Statistical Modeling - Project

Ksenia Lepikhina

May 2, 2019

1 Introduction

To give some context to the problem we are exploring, I would like to begin by describing the datasets. The first file contains two non-consecutive years of average monthly temperatures (in Celsius) for 51 US cities; one slightly colder year and one slightly warmer. The data is structured as such: each row is one of the 51 US cities and each column is one month for those two years. The second file contains electricity usage (in hundreds of gigawatt hours, Gwh) for the cities (rows) during those two years in months (columns). The final file contains the average monthly temperatures (in Celsius) for Denver in a 3rd random year. Our goal is to predict the energy usage for that third year.

2 Analysis

2.1 Model 1: SLR

As an initial model, we can begin by simply fitting a linear model between the data sets where Y is the energy usage and X is the average temperature in the cities.

Simply by regressing temperature on energy, we find that the relationship is non linear. This can be seen in figure 1. There is a curve that looks like it follows a similar shape to $\frac{1}{x}$ rather than a line. While this model would be able to produce an energy for a given temperature, it is almost certainly not the "best" model.

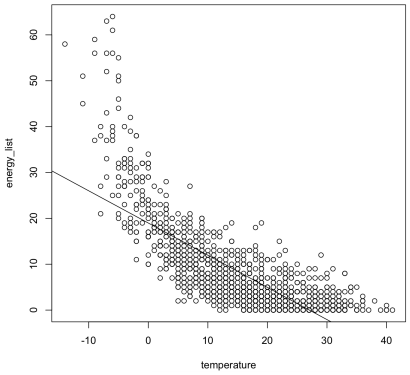


Figure 1: Plot regression

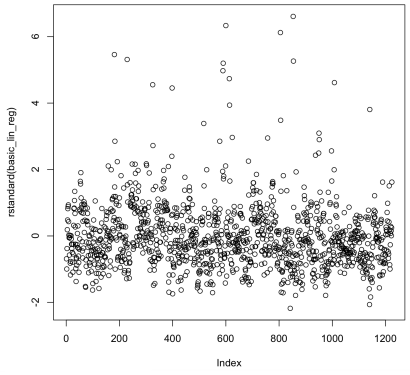


Figure 2: Std resid vs fit vals

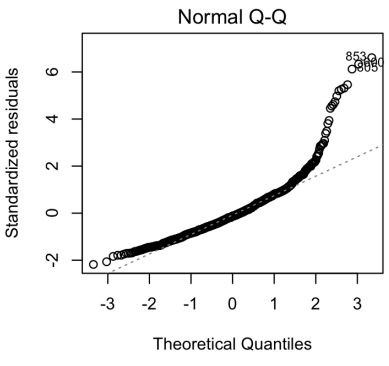


Figure 3: Normal QQ

Not only does temperature have a nonlinear relationship with energy, this model also does not satisfy the linearity assumptions as demonstrated in plots 2 and 3. The first is an image that shows that the data is not homoskedatic because of the data appearing to cluster lower down while having some outliers above. The normal QQ plot shows us that the data is fairly normal except for higher residual values.

The R^2 value of this regression is 0.5837 which is implying that only a little over half of the variation is explained by the regression.

2.2 Model 2: SLR with exponential transformation

Above, I noted that the shape of the data was similar to $\frac{1}{x}$. This function is also similar to e^x. Because of this, the transformation we are looking for to make our data linear is the exponential transformation. Right now the equation for our data is similar to:

$$Y = \alpha e^{\beta_1 x}$$

When we apply the exponential transform, we get the following:

$$ln(Y) = ln(\alpha) + \beta_1 x$$
$$ln(Y) = \beta_0 + \beta_1 x$$

If we apply an exponential model to the data as our transformation, we already have better results than in model 1. The plot of the transformed data and its regression line can be seen below:

Though the residuals aren't tightly spread around the regression line, the regression evidently fits the data better here than it did in the simple untransformed linear regression above.

