Monthly Giving Per Week Modeling

Sean Davern

June 26, 2019

Contents

Overview

Work to model giving to Seed Church in 2015 noted that giving each month is a function of the number of Sundays that occur in the month on top of varying, somewhat predictably, from month to month. It is also a function of the number of giving families attending Seed Church. This work aims at regressing linear a model of the measure that includes those factors to tools useful for forecasting.

$$\frac{Monthly\ Giving}{n_{Sundays\ in\ Month} \cdot n_{Giving\ Families}} = a + b_{year} + c_{month}$$

Load Data

Loading cleaned and transformed (sanitized) giving data originally provided by John Earling:

```r
df <- readRDS("../data/Cleaned and Transformed Giving Data.rds")
df
```

week.ending	month	year	total	monthly.giving.families	SundaysInMonth	MonthsGivingPerWeek
2010-01-31	January	2010	14241.15	45	5	2848.230
2010-02-28	February	2010	22437.50	41	4	5609.375
2010-03-28	March	2010	13317.00	32	4	3329.250
2010-04-25	April	2010	13232.50	36	4	3308.125
2010-05-30	May	2010	13478.15	40	5	2695.630
2010-06-27	June	2010	14930.50	43	4	3732.625

week.ending	month	year	total	monthly.giving.families	SundaysInMonth	MonthsGivingPerWeek
2010-07-25	July	2010	12824.75	36	4	3206.188
2010-08-29	August	2010	15776.00	38	5	3155.200
2010-09-26	September	2010	9965.75	30	4	2491.438
2010-10-31	October	2010	16112.50	40	5	3222.500
2010-11-28	November	2010	11976.96	30	4	2994.240
2010-12-26	December	2010	15511.80	37	4	3877.950
2011-01-30	January	2011	13992.25	37	5	2798.450
2011-02-27	February	2011	13034.00	30	4	3258.500
2011-03-27	March	2011	18106.50	29	4	4526.625
2011-04-24	April	2011	11416.50	29	4	2854.125
2011-05-29	May	2011	15182.00	35	5	3036.400
2011-06-26	June	2011	11572.00	31	4	2893.000
2011-07-31	July	2011	20739.00	35	5	4147.800
2011-08-28	August	2011	10144.00	30	4	2536.000
2011-09-25	September	2011	13236.75	31	4	3309.188
2011-10-30	October	2011	18500.50	37	5	3700.100
2011-11-27	November	2011	10759.00	30	4	2689.750
2011-12-25	December	2011	15518.00	32	4	3879.500
2012-01-29	January	2012	13648.00	35	5	2729.600
2012-02-26	February	2012	12239.25	34	4	3059.812
2012-03-25	March	2012	13673.00	34	4	3418.250
2012-04-29	April	2012	20654.00	38	5	4130.800
2012-05-27	May	2012	14723.00	32	4	3680.750
2012-06-24	June	2012	16512.00	32	4	4128.000
2012-07-29	July	2012	17662.00	32	5	3532.400
2012-08-26	August	2012	13620.00	28	4	3405.000
2012-09-30	September	2012	16768.00	32	5	3353.600
2012-10-28	October	2012	18888.00	41	4	4722.000
2012-11-25	November	2012	13203.00	31	4	3300.750
2012-12-30	December	2012	25901.82	43	5	5180.364
2013-01-27	January	2013	12812.02	30	4	3203.005
2013-02-24	February	2013	16249.00	44	4	4062.250
2013-03-31	March	2013	18255.00	39	5	3651.000
2013-04-28	April	2013	17104.25	38	4	4276.062
2013-05-26	May	2013	14321.00	30	4	3580.250
2013-06-30	June	2013	16460.00	29	5	3292.000
2013-07-28	July	2013	15013.00	35	4	3753.250
2013-08-25	August	2013	15787.59	37	4	3946.898
2013-09-29	September	2013	24086.25	43	5	4817.250
2013-10-27	October	2013	20302.00	40	4	5075.500
2013-11-24	November	2013	17104.20	39	4	4276.050
2013-12-29	December	2013	24291.36	41	5	4858.272
2014-01-26	January	2014	15141.00	34	4	3785.250
2014-02-23	February	2014	16818.00	34	4	4204.500
2014-03-30	March	2014	23975.00	40	5	4795.000
2014-04-27	April	2014	18757.50	41	4	4689.375
2014-05-26	May	2014	25092.46	44	4	6273.115
2014-06-29	June	2014	21425.25	40	5	4285.050
2014-07-27	July	2014	26726.07	35	4	6681.517
2014-08-31	August	2014	28223.86	41	5	5644.772
2014-09-28	September	2014	26282.00	50	4	6570.500
2014-10-26	October	2014	22165.00	38	4	5541.250

