Calculating and Forecasting HVAC System Energy Consumption

SYS 5581: Timseseries and Forecasting

Caleb Neale

Spring 2021

Abstract

Commercial Heating, Ventilation, and Cooling (HVAC) systems contribute significantly to US energy consumption, but the ability to forecast and model HVAC energy usage can be difficult without dedicated metering. This in turn makes simulation studies for the purpose of energy efficiency optimization difficult. This study aims to establish a model for effectively calculating HVAC energy usage without direct metering by leveraging engineering equations based on collected data on system dynamics.

Using calculated energy consumption, a forecasting model will be generated to predict future energy consumption. The generated model suggests that a dynamic regression model leveraging local temperature with ARIMA errors is an effective predictor of calculated energy use.

However, limitations on the accuracy of calculated energy consumption were found and further tuning of the parameters used in the equations would be necessary to achieve calculations useful enough for effective simulation and energy efficiency optimization. Additionally, longer data histories would allow for the proper modeling of yearly seasonal energy consumption changes where this model likely falls short.

Introduction

In UVA's LinkLab, as a part of the Living Link Lab Program, there is a significant amount of environmental, occupancy data, and HVAC system operational data available for analysis. This presents an opportunity for a detailed case study of the performance of an HVAC system among multiple metrics outside of just temperature and humidity with the intention of improving HVAC control systems' ability to maintain occupant comfort with reduced energy consumption. Investigation of multiple metrics of occupant comfort, whether a given room even has occupants which require comfort, various metrics of system operation, and energy consumption consumption (actual and calculated) has the potential to produce a environmental model which may aid in the development of an improved policy for the HVAC system control problem.

Considering HVAC usage accounts for 30% of total commercial building energy consumption Goetzler et al. (2017), there is significant environmental and economic incentive to reducing the energy load of HVAC systems both for regulators and commercial operators. This same commissioned report cites "Technology Enhancements for Current Systems" as one of four groups of high priority technology options, with "Advanced HVAC Sensors" as the top ranked technology within this category at an estimated Technical Energy Savings Potential (Quadrillion BTU/yr.) of 0.63, lending particular credence to the idea that advanced sensing combined with more efficient control could be a significant contributor to reduced HVAC system burden on energy resources.

This project will investigate the first phase of determining if it possible to create a more efficient control policy for HVAC systems by developing a model of energy consumption calculated from system operation metrics, validating this model with a limited amount of actual collected energy usage data, and attempting to forecast energy demand with this data.

The data and the data-generating process

The data which will be analyzed consists of data exported from an internal UVA server which hosts all HVAC system data related to Olsson Hall and the 2nd floor Link Lab. Additionally, local weather data on humidity, atmospheric pressure, and local temperature data from the KVACHARL114 station attached to Olsson hall will be obtained as potential predictors of energy consumption. Finally, although energy data from the individual HVAC units under consideration is not available, building-wide energy consumption data is available for Olsson Hall.

The Data

Air Handling Unit Level Data

Index: Time (30min)

Key: HVAC Air Handling Unit (AHU2E or AHU2W)

Values of Interest: - SAT - Output Air Temperature (deg C) - MAT - Input Air Temperature (deg C) - SAF - Air Supply Flow (CFM) - SAFSP - Air Supply Flow Set point (CFM) - SAEnthalpy - Output air enthalpy (calculated, kW) - MAEnthalpy - Input air enthalpy (calculated, kW) - totalHeat - Change in enthalpy (calculated, kW) - fanEnergy - Energy used by fan (calculated, kW)

```
head(equip)
```

```
## # A tsibble: 6 x 18 [30m] <UTC>
## # Key:       equipment [1]
##   equipment time                 MAT RFNVFDO   SAF SATSP  RFNC SFNVFDO   SAT
##   <chr>     <dttm>             <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
## 1 AHU2E     2020-07-24 09:30:00  76.8    20   4208.  53      NA    53.8  52.6
## 2 AHU2E     2020-07-24 10:00:00  77.7    20   4222.  53      NA    53.8  52.8
## 3 AHU2E     2020-07-24 10:30:00  77.9    20.5 3904.  53.3    NA    54.8  56.6
## 4 AHU2E     2020-07-24 11:00:00  78.7    20.2 4522.  53      NA    52.5  56.2
## 5 AHU2E     2020-07-24 11:30:00  79.1    23.4 4577.  53      NA    53.1  54.6
## 6 AHU2E     2020-07-24 12:00:00  79.9    22.8 4351.  53      NA    53.1  51.6
## # ... with 9 more variables: NA <dbl>, SFNSTS <dbl>, SFNC <dbl>, RFNSTS <dbl>,
## #   SFNVFDALA <dbl>, SAEnthalpy <dbl>, MAEnthalpy <dbl>, totalHeat <dbl>,
## #   fanEnergy <dbl>
```

Room Level Data

Index: Time (30min)

Key: Room in Olsson Hall

Values: - SAF - Supply Air Flow (CFM) - outputTemp - Reheated air temperature from room level treatment (deg C) - ZNT - Room temperature setpoint (deg C) - SAFSP - Supply air flow setpoint (CFM) - equipment - AHU which serves given room (AHU2E or AHU2W) - inputTemp - Air temperature from AHU given to room for reheat (deg C) - inputEnthalpy - Energy of input air (calculated, kW) - outputEnthalpy - Energy of output air (calculated, kW) - totalHeat - Energy used in reheat (calculated, kW)

```
head(rooms)
```

```
## # A tsibble: 6 x 11 [30m] <UTC>
## # Key:       room [1]
##   room  time                outputTemp   SAF SAFSP   ZNT equipment inputTemp
##   <chr> <dttm>                   <dbl> <dbl> <dbl> <dbl> <chr>         <dbl>
## 1 2011  2020-07-24 09:30:00         NA 1091. 1096.  74.6 AHU2W          52.5
## 2 2011  2020-07-24 10:00:00         NA  595.  590.  74.6 AHU2W          53.2
## 3 2011  2020-07-24 10:30:00         NA  520.  510.  74.9 AHU2W          62.0
## 4 2011  2020-07-24 11:00:00         NA  949.  957.  75.2 AHU2W          59.7
## 5 2011  2020-07-24 11:30:00         NA 1419. 1417.  74.9 AHU2W          56.0
## 6 2011  2020-07-24 12:00:00         NA 1100. 1110.  74.6 AHU2W          51.5
## # ... with 3 more variables: inputEnthalpy <dbl>, outputEnthalpy <dbl>,
## #   totalHeat <dbl>
```

Weather Data

Index: Time (5min)

Key: StationID, KVACHARL114

Values: - humidityAvg - Average relative humidity over 5 min interval (%) - tempAvg - Average temperature over 5 min interval (deg C) - pressureMax - Maximum air pressure over 5 min interval (hPa)

```
head(weather)
```

```
## # A tsibble: 6 x 5 [1m] <?>
## # Key:       stationID [1]
##   humidityAvg tempAvg obsTimeLocal        pressureMax stationID
##         <int>   <int> <dttm>                    <dbl> <chr>
## 1          57       7 2020-01-01 00:04:00       1009. KVACHARL114
## 2          57       7 2020-01-01 00:09:00       1008. KVACHARL114
## 3          58       7 2020-01-01 00:14:00       1008. KVACHARL114
## 4          58       7 2020-01-01 00:19:00       1008. KVACHARL114
## 5          58       7 2020-01-01 00:24:00       1009. KVACHARL114
## 6          58       7 2020-01-01 00:29:00       1009. KVACHARL114
```

Aggregated Energy

Index: Time (30min)

Key: HVAC Air Handling Unit (AHU2E or AHU2W)

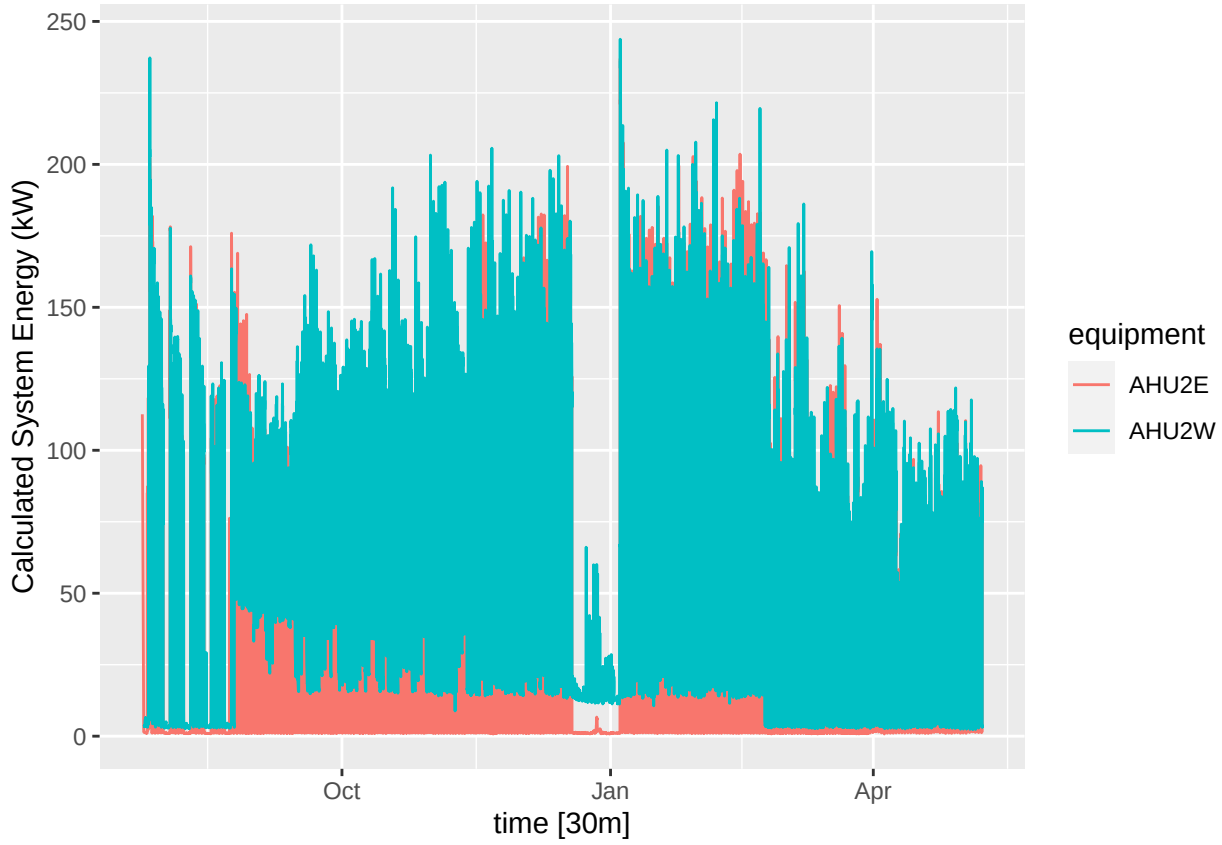
Values: - totalHeat.AHU - Energy used in AHU level treatment (calculated, kW) - fanEnergy - Energy used by fan (calculated, kW) - totalHeat.room - Energy used in room level treatment (calculated, kW) - totalEnergy - sum of all three above energy categories (calculated, kW)

```
head(totalEnergyDF %>% filter(!is.na(totalEnergy)))
```

```
## # A tsibble: 6 x 7 [30m] <UTC>
## # Key:       equipment [1]
##   totalHeat.AHU fanEnergy time                equipment totalHeat.room totalHeat
##           <dbl>     <dbl> <dttm>              <chr>              <dbl>     <dbl>
## 1        74.3      27.1   2020-07-24 15:00:00 AHU2E             11.2      85.5
## 2         0.179     0.691 2020-07-25 02:30:00 AHU2E              0.618     0.796
```

```
## 3         0.163     0.691 2020-07-25 03:00:00 AHU2E              0.629     0.792
## 4         0.150     0.691 2020-07-25 03:30:00 AHU2E              0.638     0.788
## 5         0.138     0.691 2020-07-25 04:00:00 AHU2E              0.621     0.759
## 6         0.126     0.691 2020-07-25 04:30:00 AHU2E              0.616     0.742
## # ... with 1 more variable: totalEnergy <dbl>
```

```
autoplot(totalEnergyDF, .vars = totalEnergy) + ylab("Calculated System Energy (kW)")
```



