4           22  20.9   24.4
## RDCLNF2        21  19.5   23.0
## PBAPLUS18      20  17.5   21.3
## PBAPLUS32      18  15.2   19.1
## FKUSED2        16  13.5   17.5
## MONUSE         16  13.0   17.1
## PBA26          15  12.0   16.1
## HDD65          14  11.4   15.4
## PUBCLIM7       14  11.4   15.4
## PBAPLUS33      11   8.9   12.8
## OTCLEQ2        10   7.8   11.9
## FLCEILHT        9   6.9   10.9
## PRUSED2         8   6.0   10.0
## NOCCAT          7   5.3    9.3
## PBAPLUS46       6   4.4    8.3
## PBAPLUS25       5   3.6    7.4
## HT12            4   3.1    6.6
## FACIL2          2   1.9    4.5
## WKHRS           1   1.5    3.3
```

When *earth()* prunes backward from the full model, it counts the number of models in which each variable is present. That number is in "nsubsets". Thus, *SQFT* was in 34 of the models checked during the backward prune. *SQFT* is also most important in GCV (similar to LOOCV) and RSS (which measures the RSS drop when a predictor is removed from the model). Thus, the top-3 most important variables are: *square footage*, *cooling percentage (of the building)*, and *cooling degree days*.

Conclusion

I choose the MARS model because it performed well, runs quickly and is interpretable!

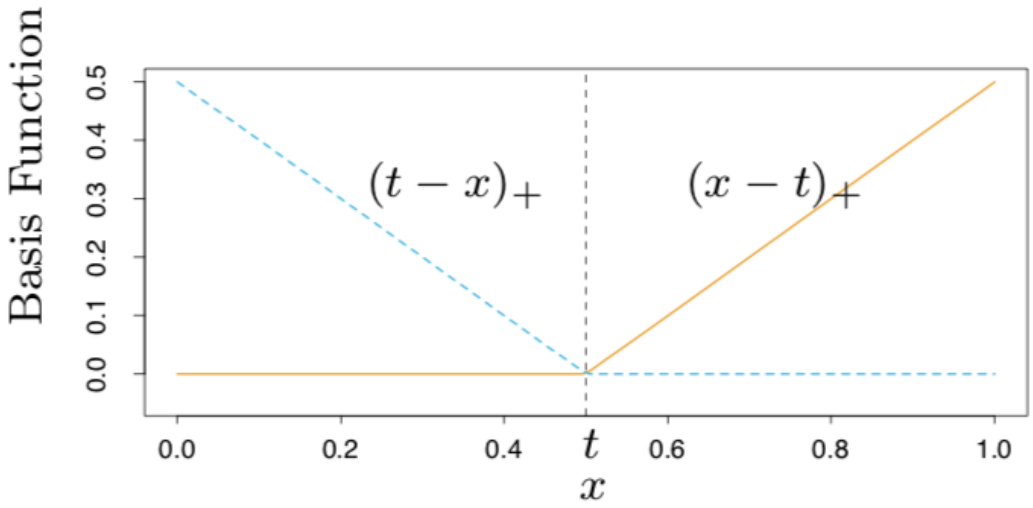


Figure 1: Hinge Function

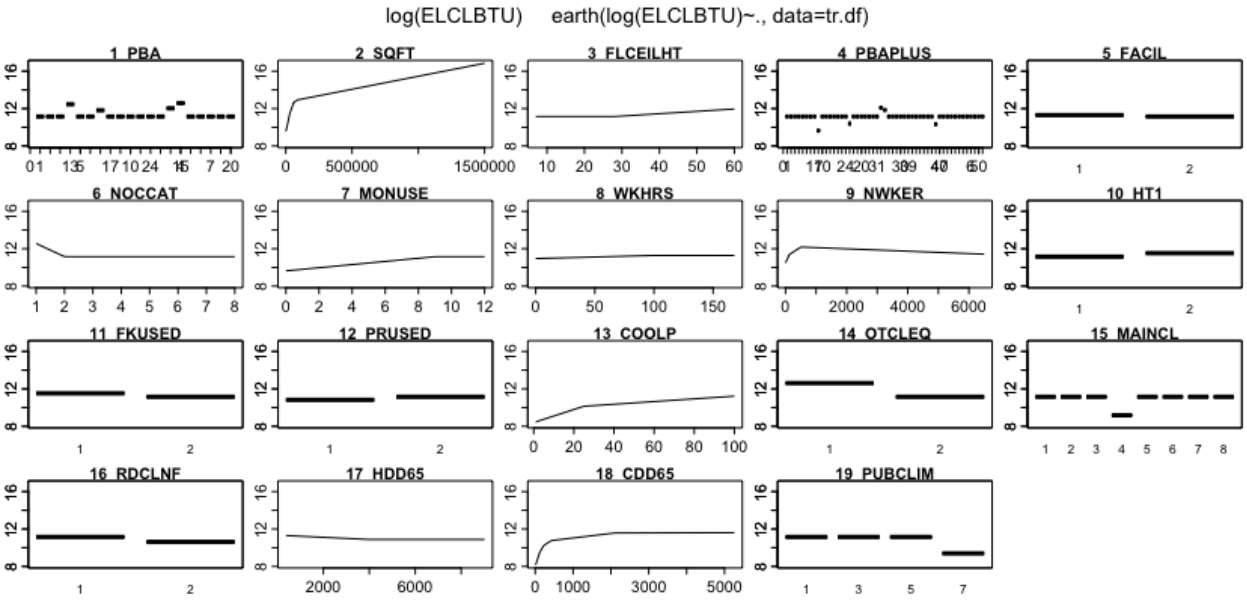


Figure 2: MARS Output Charts