week.ending	month	year	total	monthly.giving.families	SundaysInMonth	MonthsGivingPerWeek
2014-11-30	November	2014	31675.75	54	5	6335.150
2014-12-28	December	2014	28709.10	44	4	7177.275
2015-01-25	January	2015	21491.67	43	4	5372.917
2015-02-22	February	2015	32134.50	49	4	8033.625
2015-03-29	March	2015	33296.00	49	5	6659.200
2015-04-26	April	2015	33733.15	59	4	8433.288
2015-05-31	May	2015	29626.00	47	5	5925.200
2015-06-28	June	2015	21316.00	42	4	5329.000
2015-07-26	July	2015	37795.00	44	4	9448.750
2015-08-30	August	2015	33004.00	50	5	6600.800
2015-09-27	September	2015	23513.00	42	4	5878.250
2015-10-25	October	2015	31935.68	50	4	7983.920
2015-11-29	November	2015	35684.00	42	5	7136.800
2015-12-27	December	2015	38209.00	51	4	9552.250
2016-01-31	January	2016	25290.00	49	5	5058.000
2016-02-28	February	2016	26553.00	49	4	6638.250
2016-03-27	March	2016	24124.40	42	4	6031.100
2016-04-24	April	2016	31186.00	48	4	7796.500
2016-05-29	May	2016	30338.55	48	5	6067.710
2016-06-26	June	2016	23434.62	39	4	5858.655
2016-07-31	July	2016	27120.37	37	5	5424.074
2016-08-28	August	2016	26536.41	37	4	6634.102
2016-09-25	September	2016	35073.51	46	4	8768.378
2016-10-30	October	2016	31743.10	46	5	6348.620
2016-11-27	November	2016	42092.80	38	4	10523.200
2016-12-25	December	2016	63802.30	38	4	15950.575
2017-01-29	January	2017	25782.65	38	5	5156.530
2017-02-26	February	2017	19794.58	35	4	4948.645
2017-03-26	March	2017	21056.35	34	4	5264.087
2017-04-30	April	2017	28764.14	41	5	5752.828
2017-05-28	May	2017	30524.58	39	4	7631.145
2017-06-25	June	2017	20171.15	33	4	5042.788
2017-07-30	July	2017	22253.58	35	5	4450.716
2017-08-27	August	2017	17046.81	34	4	4261.703
2017-09-24	September	2017	26945.58	31	4	6736.395
2017-10-29	October	2017	21422.87	30	5	4284.574
2017-11-26	November	2017	27615.58	36	4	6903.895
2017-12-31	December	2017	27601.72	38	5	5520.344
2018-01-28	January	2018	17570.18	32	4	4392.545
2018-02-25	February	2018	19761.75	35	4	4940.438
2018-03-25	March	2018	18819.41	35	4	4704.852
2018-04-29	April	2018	20386.74	35	5	4077.348
2018-05-27	May	2018	21185.58	32	4	5296.395

Adding Columns to Accomodate Analysis

I'll add a couple columns to enable accommodating for giving anomolies:

```
suppressMessages(library("dplyr", quietly=TRUE, warn.conflicts = FALSE))
df <- mutate(df,
             removed.amount = 0,
             original.total = total)
```

Accommodating an Unusually Large Gift

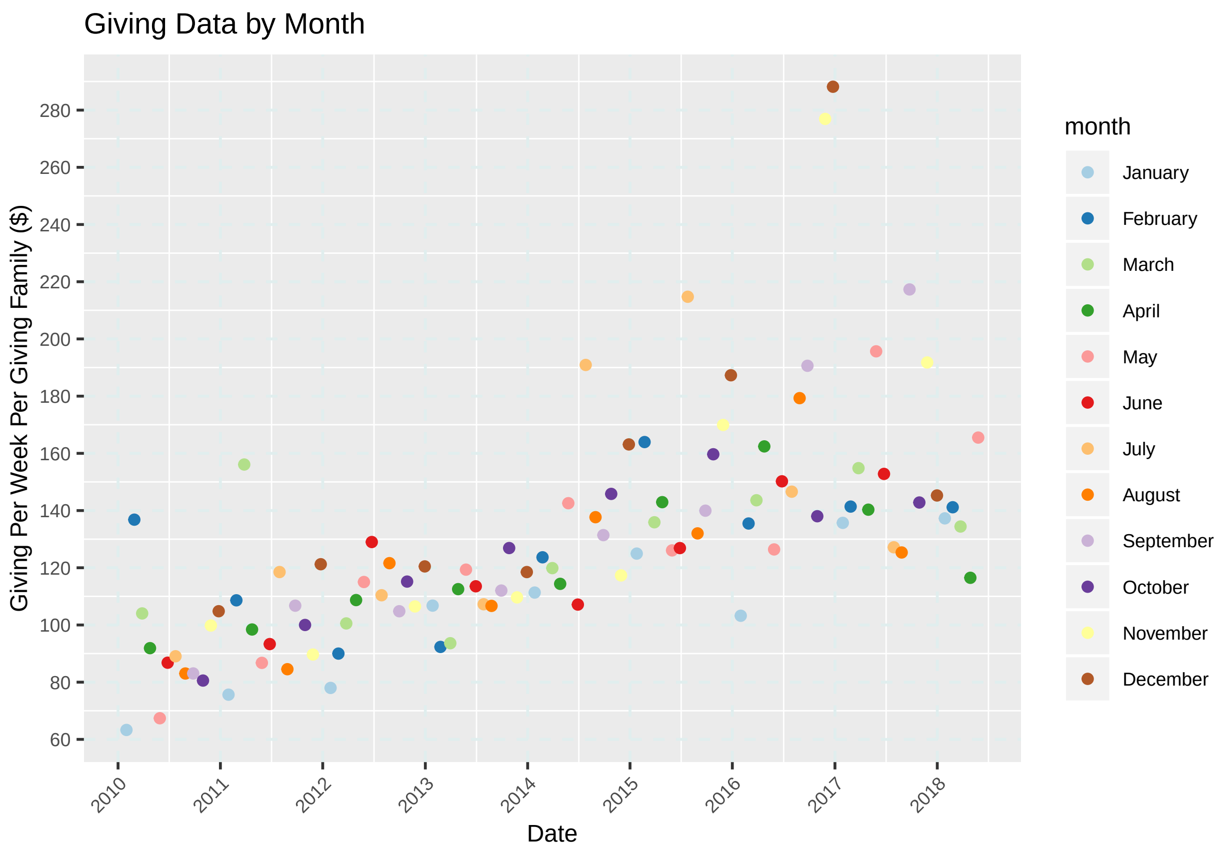
12/26/2016 had an unusual single gift skewing it and analyses. Other large gifts have been recieved over this period but this gift was at least 50% larger than all others. I'm removing the amount of the unusual gift from its month's total.

```
df$removed.amount[as.Date(df$week.ending) == "2016-12-25"] <- 20000
df$total <- df$original.total - df$removed.amount
```