Building Energy Index: Time (30min)

Key: N/A

Values: - chilledWater - Energy used in cooling/chilled water for the building (kBTU/h) - electricity - Electricity used by building (kW) - heating - Energy used bilding-wide heating (kBTU/h)

```
head(buildingEnergy)
```

```
## # A tsibble: 6 x 4 [15m] <?>
##   time                chilledWater electricity heating
##   <dttm>                     <dbl>       <dbl>   <dbl>
## 1 2020-01-01 00:00:00         14.0        26.7    44.4
## 2 2020-01-01 00:15:00         17.5        26      64.8
## 3 2020-01-01 00:30:00         16.1        26      71.6
## 4 2020-01-01 00:45:00          0          27      80.8
## 5 2020-01-01 01:00:00          0          26      78.5
## 6 2020-01-01 01:15:00          0          26      70.5
```

Data Generating Process

System Description and Terminology

The HVAC system under consideration is a VAV (variable air volume) system which consists of two air handling units (AHU) and VAV boxes. Initial treatment (heating/cooling) for air in the building is done at the air handling unit. VAV systems manage the temperature of the different rooms in the building by providing specific volumes of air to each room using VAV boxes. At the VAV box, air may be reheated (after having been initially treated at the AHU) to provide the room with the correct temperature of air needed to maintain the room at its given setpoint. The simplified model used here of the AHU consists of hot and cold water coils for heating/cooling, a return air fan, and a supply air fan. Each VAV box consists of a vent regulating air volume and a heating coil.

Definitions of Air Types

Return air is defined as air which is being returned from the conditioned environment. In this system, return air is mixed with outdoor air to produce mixed air. Mixed air serves as the input to the conditioning system at the AHU level. Once the air is treated (heating/cooling, humidity removal) by the AHU, it is output as supply air. This supply air is then provided to each VAV box for additional heating as needed, producing final supply air. As this lack of distinction between these two versions of "supply air" may be confusing, this analysis will adopt the terms "input air" and "output air" when discussing both the VAV and AHU treatment processes.

Air Supply Flow

Air supply flow is measured at the VAV box level and aggregated to determine AHU air volumes.

Enthalpy Change

As treatment can occur at both the AHU and VAV box, enthaply change (change in energy due to heating/cooling and humidity removal) must be calculated at both stages and then aggregated to find total enthalpy change at a given time.

Weather Data

Weather data was collected from KVACHARL114, a weather station located at Olsson Hall. Data can be previewd at https://www.wunderground.com/dashboard/pws/KVACHARL114.

Plan for Analysis

The primary goal of this analysis is to generate an effective model of HVAC system energy consumption over time. To do this, a model of HVAC energy consumption based on system dynamics is needed as historical energy consumption data is not available at the HVAC unit level, only for the building in aggregate. Modeling of the system can thus be divided into three steps:

- Calculate estimated HVAC energy consumption using historical data on operational metrics
- Validate estimated energy consumption with historical, building-wide energy consumption data; update parameters and assumptions as necessary
- Create a statistical model of energy consumption in the HVAC system and generate a forecast with this model

Calculate Energy Usage from HVAC Operational Metrics

The primary drivers of energy use in this HVAC system come from treatment at the AHU, treatment at the VAV box, and fan operation in the AHU. Calculations for treatment at the AHU and VAV box level leverage the same equations, and fan equations can be leveraged to estimate fan energy usage.

Air Treatment

The change in energy due to air treatment (at either AHU or VAV) can be divided into latent and sensible heat. Changes in latent heat arise from humidity added or removed from the air whereas sensible heat is affected by changes in air temperature. Enthalpy change accounts for both latent and sensible heat as is calculated the climateeng package ("Chrras/Climateeng Documentation," n.d.). Equation 1, below, can then be used to determine total heat by combining calculated enthalpy change with collected data on air volume flow.

Equation 1, Latent and Sensible Heat ("Cooling and Heating Equations," n.d.) h_t = *q*dh

where:

h_t = total heat (kW)
q = air volume flow (m3/s)
= density of air (1.202 kg/m3)
dh = enthalpy difference (kJ/kg)

```
load("data\\preCalcData.RData")
```

Load data

```
library(climateeng)

#Define constants
RHconstant = 0.010
density = 1.202
CFMtoM_SConverstionFactor = 0.00047194745

# Calculate AHU level enthalpy
equip$SAEnthalpy = enthalpy(equip$SAT, RHconstant)
equip$MAEnthalpy = enthalpy(equip$MAT, RHconstant)
equip$totalHeat = abs(equip$SAEnthalpy -equip$MAEnthalpy)*CFMtoM_SConverstionFactor*equip$SAF*density
```

Calculate Enthalpy for Mixed and supply Air, then total heat

```
# create table mapping room to equpment from system documentation
AHU_2E <- c(241, 243, 245, 247, 249, 251, 253, 257, 255, 259, 263, 261, 240, "C244", 244, 260, 213, 217
```

```
AHU_2W <- c(269, 267, 265, 273, 271, 275, 277, 279, 281, 283, 285, 274, 286, 204, 208, 272, 270, "C260"

rooms_tbl <- rbind(tibble('room' = AHU_2E, 'equipment' = "AHU2E"), tibble('room' = AHU_2W, 'equipment' =

# extract relevant information from room coding
rooms$cleaned_room = str_extract(rooms$room, "C\\d{3}|^\\d{3}")

# get relevant equipment for each room
rooms = left_join(rooms, rooms_tbl, by=c("cleaned_room" = "room"))

rooms %>% filter(!is.na(room)) -> rooms

rooms = rooms %>% select(-cleaned_room) -> rooms

# get input air temperature for each room
rel = equip %>% select(SAT)
rooms = left_join(rooms, rel, by=c("equipment", "time"))

# rename columns for clarity
colnames(rooms)[3] = "outputTemp"
colnames(rooms)[8] = "inputTemp"
```

Get input air temp for each room using AHU output temp

```
rooms$inputEnthalpy = enthalpy(rooms$inputTemp, RHconstant)
rooms$outputEnthalpy = enthalpy(rooms$outputTemp, RHconstant)
rooms$totalHeat = abs(rooms$inputEnthalpy - rooms$outputEnthalpy)*CFMtoM_SConverstionFactor*rooms$SAF*d
```

Calculate energy consumption from room level treatment

Fans

Power used by a fan can be calculated using equation 2, below. Pressure set to a constant typical of commercial HVAC fans, fan efficiency will be assumed to be 60%, and air volume is measured.

Equation 2, Power Used by Fan ("Fans - Efficiency and Power Consumption," n.d.) P = dp*q/mu

where:

mu = fan efficiency (values between 0 - 1)
dp = total pressure (Pa)
q = air volume delivered by the fan (m3/s)
P = power used by the fan (W, Nm/s)

```
# Set constants
fanEfficiencyCoef = 0.6
```

```
pressure = 4000
Wto_kW =1000

# Calculate energy (2* for two fans)
equip$fanEnergy = 2*equip$SAF*CFMtoM_SConverstionFactor*pressure/fanEfficiencyCoef/Wto_kW
```

Fan Energy

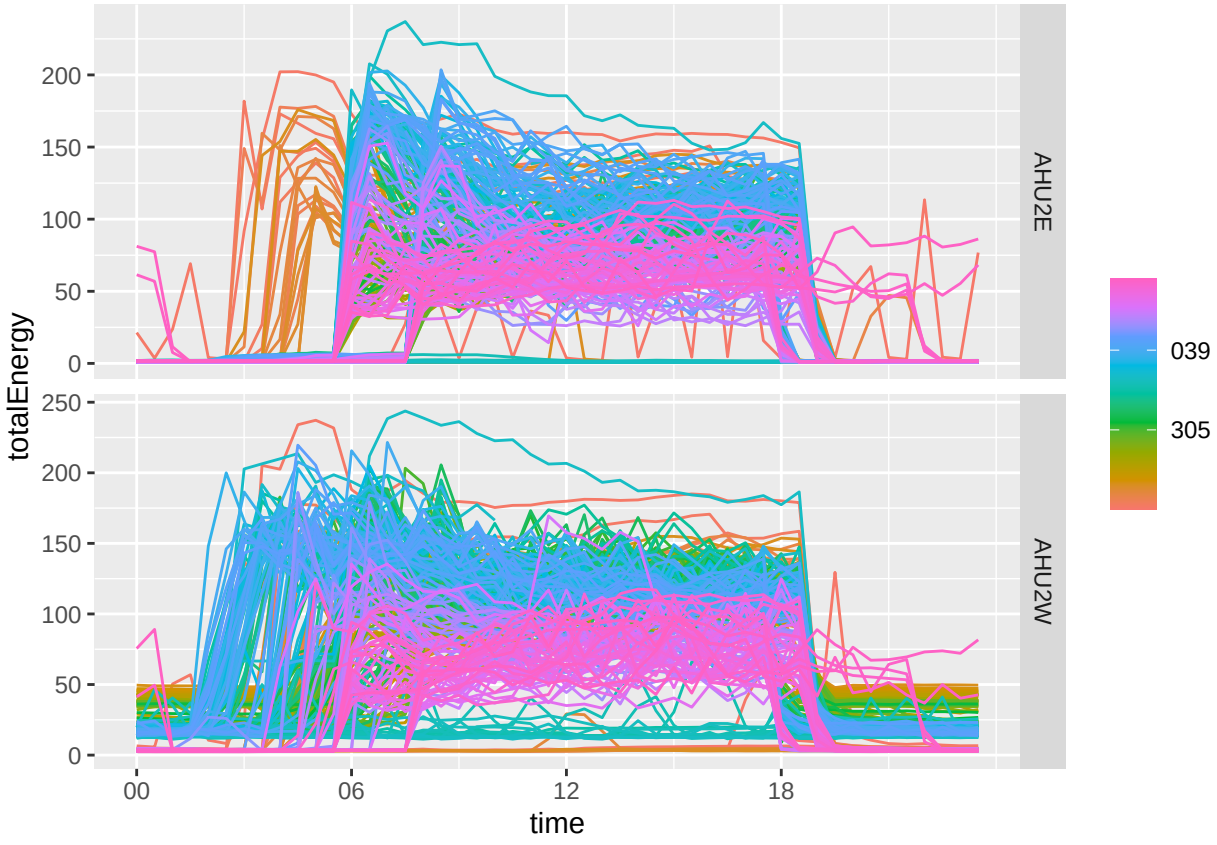
Create tsibble with room and AHU heat and Fan Energy

```
rooms %>% group_by(equipment) %>% summarise(totalHeat = sum(totalHeat)) -> roomAHUHeat

equip %>% select(totalHeat, fanEnergy) %>% inner_join(roomAHUHeat, by=c("time", "equipment"), suffix = c

totalEnergyDF$totalHeat = totalEnergyDF$totalHeat.AHU + totalEnergyDF$totalHeat.room
totalEnergyDF$totalEnergy = totalEnergyDF$totalHeat + totalEnergyDF$fanEnergy
```