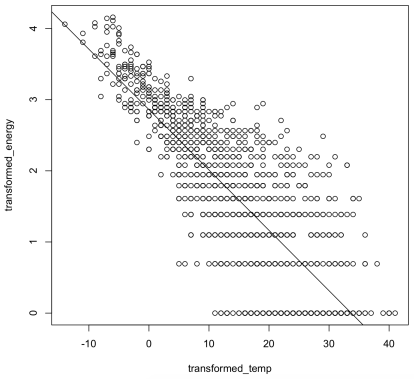


Figure 4: Plot reg - exponential

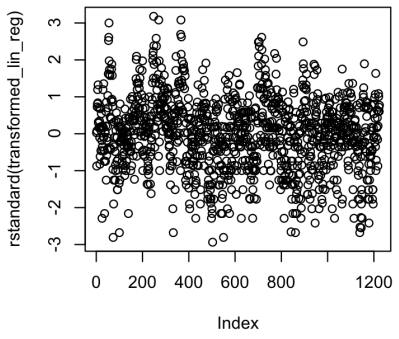


Figure 5: Std resid vs fit vals - exponential

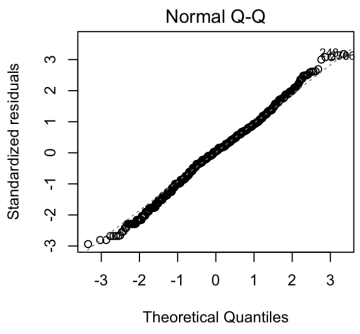
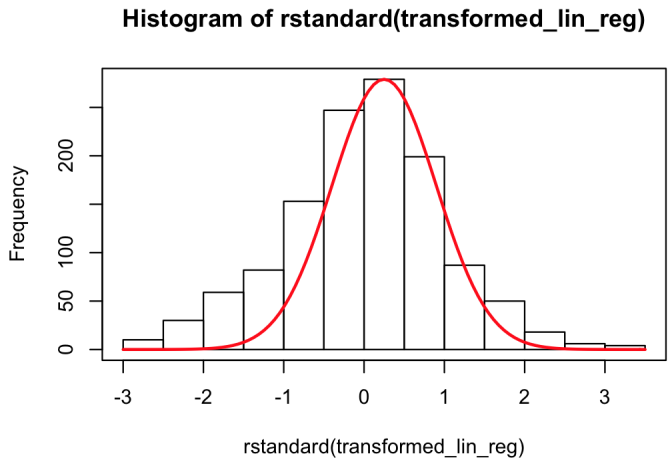


Figure 6: Normal QQ - exponential

Figures 5 and 6 confirm that this model is already better than the first. The standardized residuals are significantly better spread and imply that the homoskedasticity assumption is satisfied. Additionally, since the residuals appears to lack any distinct pattern and appears to be even spread, then we can state that the linearity assumption is also satisfied. Normally, we could also look at the component plus residual plots as well, however since we only have one predictor, it isn't necessary here.

We also see that the normality assumption is satisfied by studying the other figure. The points hug the dashed diagonal line well. We can double check the normal QQ results by simply plotting the standardized residuals and a normal distribution in one. We can see this below:



The R^2 value in this model is 0.6454. While this transformed model does not perform significantly better than the previous model, it does explain more of the variation than in model 1.