The Measure to be Modeled

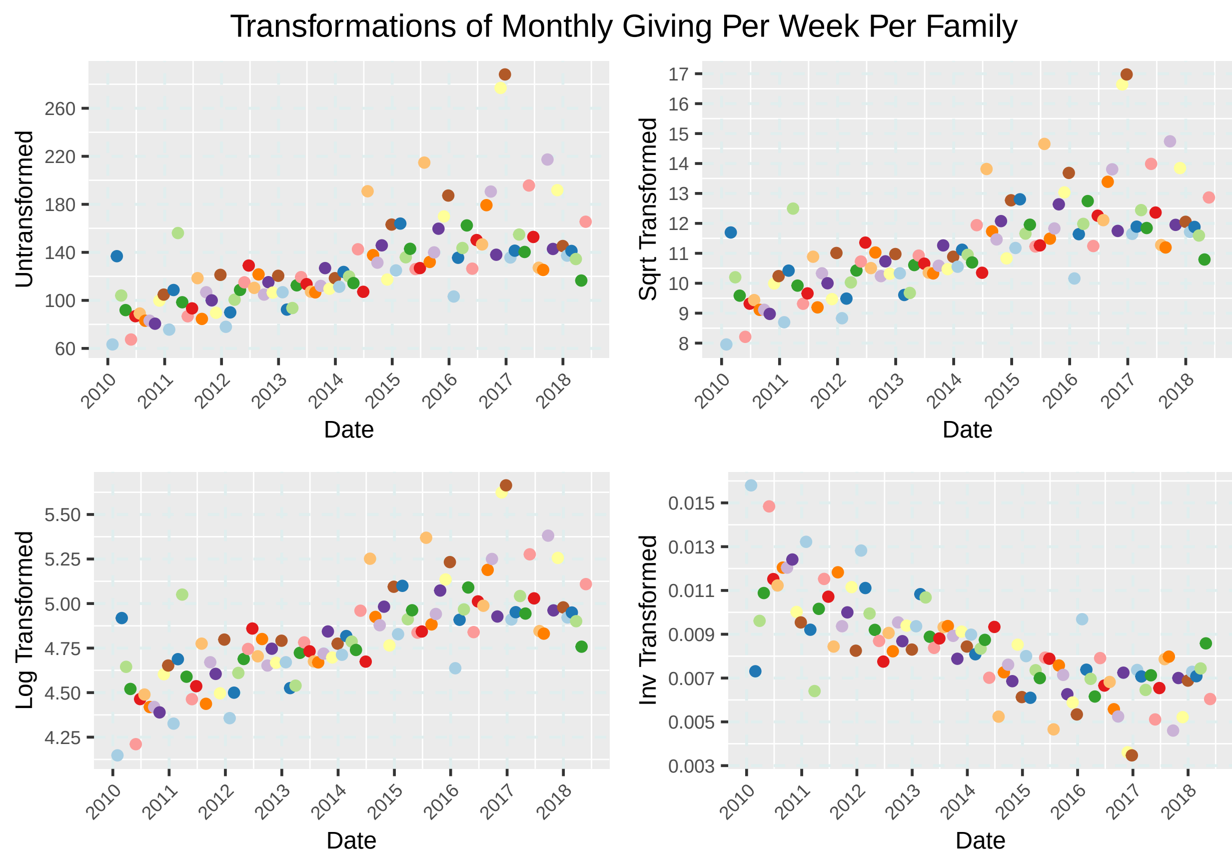
The following plot shows the trend of each month's giving per week per giving family.

```
df$MonthsGivingPerWeek <- df$total /
                                df$SundaysInMonth /
                                df$monthly.giving.families
```



This chart seems to show increasing variation over time especially if the high November and December 2016 data is not anomolous. Following methods recommended by NIST the following plots allow assessing heteroscedasticity for potential measure transformations (square root, natural log, and inverse (1/x)):

```
# Create columns of transformed responses:
library("magrittr")
df <- mutate(df, SqrtMonthsGivingPerWeek = sqrt(df$MonthsGivingPerWeek))
df <- mutate(df, logMonthsGivingPerWeek = log(df$MonthsGivingPerWeek))
df <- mutate(df, InvMonthgivingPerWeek = 1/df$MonthsGivingPerWeek)
```



None of the transformations are dramatically better heterscadicity. Of particular question is whether the trend in the measure is mostly linear through the entire period or if it flattens and perhaps even begins to decrease. Keep in mind, this isn't the total giving, that definitely decreases into 2018. This is whether the average giving per week per family decreases. Such a shift could be a signal of decreased faith in the church or a byproduct of the nature of the giving by the collection of families making up the congregation. Anyway, the end of 2016 high values in all cases seem to be slightly anamolous even though I've alread removed the single large gift in December 2016. Despite the similarities in variance consistency, I'm going to move forward with the log transformation because it seem slightly better than untransformed.