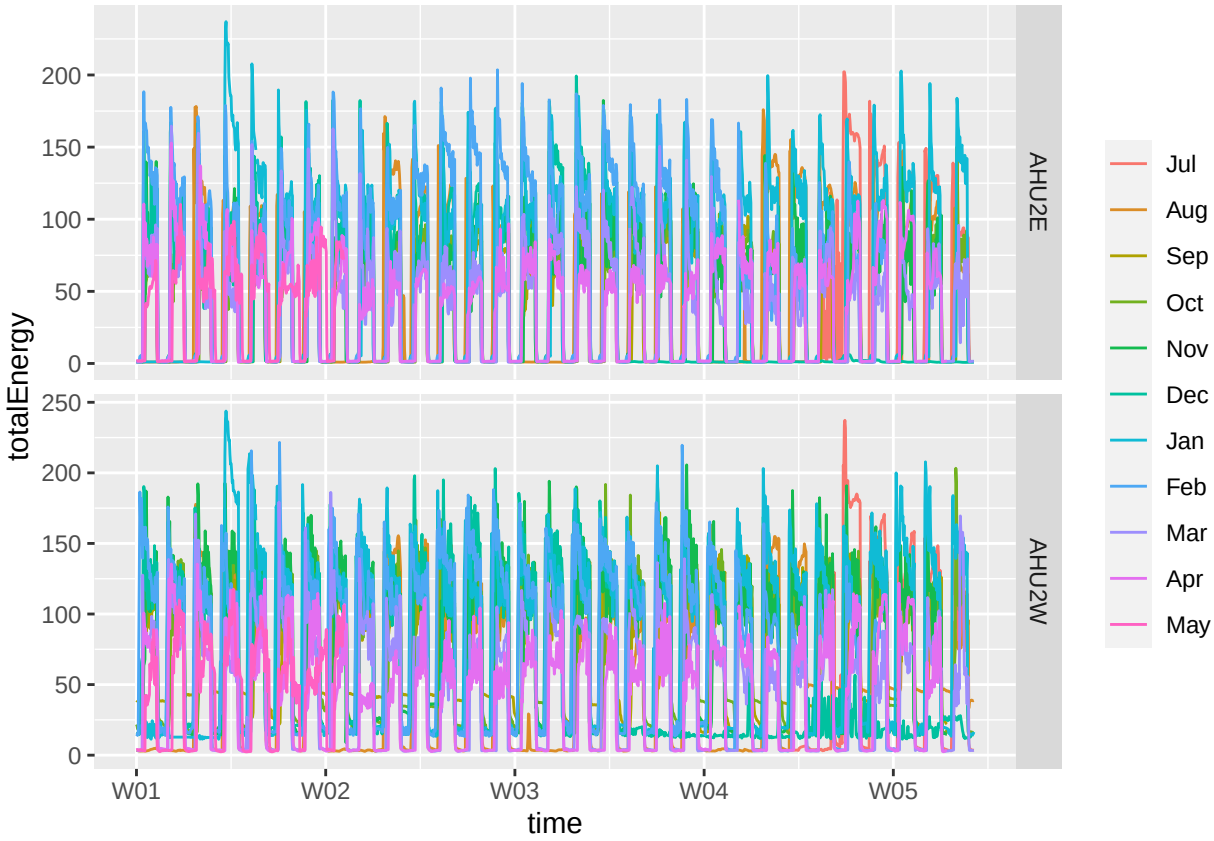
Seasonality of Energy Consumption

```
totalEnergyDF = fill_gaps(totalEnergyDF)
gg_season(totalEnergyDF, totalEnergy, period = "week")
```

```
gg_season(totalEnergyDF, totalEnergy, period = "day")
```



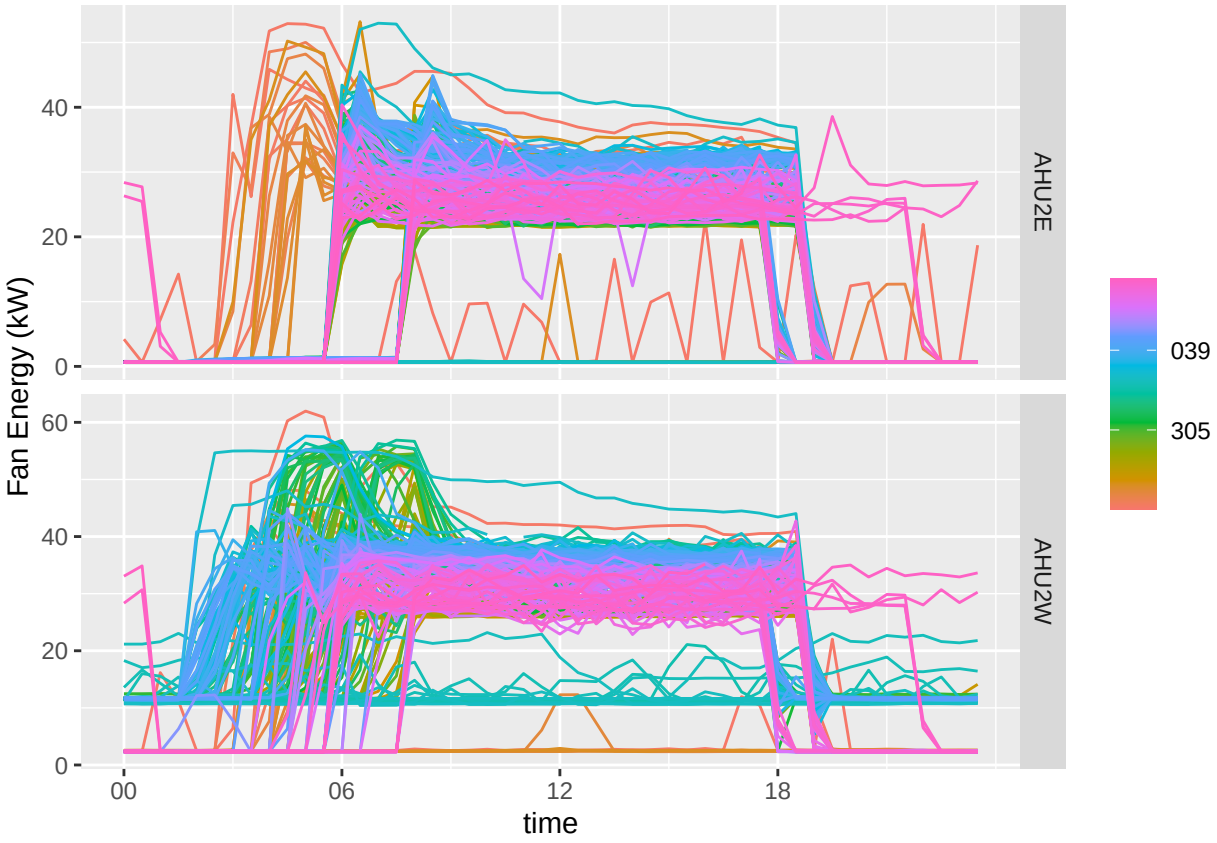

```
gg_season(totalEnergyDF, totalEnergy, period = "month")
```



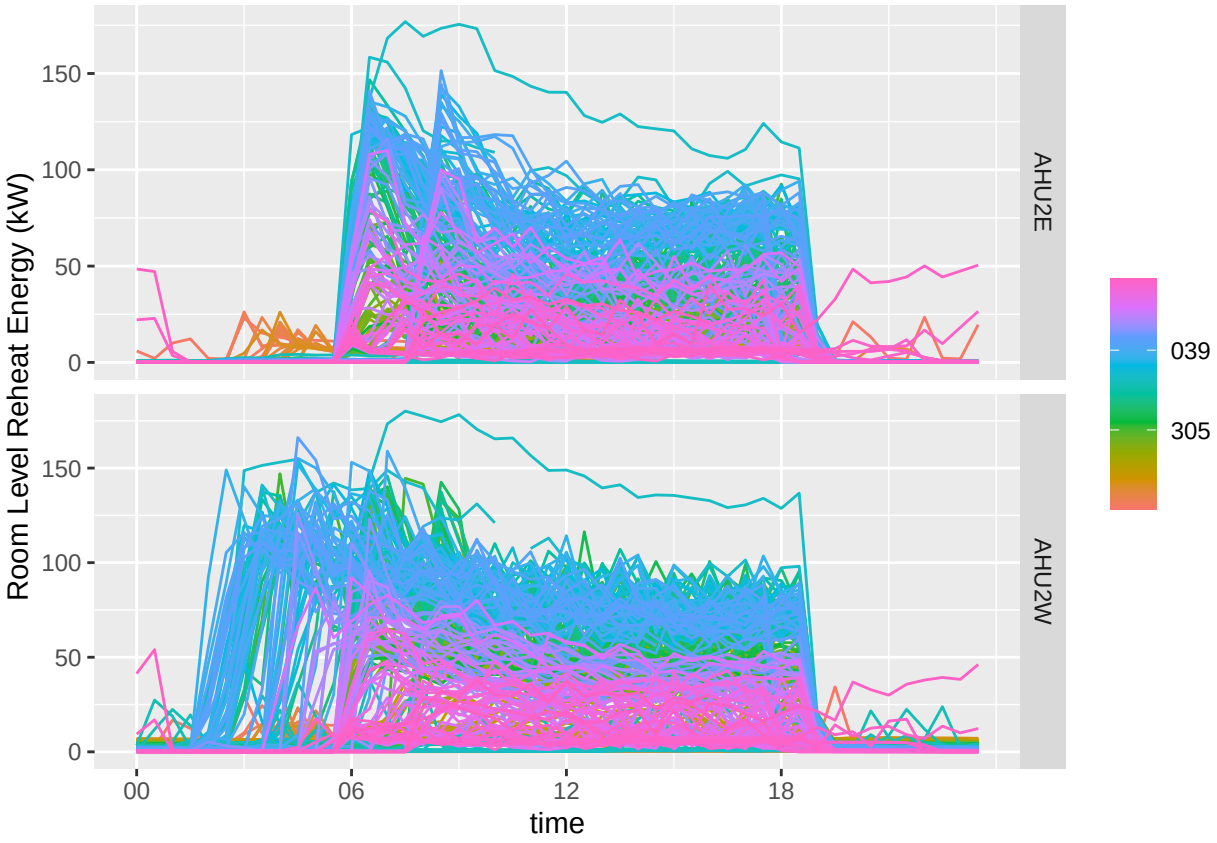
Energy consumption has a clear daily cycle where the system is off over night then turns on at a time early enough in the morning to ensure that the internal temperature in the building meets the setpoint by the time occupants arrive. The shifting start times throughout the year as seen on the daily seasonal plot are a result of this control mechanism which predicts how long the system will need to arrive at the setpoint given the current internal temperature and begins treatment accordingly. During colder/warmer seasons, internal temperature drifts farther from the setpoint overnight and thus takes longer to return to the setpoint.