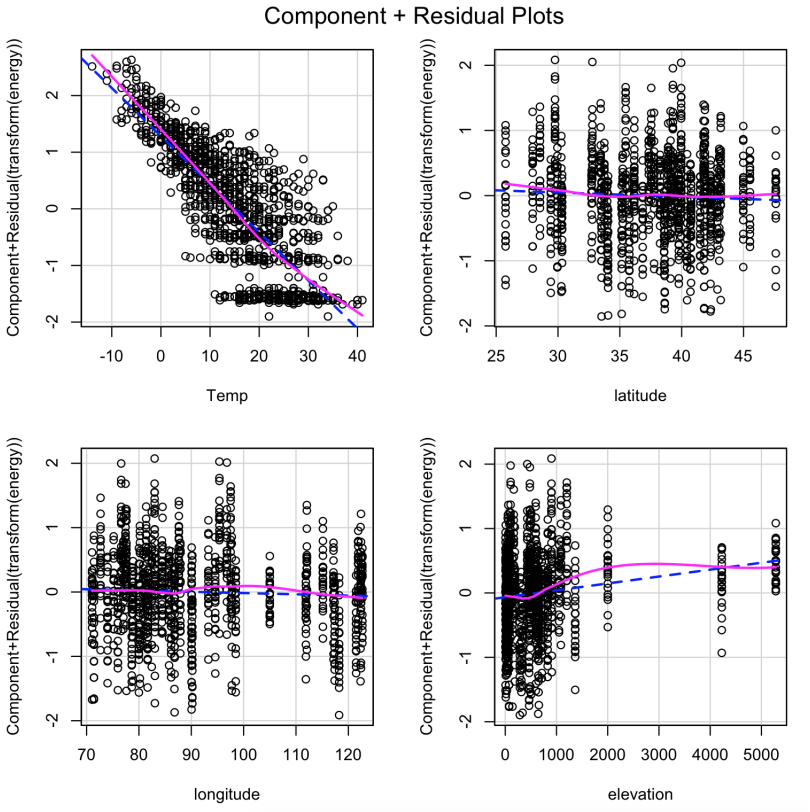
2.3 Model 3: MLR

Since the capabilities of a simple linear regression have a limit, I decided to try a multiple linear regression next. This is the multiple linear regression used:

$$ln(\text{energy}) = \beta_0 + \beta_1 \text{temperature} + \beta_2 \text{latitude} + \beta_3 \text{longitude} + \beta_4 \text{elevation}$$

The transformation is the same as model 2. Here, I added latitude, longitude and elevation (essentially constant values regardless of time). Adding these constant parameters did improve the model. The adjusted R^2 value here is 0.6522 which is an improvement on the R^2 value in model 2: 0.6454.

Notice, since the improvement in R^2 is fairly small, we do not see a significant change in the standardized residual versus fitted value plot and the normal QQ plot between model 2 and model 3. In fact, those two plots were excluded from this paper due to their extreme similarity. This model maintains the linearity assumptions: homoskedasticity, linearity, normality and implicitly independence.



Instead of plotting the regression line (for simple linear regression) we can look at the component plus residual plots seen above. These plots are an excellent way to visualize whether or not each predictor has a good linear relationship with the dependent variable. The pink line represents the component line and the blue line represents the line of best fit. If the two lines match up pretty well, then the predictor has a linear relationship with the dependent variable. From these definitions we can state that each of the predictors has a reasonable linear fit.

3 Comparison

We can compare the models using AIC and BIC. AIC is an estimate of a constant and the distance between the true likelihood function of the data and the fitted likelihood of the model. BIC is an estimate measured under a Bayesian setup that checks for a function of the posterior density of the model being true. Based on how these are defined, a lower AIC and a lower BIC are desired.

```
> AIC(basic_lin_reg, transformed_lin_reg, mlr)
                    df      AIC
basic_lin_reg        3 7939.224
transformed_lin_reg  3 2447.387
mlr                  6 2425.637
> BIC(basic_lin_reg, transformed_lin_reg, mlr)
                    df      BIC
basic_lin_reg        3 7954.554
transformed_lin_reg  3 2462.716
mlr                  6 2456.296
```

From the results above, we note that the multiple linear regression returns the lowest AIC and BIC value. This implies that the third model is indeed the best model. We can also see that since the transformed regression has a lower AIC and BIC than the simple linear regression, that the second model is better than the first.

Another way to compare models is using the F-test (*anova()* in R). However, the F-test is only applicable for nested models. Since our models are not nested but rather each apply new methods of analysis, AIC and BIC are best for model comparison.