Modeling

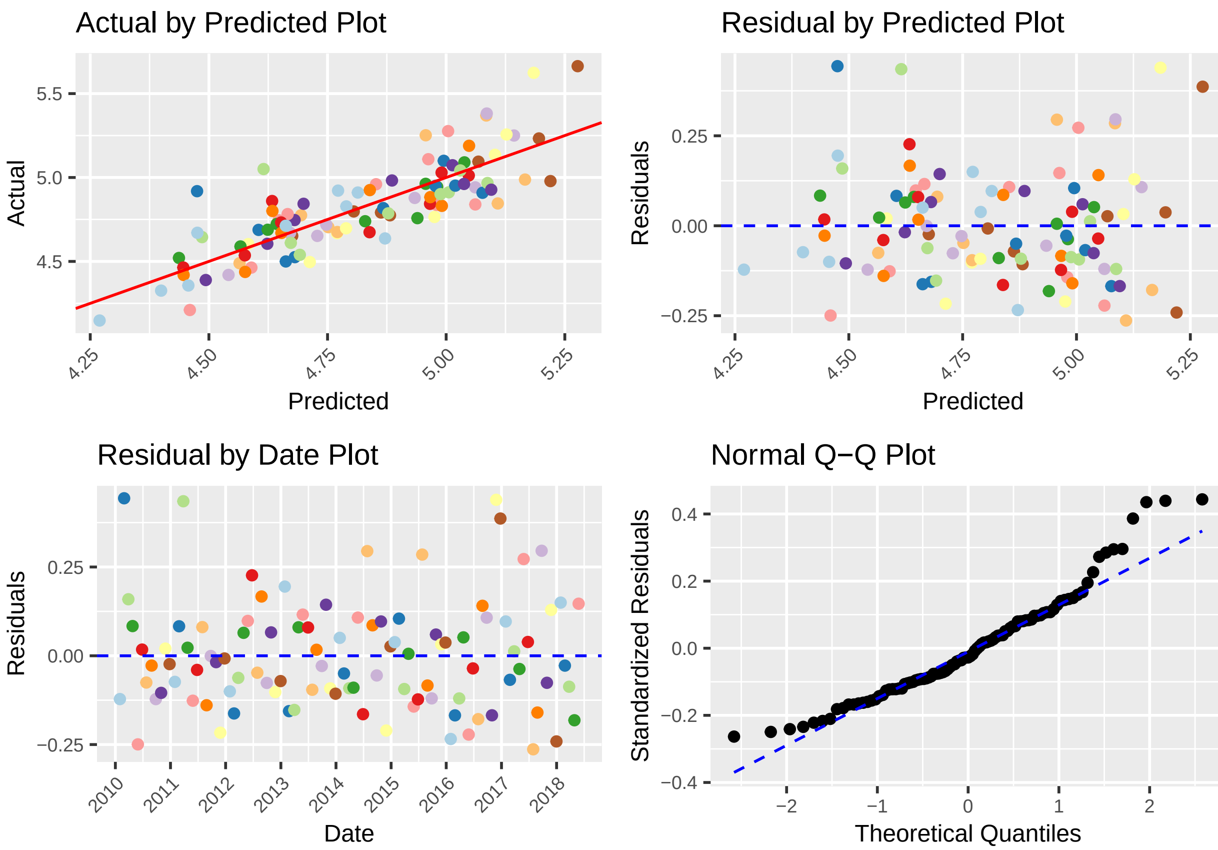
Regressing the R version (vs. the JMP approach) of this model:

$$ln\left[\frac{Monthly\ Giving}{Number\ of\ Sundays\ in\ Month * Giving\ Families}\right] = a + b_{year} + c_{month} + \epsilon$$

where a, b_{year} and c_{month} are regressed model coefficients and ϵ is error.

```
mod <- lm( formula = logMonthsGivingPerWeek ~ year + month, data = df)
# Calculate month total giving using regression results:
df <- mutate(df, predicted.values = exp(predict(mod,df))  *
                SundaysInMonth *
                monthly.giving.families) %>%
  mutate(residual.values = total - predicted.values )
```

gives the following model fit assessment:



	Df	Sum Sq	Mean Sq	F value	Pr(>F)
year	8	4.1901904	0.5237738	17.999268	0.0000000
month	11	0.8924638	0.0811331	2.788104	0.0040019
Residuals	81	2.3570779	0.0290997	NA	NA