AHU2W appears to have a lower bound of energy consumption above that of AHU2E. Investigating the components of totalEnergy reveals that this lower bound comes from increased nighttime fan energy and AHU level energy consumption compared to AHU2E.

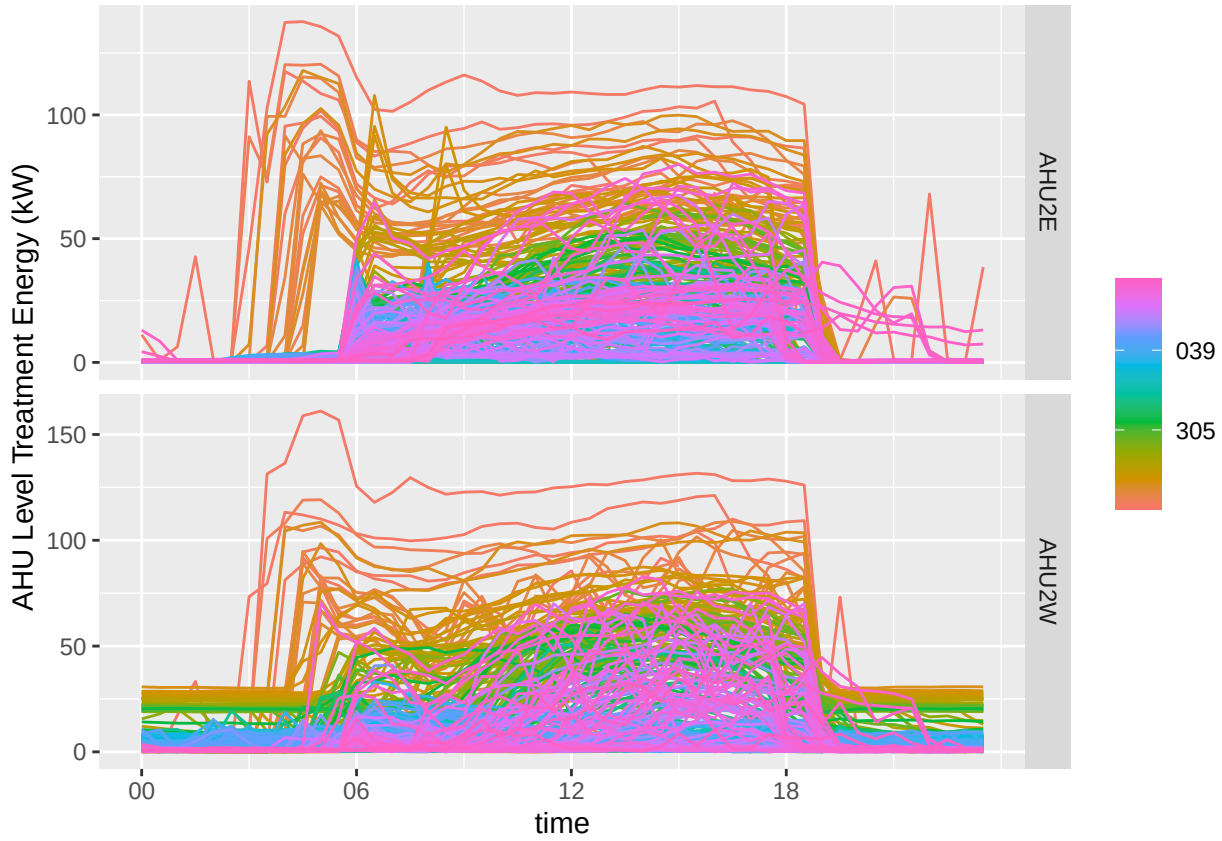
```
gg_season(fill_gaps(equip), fanEnergy, period = "day") + ylab("Fan Energy (kW)")
```




```
gg_season(fill_gaps(roomAHUHeat), totalHeat, period = "day") + ylab("Room Level Reheat Energy (kW)")
```



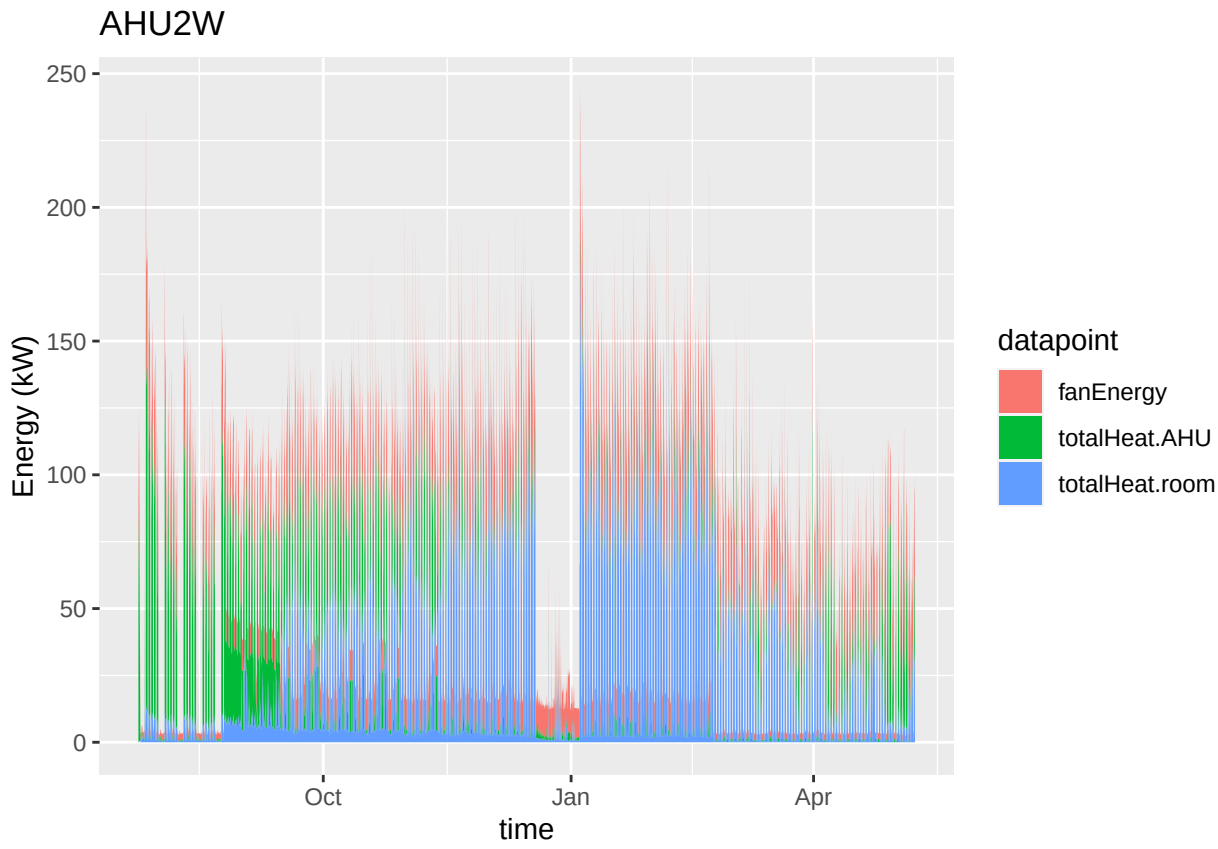

```
gg_season(fill_gaps(equip), totalHeat, period = "day") + ylab("AHU Level Treatment Energy (kW)")
```



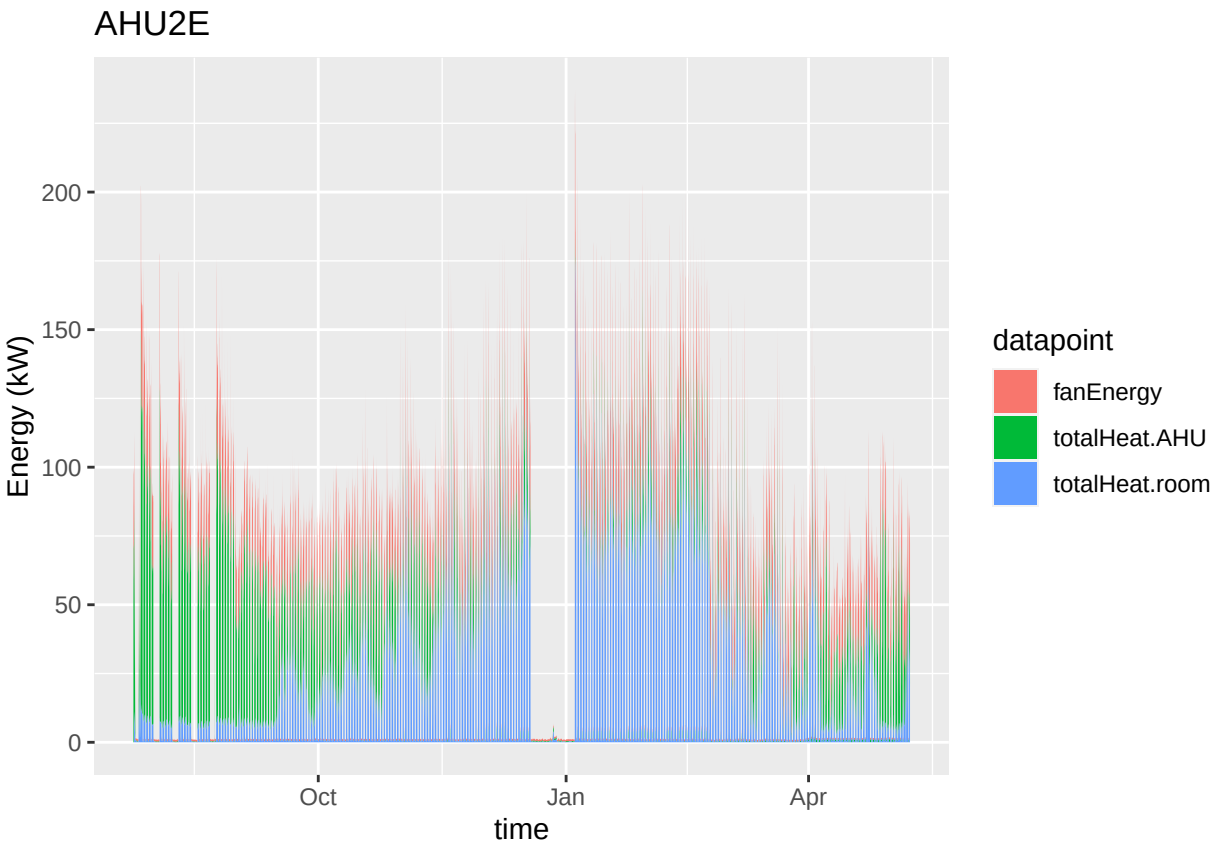
Breaking down energy consumption in this manner also reveals that fan energy consumption follows a daily cycle but is not as significantly influenced by time of year as room or AHU treatment consumption. Interestingly as well, though both AHU and room energy consumption show a yearly trend, they appear to be nearly inverse. Room consumption is highest on blue days and lowest on pink, while AHU consumption appears to be highest on orange and lowest on blue.