Model comparison (discuss validity of assumptions, model fit, parsimony/complexity, and interpretability of the models)

4 Prediction

The prediction intervals of the above predicted energy usage for each of the months can be seen in the table below, where the first three columns are the transformed fitted results and the next set of three columns are the fitted results:

	fit ln(Gwh)	lwr ln(Gwh)	upr ln(Gwh)	fit Gwh	lwr Gwh	upr Gwh
Jan	2.8597876	1.5711607	4.148415	17.457819	4.8122305	63.333511
Feb	2.5624920	1.2740233	3.850961	12.968093	3.5752078	47.038229
Mar	2.1632664	0.8747922	3.451741	8.699507	2.3983768	31.555269
Apr	1.9084415	0.6198331	3.197050	6.742572	1.8586178	24.460265
May	1.7130758	0.4242955	3.001856	5.545994	1.5285132	20.122852
Jun	1.0760137	-0.2137424	2.365770	2.932964	0.8075564	10.652235
Jul	0.5578698	-0.7331479	1.848888	1.746947	0.4803944	6.352748
Aug	0.4899165	-0.8012978	1.781131	1.632180	0.4487462	5.936565
Sep	0.9655896	-0.3244002	2.255579	2.626336	0.7229608	9.540819
Oct	1.4157801	0.1266236	2.704937	4.119699	1.1349897	14.953370
Nov	2.0273598	0.7388267	3.315893	7.594010	2.0934777	27.546980
Dec	2.5030329	1.2145792	3.791487	12.219498	3.3688761	44.322237

Note that the prediction intervals for each month are incredibly large. Evidently, while the third model was the best of the three models, it is still far from the hypothetical best model for the data.

5 Discussion

Again, the third model is the best model in this paper, but is far from the best model that can possibly be used. One significant indicator of this is the adjusted R^2. This value is closer to 0.5 than it is to 1 where 1 means that the regression prediction perfectly fits the data. While the third model is an improvement on the second model, it is clear that the predictors latitude, longitude, and elevation do not improve the model significantly as judged by the R^2 value, LS assumptions and AIC/BIC, implying that the second model can be used instead of the third, for simplicity purposes.

To continue this work, it would be interesting to look into the applications of generalized linear models to this problem by specifically, looking into how seasonality affects the energy predictions and looking at change point problems as well as poisson regression.

6 Sources

- http://www.r-tutor.com/elementary-statistics/multiple-linear-regression/prediction-interval-mlr
- http://www.r-tutor.com/elementary-statistics/simple-linear-regression/prediction-interval-linear-regression
- https://stattrek.com/regression/linear-transformation.aspx

7 Appendix

Read the data:

```
temp = read.table('Documents/Stats Modeling/Project/FinalTempdata1.csv',
    sep=",", header=TRUE)
energy = read.table('Documents/Stats Modeling/Project/EnergyData6.csv',
    sep=",", header=TRUE)
temp3 = read.table('Documents/Stats Modeling/Project/FinalTempYr3.csv',
    sep=",", header=TRUE)
```

Convert the data into list format:

```
temp_cropped <- temp[ , !(names(temp) %in% c('City'))]
temperature <- c(t(temp_cropped))
energy_list <- c(t(energy))
```

SLR Model 1:

```
simple = data.frame(energy=energy_list, Temp=temperature)
basic_lin_reg <- lm(energy ~ Temp, data=simple)
summary(basic_lin_reg)

par(mfrow=c(1,1))
plot(temperature, energy_list)
abline(basic_lin_reg)

par(mfrow=c(2,2))
plot(basic_lin_reg)

par(mfrow=c(1,1))
plot(rstandard(basic_lin_reg))
```

SLR transformed Model 2:

```
library(MASS)
```

```
transform = function(lst){
  lst_new = c()
  for (i in 1:length(lst)){
    if (lst[i] <= 0 ){ lst_new[i] = 0 }
    else{ lst_new[i] = log(lst[i]) }
  }
  lst_new
}

transformed_energy <- transform(energy_list)
transformed_temp <- temperature

transformed_lin_reg <- lm(transformed_energy ~ transformed_temp)
summary(transformed_lin_reg)

plot(transformed_temp, transformed_energy)
abline(transformed_lin_reg)

par(mfrow=c(2,2))
plot(transformed_lin_reg)

plot(rstandard(transformed_lin_reg))

hist(rstandard(transformed_lin_reg),10)
par(new=TRUE)
xgrid = seq(-5,5,by=0.1)
xdensity = dnorm(xgrid,0,1)
plot(xgrid, xdensity, 'l', lwd=2,col='red', xaxt='n', yaxt='n', xlab='', ylab='')

library(car)
par(mfrow=c(1,1))
influencePlot(transformed_lin_reg)
```