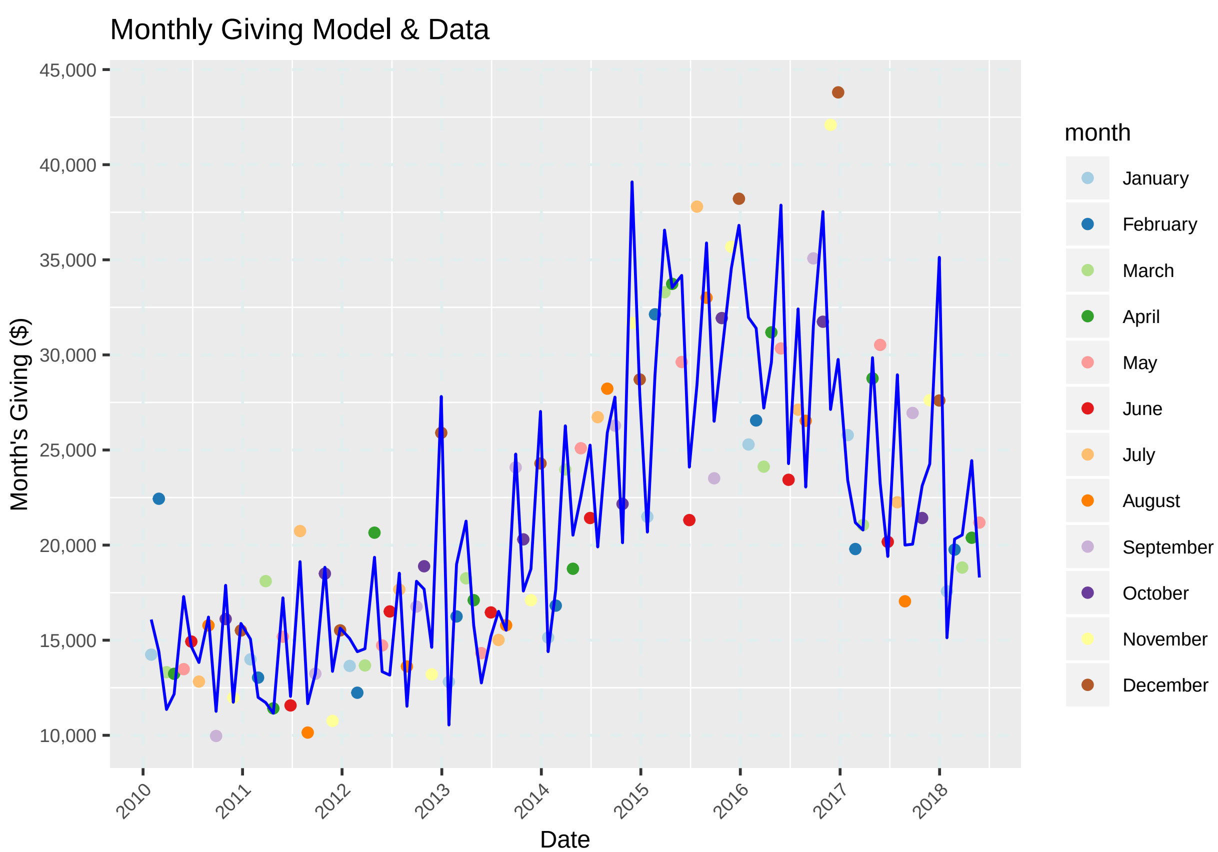
```
##
## Call:
## lm(formula = logMonthsGivingPerWeek ~ year + month, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.26331 -0.10444 -0.02756  0.08368  0.44355
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.26972    0.07373  57.911  < 2e-16 ***
## year2011          0.12964    0.06964   1.862 0.066292 .
## year2012          0.18706    0.06964   2.686 0.008771 **
## year2013          0.20607    0.06964   2.959 0.004045 **
## year2014          0.39235    0.06964   5.634 2.47e-07 ***
## year2015          0.51987    0.06964   7.465 8.43e-11 ***
## year2016          0.60149    0.06964   8.637 4.15e-13 ***
## year2017          0.54415    0.06964   7.814 1.75e-11 ***
## year2018          0.50249    0.09311   5.397 6.59e-07 ***
## monthFebruary     0.20535    0.08042   2.554 0.012535 *
## monthMarch        0.21589    0.08042   2.685 0.008802 **
```

```
## monthApril      0.16721    0.08042   2.079 0.040745 *
## monthMay        0.19020    0.08042   2.365 0.020411 *
## monthJune       0.17646    0.08338   2.116 0.037378 *
## monthJuly       0.29491    0.08338   3.537 0.000673 ***
## monthAugust     0.17706    0.08338   2.124 0.036751 *
## monthSeptember  0.27169    0.08338   3.259 0.001637 **
## monthOctober    0.22375    0.08338   2.684 0.008827 **
## monthNovember   0.31329    0.08338   3.758 0.000322 ***
## monthDecember   0.40589    0.08338   4.868 5.48e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1706 on 81 degrees of freedom
## Multiple R-squared:  0.6832, Adjusted R-squared:  0.6089
## F-statistic: 9.193 on 19 and 81 DF,  p-value: 2.571e-13
```

This fit looks reasonably good though the normal quantile plot shows slightly heavy distribution tails. High a low months are not predicted as well. The model clearly differentiates month giving as shown by the significance of the month coefficient terms (see both ANOVA table and coefficients table).