Investigate trends in relative energy consumption

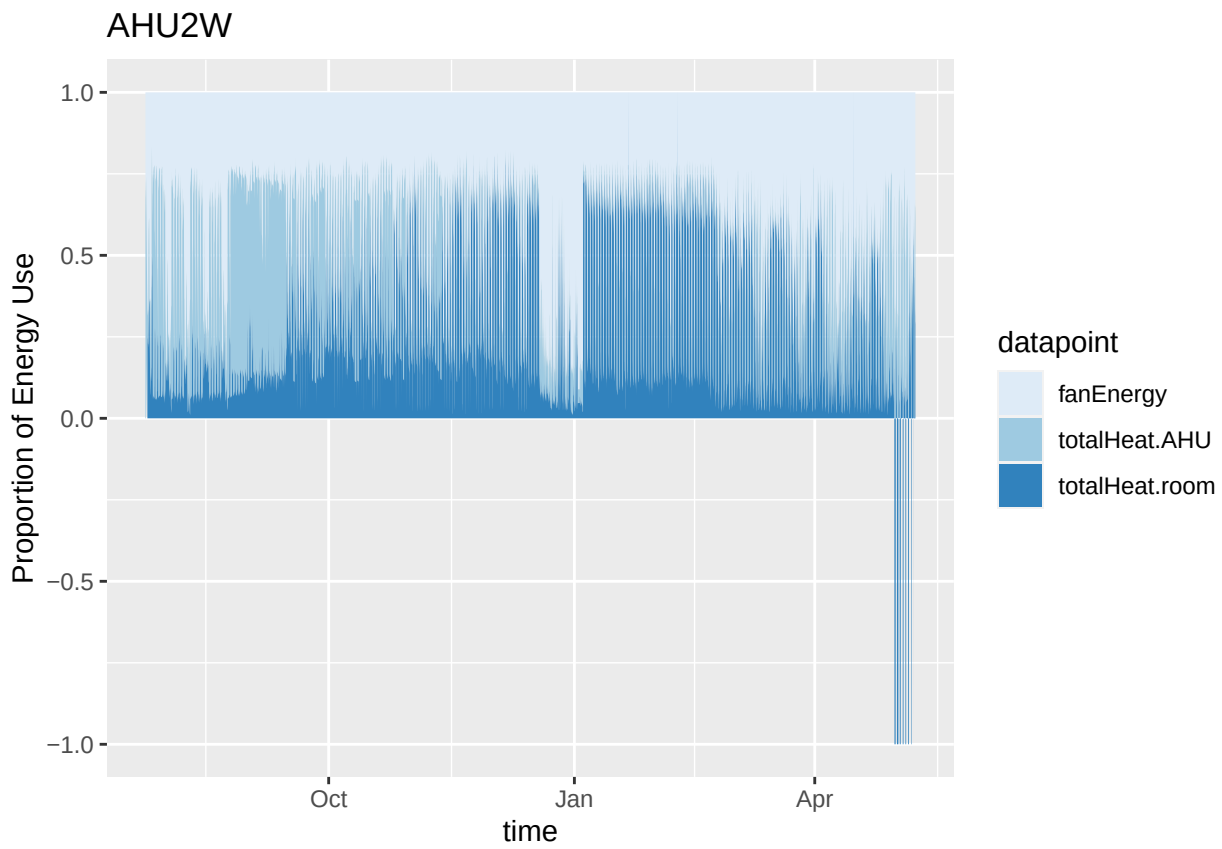
```
# get longform data for plotting
totalEnergyDF %>% select(-totalHeat, -totalEnergy) %>% filter(equipment == "AHU2E") %>% pivot_longer(col

totalEnergyDF %>% select(-totalHeat, -totalEnergy) %>% filter(equipment == "AHU2W") %>% pivot_longer(col

# stacked area plots
ggplot(data = longAHU2W, aes(x=time, y=value, fill=datapoint)) + geom_area() +ylab("Energy (kW)") + ggti
```



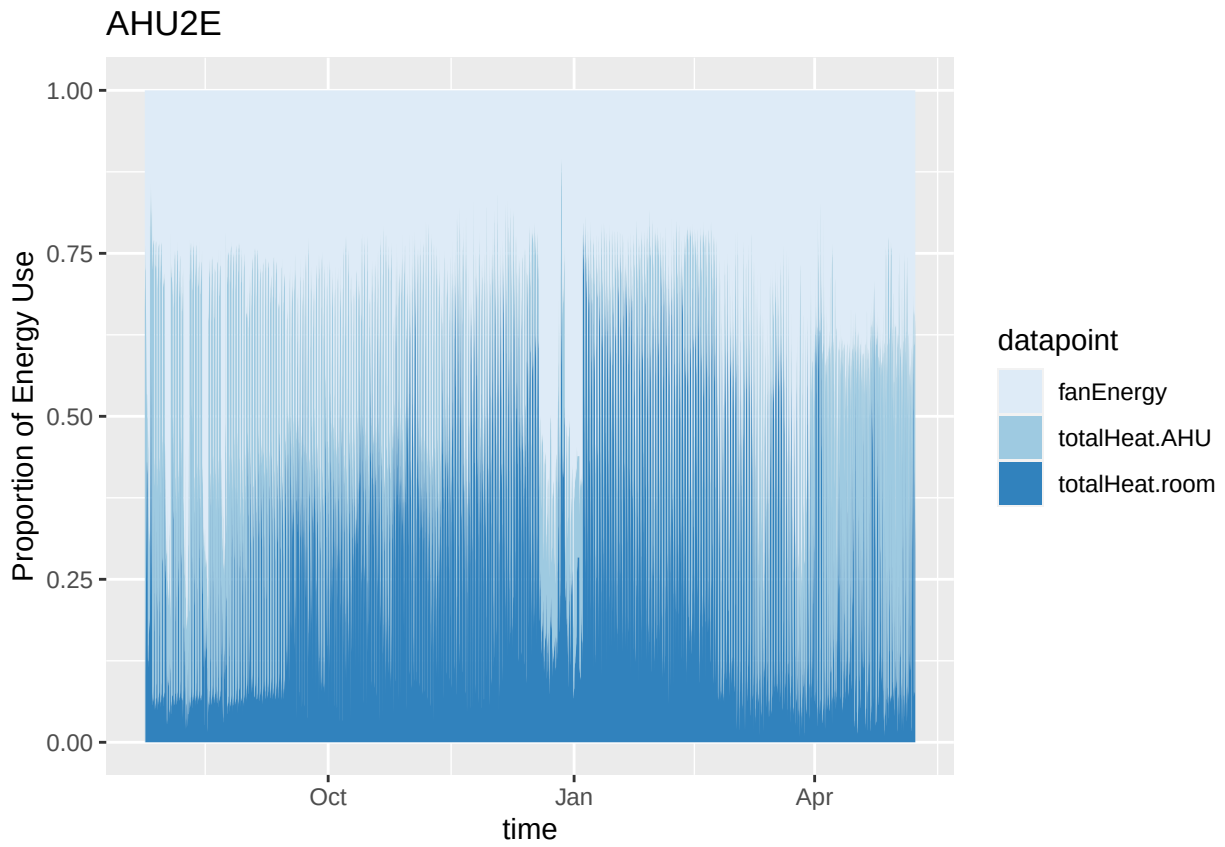

```
ggplot(data = longAHU2E, aes(x=time, y=value, fill=datapoint)) + geom_area() +ylab("Energy (kW)")+ ggti
```




```
# proportional stacked area plots
ggplot(longAHU2W, aes(x=time, y=value, fill=datapoint)) +
    geom_area(position = "fill") + scale_fill_brewer(palette = "Blues") +ylab("Proportion of Energy Use
```



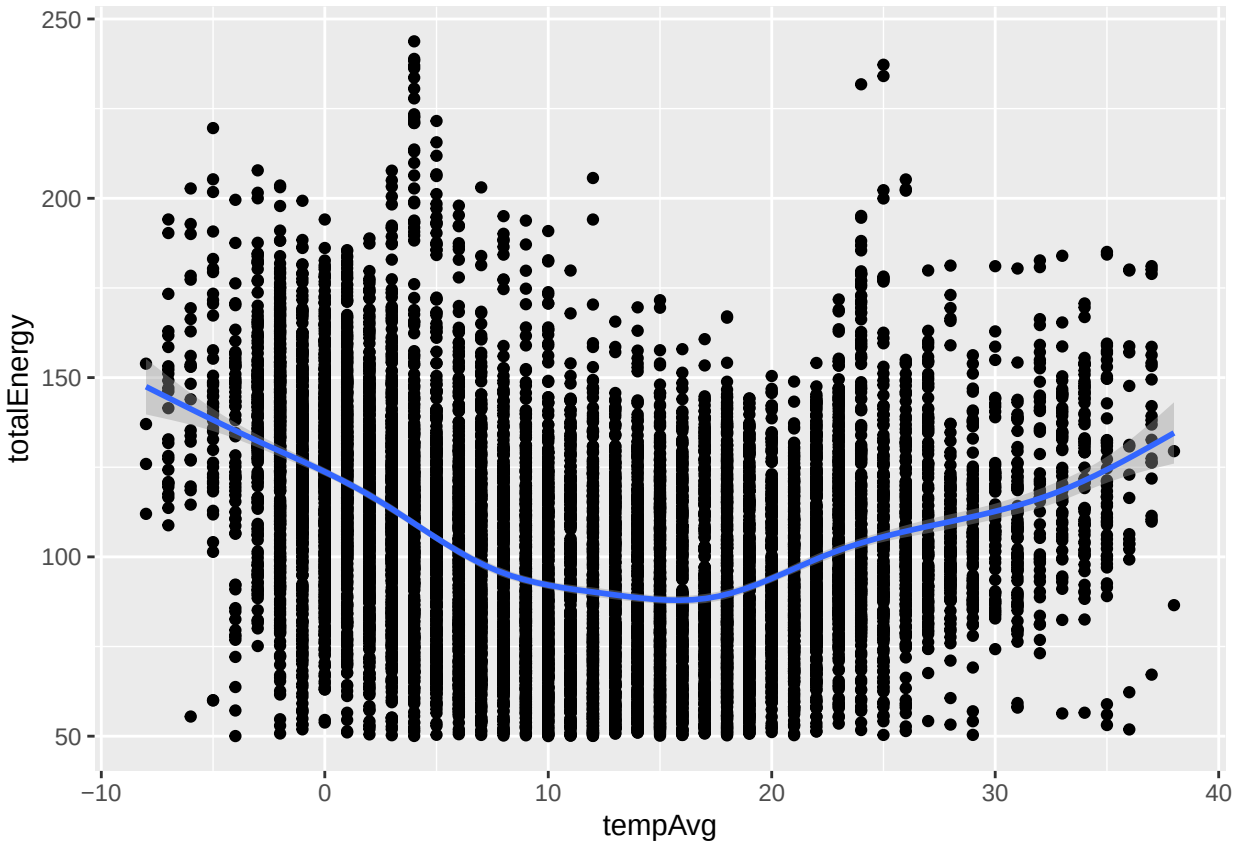

```
ggplot(longAHU2E, aes(x=time, y=value, fill=datapoint)) +
    geom_area(position = "fill") + scale_fill_brewer(palette = "Blues") +ylab("Proportion of Energy Use
```



It appears that room reheating uses a much larger proportion of the energy of the system in months where heating is more likely to be active than cooling and the proportion of energy expended on the fans increases dramatically as overall system energy use declines. Though we don't have a complete year of data, it appears shoulder seasons (fall/spring) consume less energy than winter/summer, in line with what we might expect given outdoor temperatures.