Get lat, long, elevation:

```
cities <- as.vector(temp['City'])
latitude <- c(33.75, 30.27, 39.29, 33.52, 42.36, 42.89, 35.23, 41.88, 39.10,
            41.50, 39.96, 32.78, 39.74, 42.33, 41.77, 29.76, 39.77, 30.33,
            39.10, 36.17, 34.05, 38.25, 35.15, 25.76, 43.04, 44.98, 36.16,
            29.95, 40.71, 35.47, 28.54, 39.95, 33.45, 40.44, 45.52, 41.82,
            35.77, 37.54, 33.98, 43.16, 38.58, 40.76, 29.42, 32.72, 37.77,
            37.34, 47.61, 38.63, 27.95, 36.85, 38.91)
longitude <- c(84.39, 97.74, 76.61, 86.81, 71.06, 78.88, 80.84, 87.63, 84.51,
            81.69, 83.00, 96.80, 105.00, 83.05, 72.67, 95.37, 86.16, 81.66,
            94.58, 115.14, 118.24, 85.76, 90.05, 80.19, 87.91, 93.27, 86.78,
            90.07, 74.01, 97.52, 81.38, 75.17, 112.07, 80.00, 122.68, 71.41,
```

```
            78.63, 77.44, 117.38, 77.61, 121.49, 111.89, 98.49, 117.17,
            122.42, 121.89, 122.33, 90.20, 82.45, 75.98, 77.04)
elevation <- c(1050, 489, 480, 643, 141, 600, 746, 594, 482, 653, 902, 430,
            5280, 656, 59, 105, 715, 16, 909, 2001, 285, 466, 338, 6, 617,
            830, 597, 20, 33, 1201, 82, 39, 1086, 1365, 50, 75, 315, 166,
            827, 505, 30, 4226, 650, 62, 52, 82, 187, 466, 48, 10, 10)

geographic = data.frame(cities=cities, latitude=latitude, longitude=longitude,
    elevation=elevation)

plot(latitude, longitude)

geographic_expanded <- geographic[rep(seq_len(nrow(geographic)), each=24),]

cities_expanded <- as.vector(geographic_expanded['City'])
latitude_expanded <- as.vector(geographic_expanded['latitude'])
longitude_expanded <- as.vector(geographic_expanded['longitude'])
elevation_expanded <- as.vector(geographic_expanded['elevation'])

months_expanded <- c(rep(c(1,2,3,4,5,6,7,8,9,10,11,12), 102))

all_data <- data.frame(cities=cities_expanded, month=months_expanded,
    temperature=temperature, energy=energy_list,
    latitude=latitude_expanded, longitude=longitude_expanded,
    elevation=elevation_expanded)
```

MLR Model 3:

```
mlr <- lm(transform(energy) ~ temperature + latitude + longitude +
    elevation, data=all_data)
summary(mlr)

par(mfrow=c(2,2))
plot(mlr)

plot(rstandard(mlr))

library(car)
crPlots(mlr)
```

Prediction Intervals

```
predict(basic_lin_reg, temp3["Temp"], interval="predict")

predict(transformed_lin_reg, temp3["Temp"], interval="predict")
```

```
temp3['latitude'] <- rep(c(latitude[13]), 12)
temp3['longitude'] <- rep(c(longitude[13]), 12)
temp3['elevation'] <- rep(c(elevation[13]), 12)

predict(mlr, temp3[,2:5], interval="predict")
exp(predict(mlr, temp3[,2:5], interval="predict"))
```