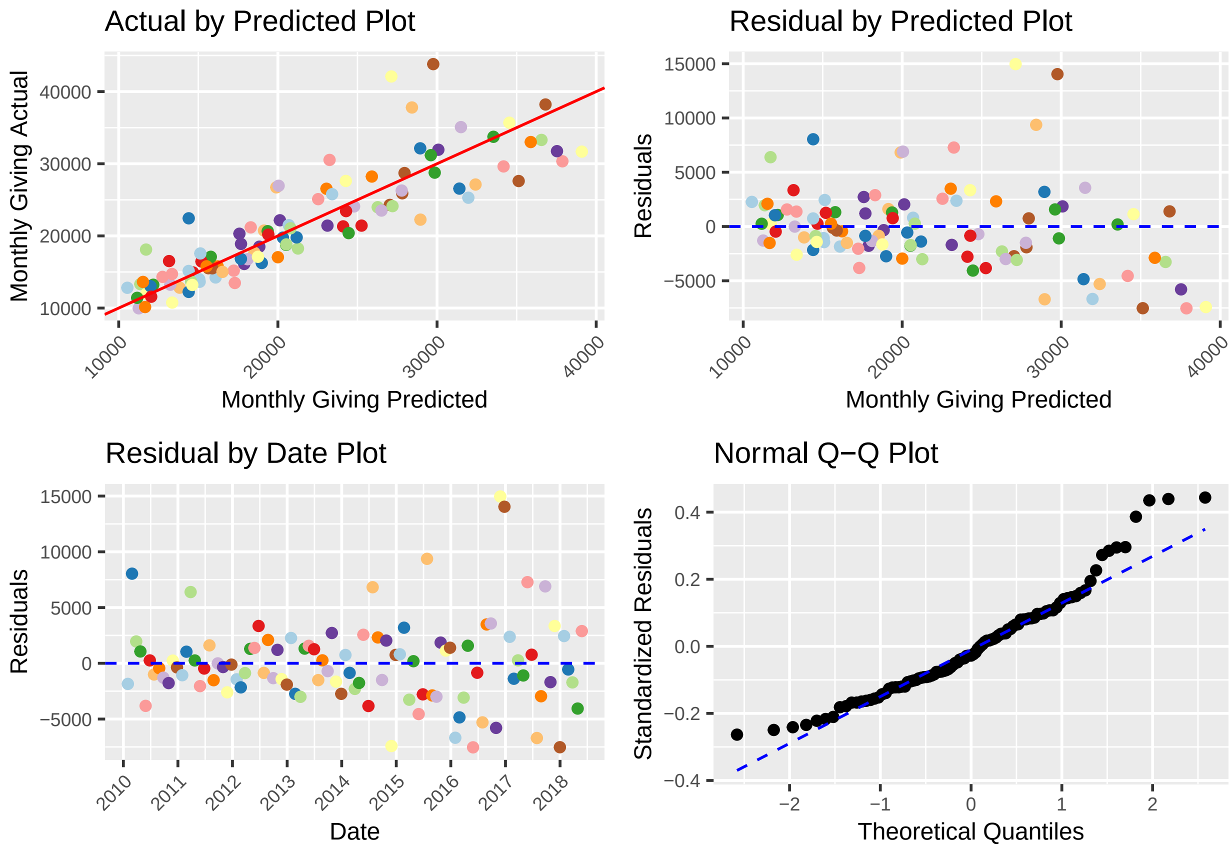
Model Predictions for Monthly Total Giving

Model results can be used to back calculate the monthly giving totals to observe the fit in that context.



Model Fit Assessment

Here are the fit assessment graphs with the fit results in the context of the monthly total giving.



The Residuals by Predicted Plot, in particular, shows the highest giving is over predicted. This could possibly be caused by including a number of other larger donations that might have been one-time donations not representative of "regular" giving. However, those obviously happen occassionally so including them causes the model to predict presuming they will happen.

The overall model root mean square error (RSME) in the modeled monthly giving is:

```
## [1] $ 3,798.87
```

This value is helpful in establishing an income variability fund to offset giving variation. The RSME is a good, non-biased estimator of the standard deviation (σ) of the distribution of residuals. The normal quantile plot above shows the majority of the residual distribution are normally distributed. While the tails (residuals at the high and low end of giving) are more common than a normal distribution, the high giving is more common than the lows.

Modeling Monthly Variation

An useful aspect of the model chosen is that it can be rearranged in a useful way for forecasting. The following discussion shows that rearrangement. Starting with the model for the natural log transformed giving per Sunday per month per giving family:

$$ln\left[\frac{Monthly\ Giving}{n_{Sundays\ in\ Month} \cdot n_{Giving\ Families}}\right] = a + b_{year} + c_{month}$$

$$\frac{Monthly\ Giving}{n_{Sundays\ in\ Month} \cdot n_{Giving\ Families}} = e^{(a+b_{year}+c_{month})}$$

Shifting the b_{year} and c_{month} coefficients as previously discussed in the "Transfer of 2013 Analysis" notebook gives:

$$\begin{aligned}\frac{Monthly\ Giving}{n_{Sundays\ in\ Month} * n_{Giving\ Families}} &= e^{\left(\left[a+\overline{b_{year}}+\overline{c_{month}}\right]+(b_{year}-\overline{b_{year}})+(c_{month}-\overline{c_{month}})\right)} \\ &= e^{\left(\left[a+\overline{b_{year}}+\overline{c_{month}}\right]+b'_{year}+c'_{month}\right)} \\ &= e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{(b'_{year})} \cdot e^{(c'_{month})}\end{aligned}$$

The $e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{(b'_{year})}$ collection of terms is the average giving per Sunday per month per family in any particular year of interest. When $b'_{year} = 0$ and $c'_{month} = 0$ the e^0 terms are 1 and the model returns that **average** month. The $e^{c'_{month}}$ terms are multipliers that provide a specific month's estimate for a given (or average) year. These are the values of the $e^{c'_{month}}$ terms:

```
mon.coef <- c(0, mod$coefficients[10:20])
names(mon.coef) <- month.name
mon.coef <- exp(mon.coef - mean(mon.coef))
mon.coef
```

```
##   January  February     March     April       May      June      July
## 0.8024041 0.9853194 0.9957573 0.9484475 0.9704993 0.9572571 1.0776346
##    August September   October  November  December
## 0.9578336 1.0529001 1.0036176 1.0976215 1.2041195
```

Notice the average of the monthly coeffiecients is very close to 1:

```
mean(mon.coef)
```

```
## [1] 1.004451
```

I'll reorder the vector of these coefficients to concide with the order of the fiscal year rather than the calendar year. The values are not changed just the order.

```
monthsInFYOrder <- c(month.name, month.name)[7:18]
mon.coef <- mon.coef[monthsInFYOrder]
mon.coef
```

```
##      July    August September   October  November  December   January
## 1.0776346 0.9578336 1.0529001 1.0036176 1.0976215 1.2041195 0.8024041
##  February     March     April       May      June
## 0.9853194 0.9957573 0.9484475 0.9704993 0.9572571
```