Relationship of Local Weather and Energy Consumption

```
weather_merge = weather

# shift weather data by a minute to allow for merging
weather_merge$obsTimeLocal = weather_merge$obsTimeLocal + 60

# merge data
left_join(totalEnergyDF, weather_merge, by=c("time" = "obsTimeLocal")) %>% filter(!is.na(totalEnergy))-

# plot, filtering out low energy values for when HVAC system is "off"
ggplot(data = weather_energy %>% filter(totalEnergy > 50), aes(x=tempAvg, y=totalEnergy)) + geom_point(
```



As the previous charts showing lower consumption in shoulder seasons implied, energy consumption appears to be roughly quadratic with respect to temperature.

Validation of Estimated Energy Consumption

Historical data is available for building-wide energy consumption, divided into:

- chilledWater (cooling)
- heating
- electricity

```
head(buildingEnergy)
```

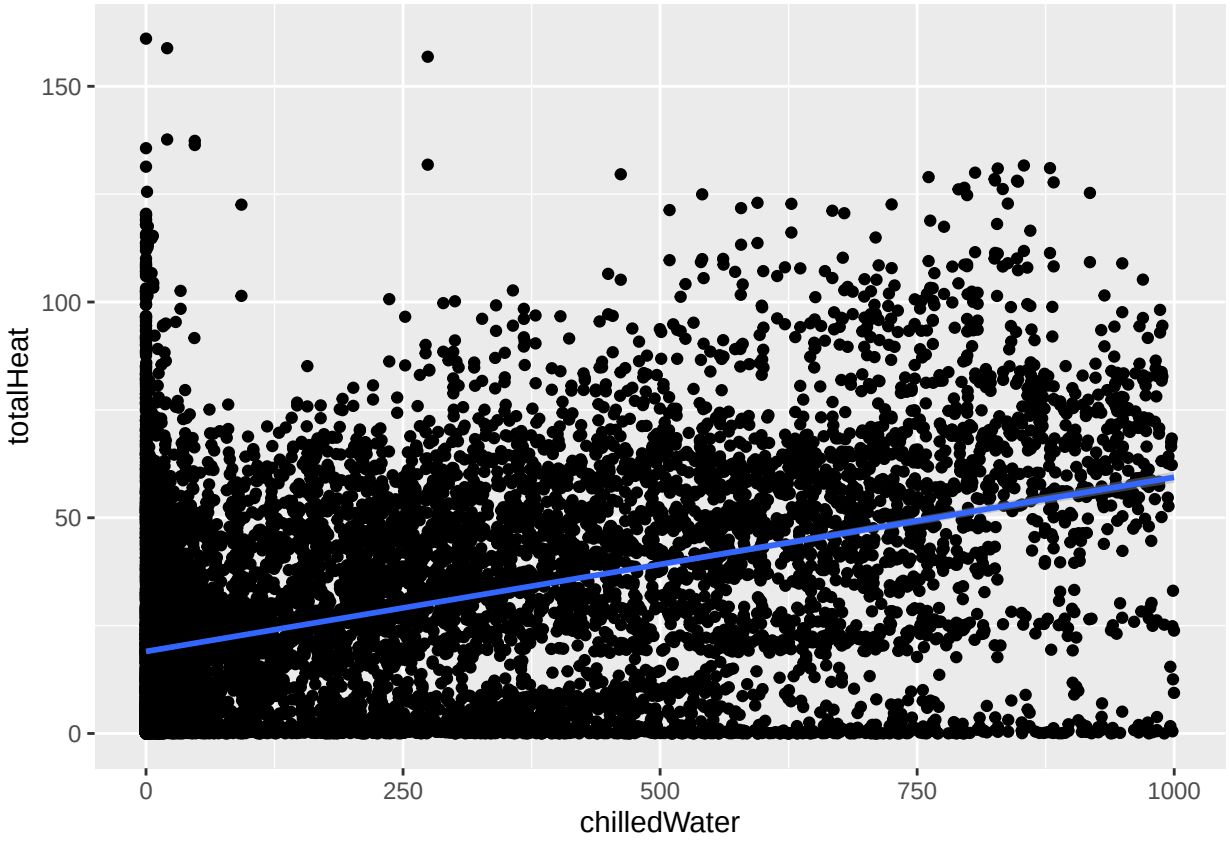
```
## # A tsibble: 6 x 4 [15m] <?>
##   time                chilledWater electricity heating
##   <dttm>                     <dbl>       <dbl>   <dbl>
## 1 2020-01-01 00:00:00         14.0        26.7    44.4
## 2 2020-01-01 00:15:00         17.5        26      64.8
## 3 2020-01-01 00:30:00         16.1        26      71.6
## 4 2020-01-01 00:45:00          0          27      80.8
## 5 2020-01-01 01:00:00          0          26      78.5
## 6 2020-01-01 01:15:00          0          26      70.5
```

As HVAC operation represents a significant part of building energy consumption, we would expect that trends in and changes of HVAC system energy consumption would closely track with building wide energy

consumption. Calculated historical energy consumption is plotted next to building-wide energy consumption for inspection:

Determine cooling state and evalutate relationship of calculated energy consumption in cooling state and actual historical chilled water consumption

```
equip %>% filter(!is.na(SAT) & !is.na(MAT)) -> equipVerify

# determine cooling status based on comparative temperature of input and output air
equipVerify <- equipVerify %>% mutate(cooling =
  if_else(
    MAT > SAT, 1, 0
  )
)

left_join(equipVerify, buildingEnergy, by="time") -> coolingVerify

coolingVerify <- coolingVerify %>% filter(cooling == 1) %>% as_tsibble(key = equipment, index = time)

# plot and model actual cooling energy consumption as a funciton of calcualted heating consumption
ggplot(data = coolingVerify, aes(x=chilledWater, y=totalHeat)) + geom_point() + geom_smooth(method = "lm
```



```
## Interactive plots not compatible with PDF output
# fig <- plot_ly(coolingVerify, x = ~time, y = ~totalHeat, name = 'AHU Heat', type = 'scatter', mode =
# fig %>% add_trace(y = ~(chilledWater*1000)/3412.142, name = 'Chilled Water', mode = 'lines')

coolingLM = lm(chilledWater~totalHeat, data = coolingVerify)

summary(coolingLM)
```

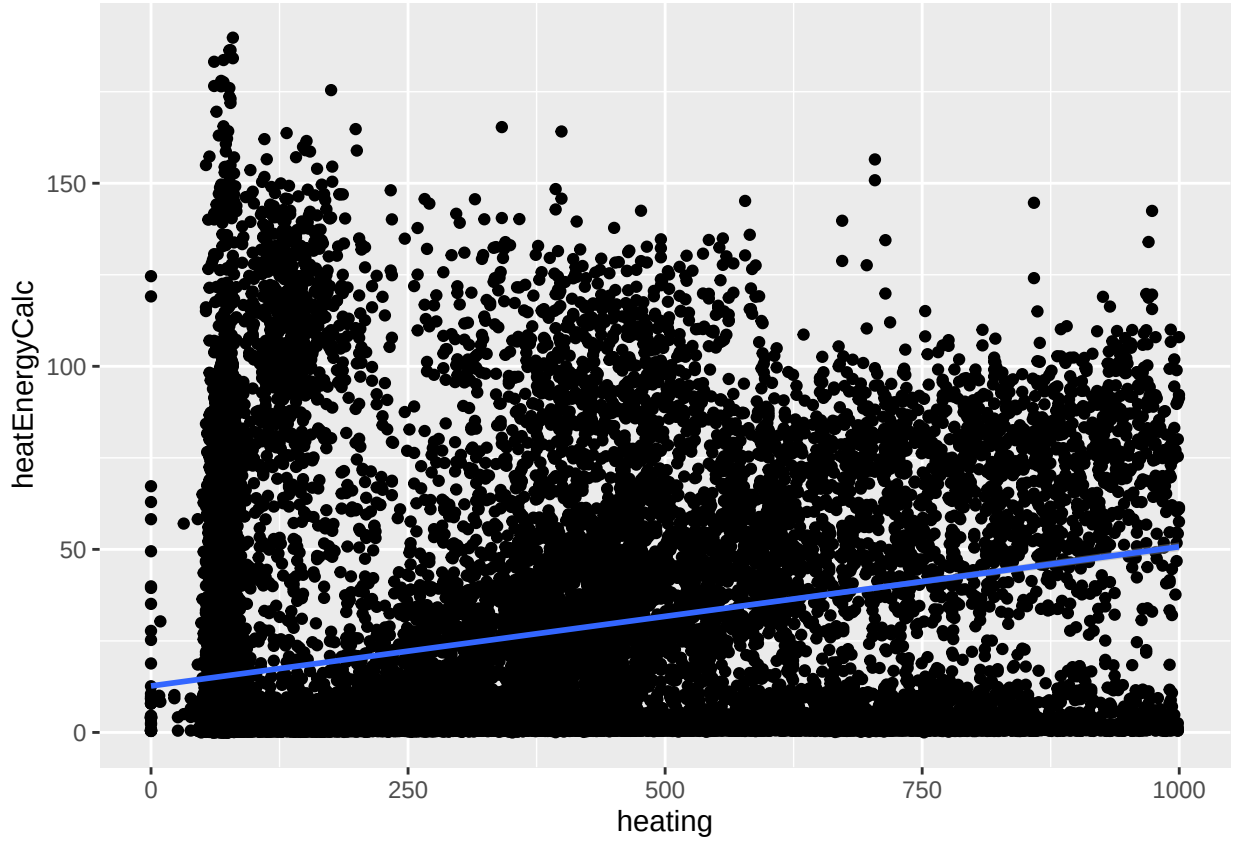
```
##
## Call:
## lm(formula = chilledWater ~ totalHeat, data = coolingVerify)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -842.1 -161.9 -100.5  154.3  890.0
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 106.01886    3.52299   30.09   <2e-16 ***
## totalHeat     4.57092    0.09118   50.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 250.1 on 11108 degrees of freedom
##   (614 observations deleted due to missingness)
## Multiple R-squared:  0.1845, Adjusted R-squared:  0.1844
## F-statistic:  2513 on 1 and 11108 DF,  p-value: < 2.2e-16
```

A linear relationship between the calculated amount of energy used in cooling and historical chilled water consumption appears to exist, though a linear model between these two quantities shows an R^2 of 0.1844. This suggests that tuning within set parameters used in calculated energy consumption might improve the accuracy of the calculations.

Evalutate relationship of calculated energy consumption in heating state and actual historical chilled water consumption

```
# aggregate room level contributions to energy consumption by AHU
rooms %>% group_by(equipment) %>% summarise(totalHeat = sum(totalHeat)) -> roomAHUHeat

# join  room and equipment level data
equip %>% inner_join(roomAHUHeat, by=c("time", "equipment"), suffix = c(".AHU", ".room")) -> totalEnergy

# create DF for enery verificaiton and determine heating status based on comparative temperature of inp
totalEnergyDF %>% filter(!is.na(SAT) & !is.na(MAT)) -> totalEnergyDFVerify

totalEnergyDFVerify <- totalEnergyDFVerify %>%
  mutate(isHeating = if_else(
    SAT > MAT, 1, 0
    )
  )
```

```r
# join in actual energy consumption data
left_join(totalEnergyDFVerify, buildingEnergy, by="time") -> heatingVerify

heatingVerify %>% filter(!is.na(heatingVerify$isHeating)) -> heatingVerify

# calculate total heating energy based on AHU heating status
heatingVerify <- heatingVerify %>%
  mutate(heatEnergyCalc = if_else(
    isHeating == 1, totalHeat.AHU + totalHeat.room, totalHeat.room
    )
  )

# plot and model actual heating energy consumption as a funciton of calcualted heating consumption
ggplot(data = heatingVerify, aes(x=heating, y=heatEnergyCalc)) + geom_point() + geom_smooth(method = "lm
```




```r
# # Interactive plots not compatible with PDF output
# fig <- plot_ly(heatingVerify, x = ~time, y = ~heatEnergyCalc, name = 'AHU Heat', type = 'scatter', mo

# # heating is adjusted from kBTU to kW
# fig %>% add_trace(y = ~(heating*1000)/3412.142, name = 'Heating', mode = 'lines')

heatingLM = lm(heating~heatEnergyCalc, data = heatingVerify)

summary(heatingLM)
```

```
##
## Call:
## lm(formula = heating ~ heatEnergyCalc, data = heatingVerify)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -559.0 -154.0 -133.0  135.4  783.2
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    214.33108    1.82976  117.14   <2e-16 ***
## heatEnergyCalc   2.21688    0.04549   48.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 242.3 on 25805 degrees of freedom
##   (1749 observations deleted due to missingness)
## Multiple R-squared:  0.08428,    Adjusted R-squared:  0.08425
## F-statistic:  2375 on 1 and 25805 DF,  p-value: < 2.2e-16
```

Modeling actual heat energy consumption as a linear function of calculated heat consumption results in an R^2 of only 0.084 though calculated heat is determined to be a significant predictor with a p-value < 0.001.

Further investigation is clearly needed to determine whether there are other significant contributors to consumption of heating and cooling energy within the building, or if the model needs more development (tuning of constant parameters used in calculations, specifically relative humidity, fan efficiency, and total pressure, would likely be a good place to start). As a further point of research, having energy readings directly metered from the HVAC unit would be the only fully reliable way to determine energy usage, especially if developing an improved control mechanism is ever attempted.

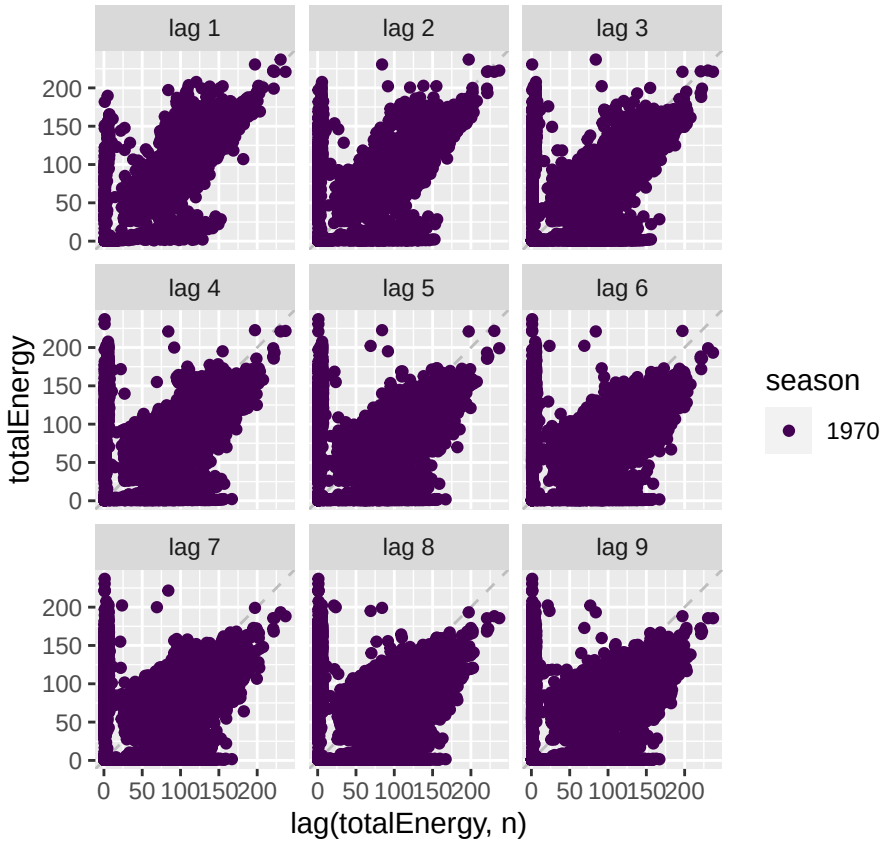
Formal model of data-generating process

Given the previously shown daily cycle of the data, shoulder season behaviour, and quadratic relationship with local temperature, regression with ARIMA errors seems a reasonable place to begin to model the data generating process behind total energy use in the system.

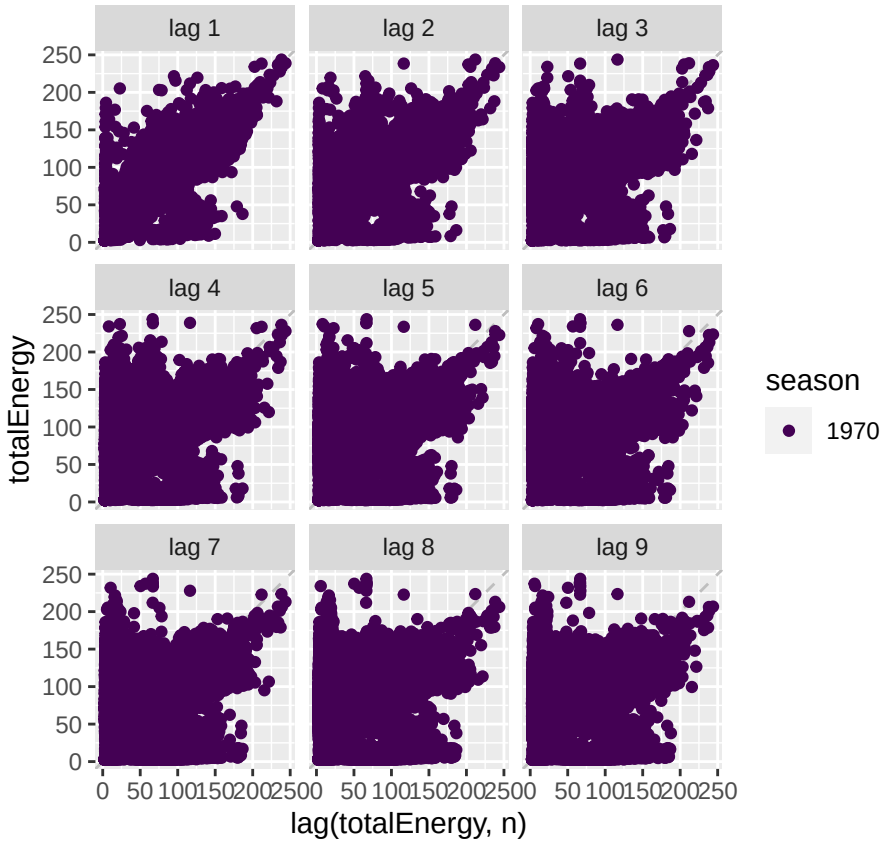
An investigation of lags and autocorrelation will provide the basis for the parameter selection of the ARIMA model.

Lag Plots

```
# retreive cleaned data, fill gaps
load("C:/Users/canea/Documents/UVA/Spring 2021/Timeseries/neale-caleb/data/cleanedData.Rdata")
totalEnergyDF = totalEnergyDF %>% fill_gaps()

totalEnergyDF %>% select(totalEnergy) %>% filter(equipment == "AHU2E") %>% gg_lag(y=totalEnergy, geom =
```



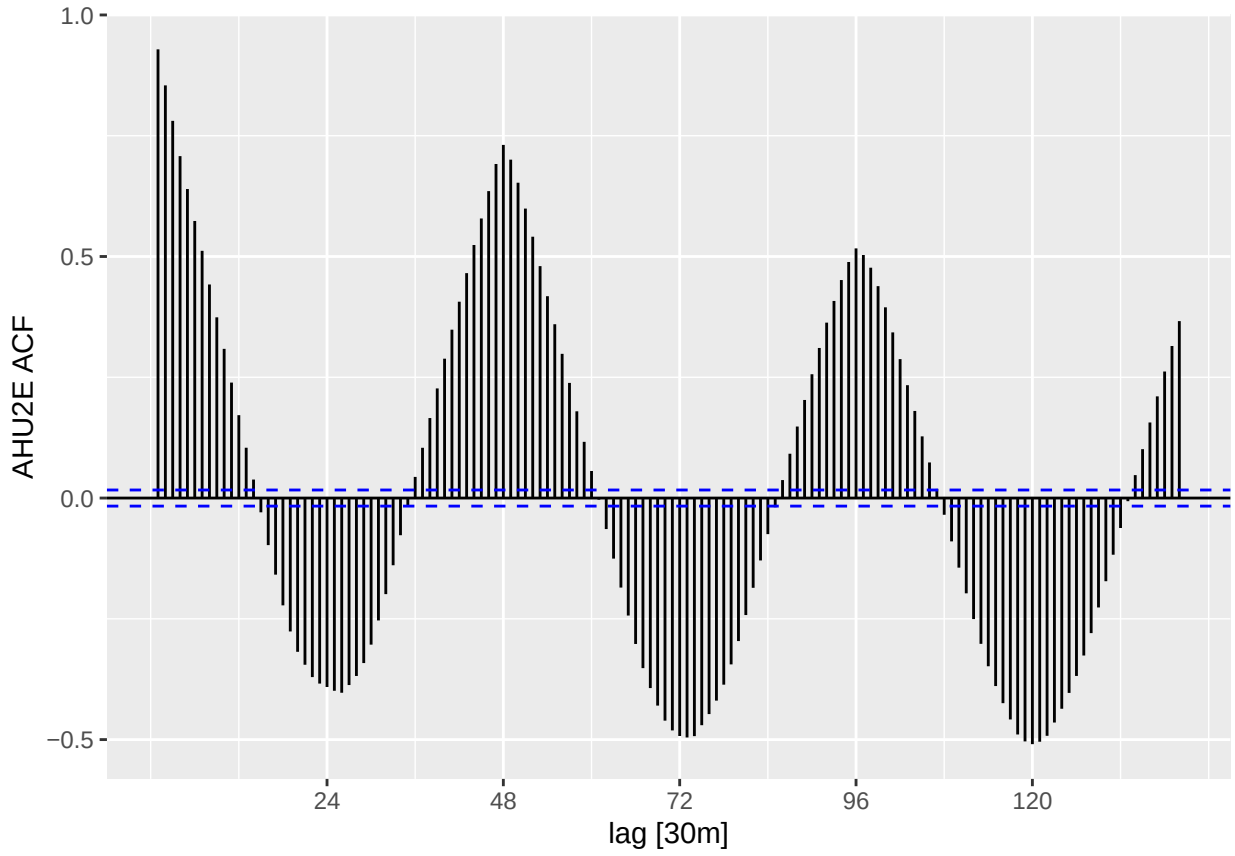

```
totalEnergyDF %>% select(totalEnergy) %>% filter(equipment == "AHU2W") %>% gg_lag(y=totalEnergy, geom =
```