The $e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{(b'_{year})}$ quantity can also be determined from annual data:

$$e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{(b'_{year})} = \frac{Giving_{year}}{n_{Sundays\ in\ Yr} \cdot \overline{n}_f}$$

Where $\overline{n}_f$ is the average number of families giving over the year.
Shortening the model expression with $n_{Sundays\ in\ Month} = n_S$ and $n_{Giving\ Families} = n_F$, both monthly values, and moving them over to the right we get:

$$Monthly\ Giving = n_S * n_F \cdot e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{\left(b'_{year}\right)} \cdot e^{c'_{month}}$$

Summing both sides over some range of months:

$$\begin{aligned}\sum_{Months} Monthly\ Giving = Giving_{Total} &= \sum_{Months} \left[n_S * n_F \cdot e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{\left(b'_{year}\right)} \cdot e^{c'_{month}}\right] \\ &= e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{\left(b'_{year}\right)} \cdot \sum_{Months} \left[n_S * n_F \cdot e^{c'_{month}}\right]\end{aligned}$$

substituting for the $e^{\left(a+\overline{b_{year}}+\overline{c_{month}}\right)} \cdot e^{\left(b'_{year}\right)}$ terms gives:

$$\begin{aligned}Giving_{Total} &= \frac{Giving_{year}}{n_{Sundays\ in\ Yr} \cdot \overline{n}_f} \cdot \sum_{Months} \left[n_S * n_F \cdot e^{c'_{month}}\right] \\ &= \frac{Giving_{year}}{\overline{n}_f} \cdot \sum_{Months} \left[\left(\frac{n_S}{n_{Sundays\ in\ Yr}}\right) \cdot e^{c'_{month}} * n_F\right]\end{aligned}$$

Then, if n_F, the number of giving families in each month, is taken to be a constant average number, $\overline{n}_f$ that term can be pulled out of the sum and cancels.

$$Giving_{Total} = Giving_{year} \cdot \sum_{Months} \left[\left(\frac{n_S}{n_{Sundays\ in\ Yr}}\right) \cdot e^{c'_{month}}\right]$$

It is this collection of terms inside the sum that provides the number-of-Sundays-in-each-month correction to the $e^{c'_{month}}$ coefficients regressed in the model.
The following function calculates the vector of $n_S/n_{Sundays\ in\ Yr}$ and shows the result for 2019:

```
Sunday.Corrections <- function(year) {
# This function get a list of the Sundays in the fiscal year from July 1, year to June 30, year+1
  getSundaysInFY <- function(yr) {
    startDate <- paste0(yr,"-07-01")
    endDate <- paste0(yr+1,"-06-30")
    dates <- seq(as.Date(startDate),as.Date(endDate), by = "day")
    dates[weekdays(dates) == "Sunday"]
  }
  SundaysInYr <- getSundaysInFY(year)  # List of dates of Sundays in the year
  numSundaysInFY <- length(SundaysInYr) # Number of Sundays in the year: 52 or 53
  x <- strftime(SundaysInYr,"%m")  # Month listed once for each Sunday it contains
  numSundaysInMonths <- table(factor(x,unique(x), ordered = TRUE))  # Sundays in each month.
  names(numSundaysInMonths) <- c(month.name,month.name)[7:18]  # Convert names to month Names
  Corrections <- numSundaysInMonths / numSundaysInFY
  Corrections
}
Sunday.Corrections(2019)
```

```
##       July     August  September    October   November   December
## 0.07692308 0.07692308 0.09615385 0.07692308 0.07692308 0.09615385
##    January   February      March      April        May       June
## 0.07692308 0.07692308 0.09615385 0.07692308 0.09615385 0.07692308
```

If each month had $52/12 = 4\frac{1}{3}$ Sundays and there were 52 Sundays in the year, this vector would be all 0.083333. Instead both the monthly and annual number of Sundays vary leading to the modeled income variation.

Making Projections

We can now calculate the vector of $(n_S/n_{Sundays\ in\ Yr}) \cdot e^{c'_{month}}$ products for the fiscal year beginning in July, 2019:

```
Giving.Fractions <- Sunday.Corrections(2019)*mon.coef
Giving.Fractions
```

```
##       July     August  September    October   November   December
## 0.08289497 0.07367951 0.10124039 0.07720135 0.08443242 0.11578072
##    January   February      March      April        May       June
## 0.06172339 0.07579380 0.09574589 0.07295750 0.09331724 0.07363516
```

Notice this vector sums to nearly 1, but given the inputs (model regressions and a fiscal year's distribution of Sundays) it is not exactly 1.

```
sum(Giving.Fractions)
```

```
## [1] 1.008402
```

To keep the months so they sum exactly to the total we can normalize them:

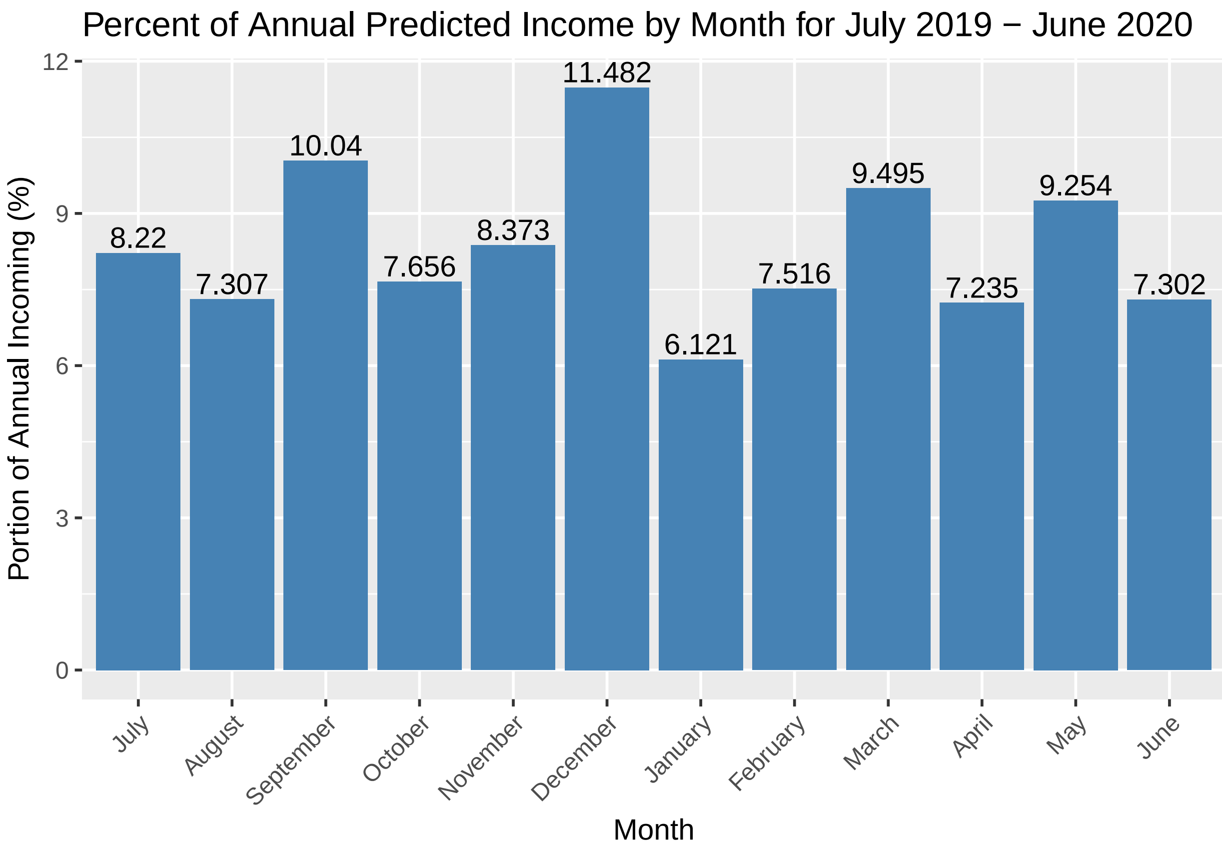
```
Giving.Fractions <- Giving.Fractions / sum(Giving.Fractions)
Giving.Fractions
```

```
##       July     August  September    October   November   December
## 0.08220426 0.07306559 0.10039682 0.07655809 0.08372890 0.11481599
##    January   February      March      April        May       June
## 0.06120909 0.07516226 0.09494811 0.07234959 0.09253969 0.07302161
```

```
cat("Sum = ", sum(Giving.Fractions))
```

```
## Sum =  1
```

Finally, a plot showing the relative magnitudes of the predicted monthly giving for the fiscal year beginning in July, 2019.



This projection assumes that 1) giving follows patterns true between 2010 and 2018, that 2) the number of giving families is constant over the projection period and 3) the giving families giving practices are stable.

Another potential use of the modeling result is to use the January through June income from the preceeding fiscal year to project the following fiscal year. A sum of the January through June portions gives:

```
previous.year <- Sunday.Corrections(2018)*mon.coef
previous.year <- previous.year / sum(previous.year)
previous.year[7:12]
```

```
##    January   February      March      April        May       June
## 0.05999136 0.07366693 0.09305914 0.07091022 0.07255891 0.08946108
```

```
cat("Sum = ", sum(previous.year[7:12]))
```

```
## Sum =  0.4596476
```

So, an estimate of FY2019 giving is:

$$FY2019\ Giving = \frac{\sum_{Jan-Jun}(2019\ Giving)}{0.46} \cdot \frac{\overline{n}_{Giving\ Families\ in\ FY2019}}{\overline{n}_{Giving\ Families\ in\ Jan\text{-}Jun\ 2019}}$$

where $\overline{n}$'s are the average number for the period indicated by subscripts.

Again, this assumes giving follows historic patterns and giving per family is the same as in Jan-Jun 2019.

Note: this prediction would be impacted by giving anomolies that impact the *2019 Giving* quantity like a significant amount of missing checks if they didn't make it in my the end of June.

Modeling Criticism

One of the issues this modeling approach has is that regression assumes that factors are not correlated. For example, giving one month does not affect giving the next month. This is almost certainly not perfectly true. For example, the model shows a repeated December to January drop in giving. It could be that givers sometimes choose to move up a gift they planned to make in January to December for tax benefits. Thus, it might be common that high giving one month could lead to lower giving the next month. This leads to a statistical correlation between the year term estimates. Separate from those examples, a regular giver's giving one month is highly likely to predict their giving the next month, similarly for the collection of regular givers'. On the one hand, that predictability is what the model is attempting to capture. On the other hand, it also provides the correlation between factors. For example, a regular giver's donation that occurs bi-weekly could land within consecutive months based on paydays. Regular givers that are paid and donate monthly are less likely to have the same "correlation". For this reason, regression is perhaps not as technically appropriate as time series analysis. Perhaps this just adds to George Box's point that all models are wrong, some are useful.