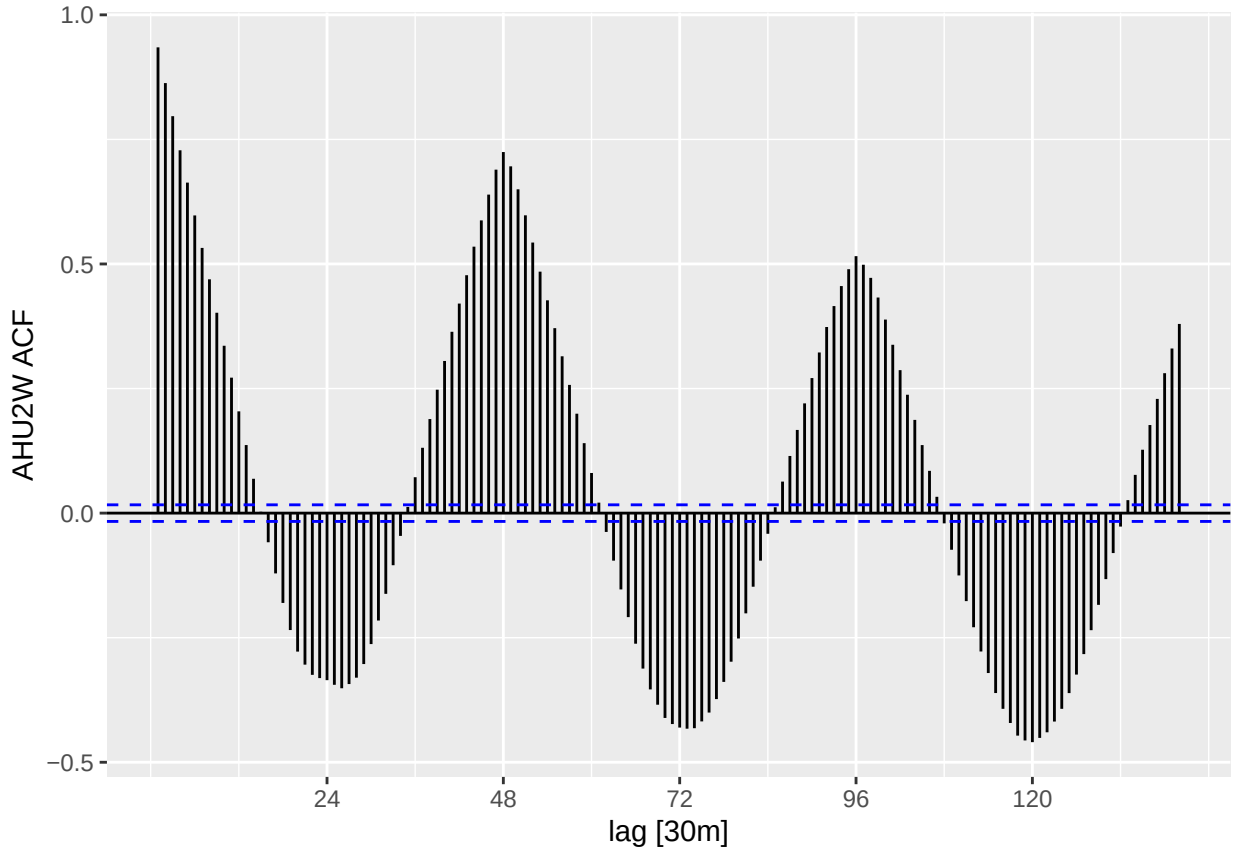
Potentially given the lower bound previously seen on AHU2W energy consumption, a stronger relationship appears in the lags of AHU2E than AHU2W.

Autocorrelation Plots

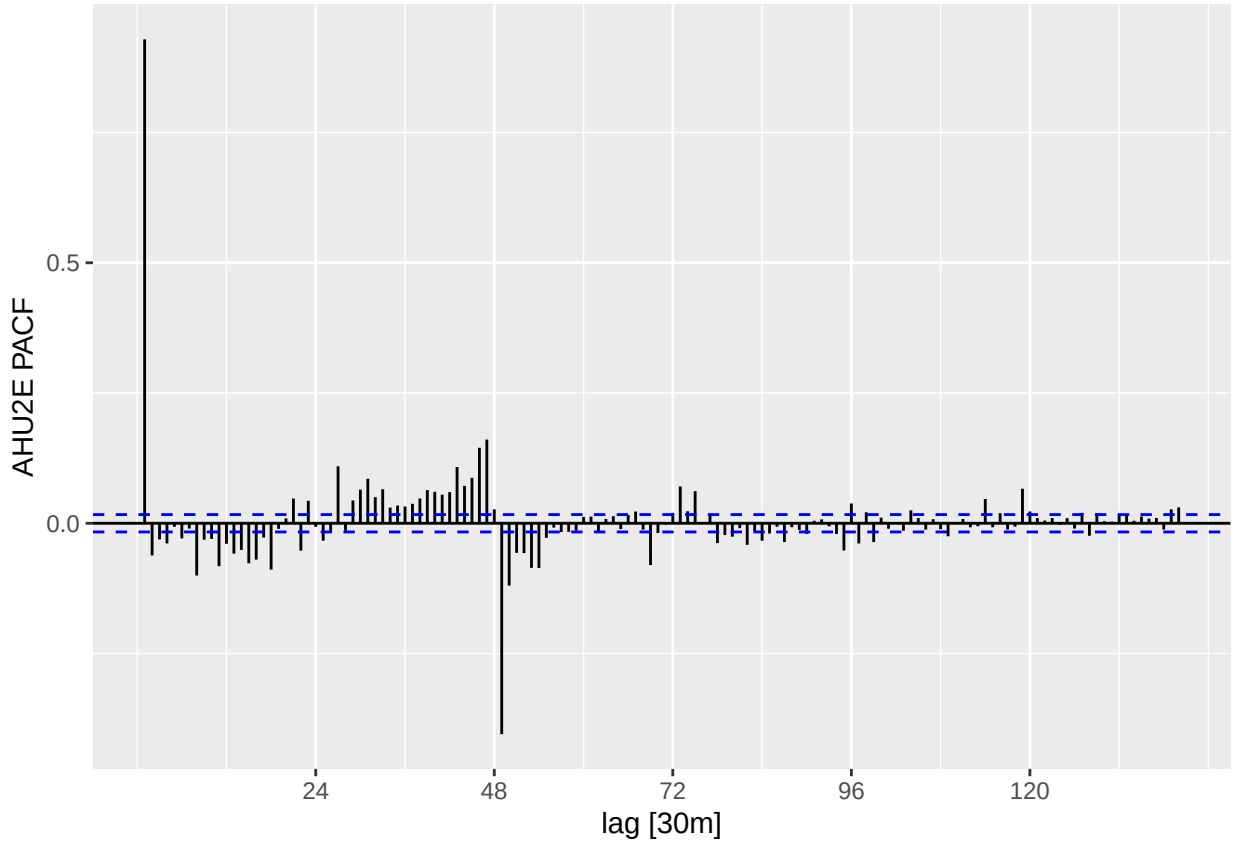
```
# ACFs
totalEnergyDF %>% filter(equipment == "AHU2E") %>% ACF(totalEnergy, lag_max = 140) %>% autoplot() + ylal
```



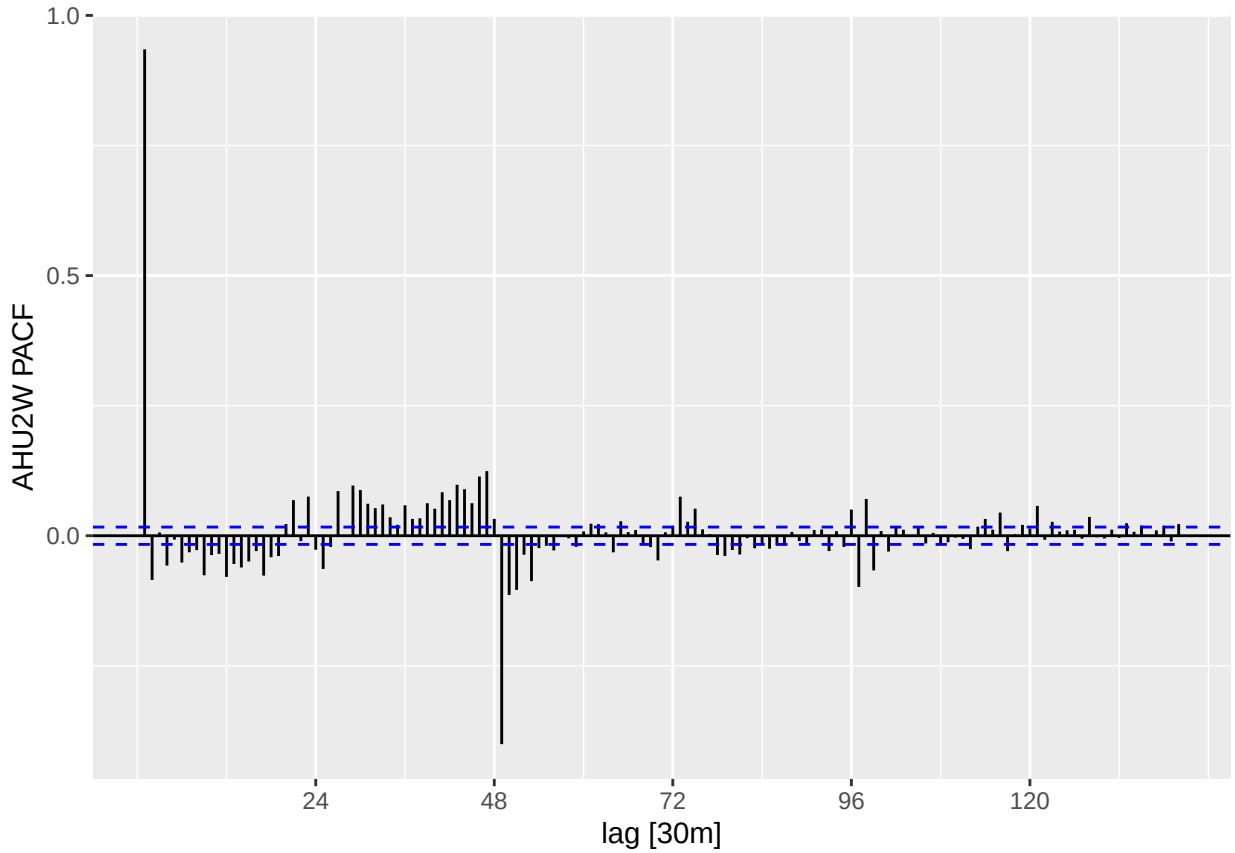

```
totalEnergyDF %>% filter(equipment == "AHU2W") %>% ACF(totalEnergy, lag_max = 140) %>% autoplot() + ylab
```




```
# PACFs
totalEnergyDF %>% filter(equipment == "AHU2E") %>% PACF(totalEnergy, lag_max = 140) %>% autoplot() + yla
```




```
totalEnergyDF %>% filter(equipment == "AHU2W") %>% PACF(totalEnergy, lag_max = 140) %>% autoplot() + yla
```

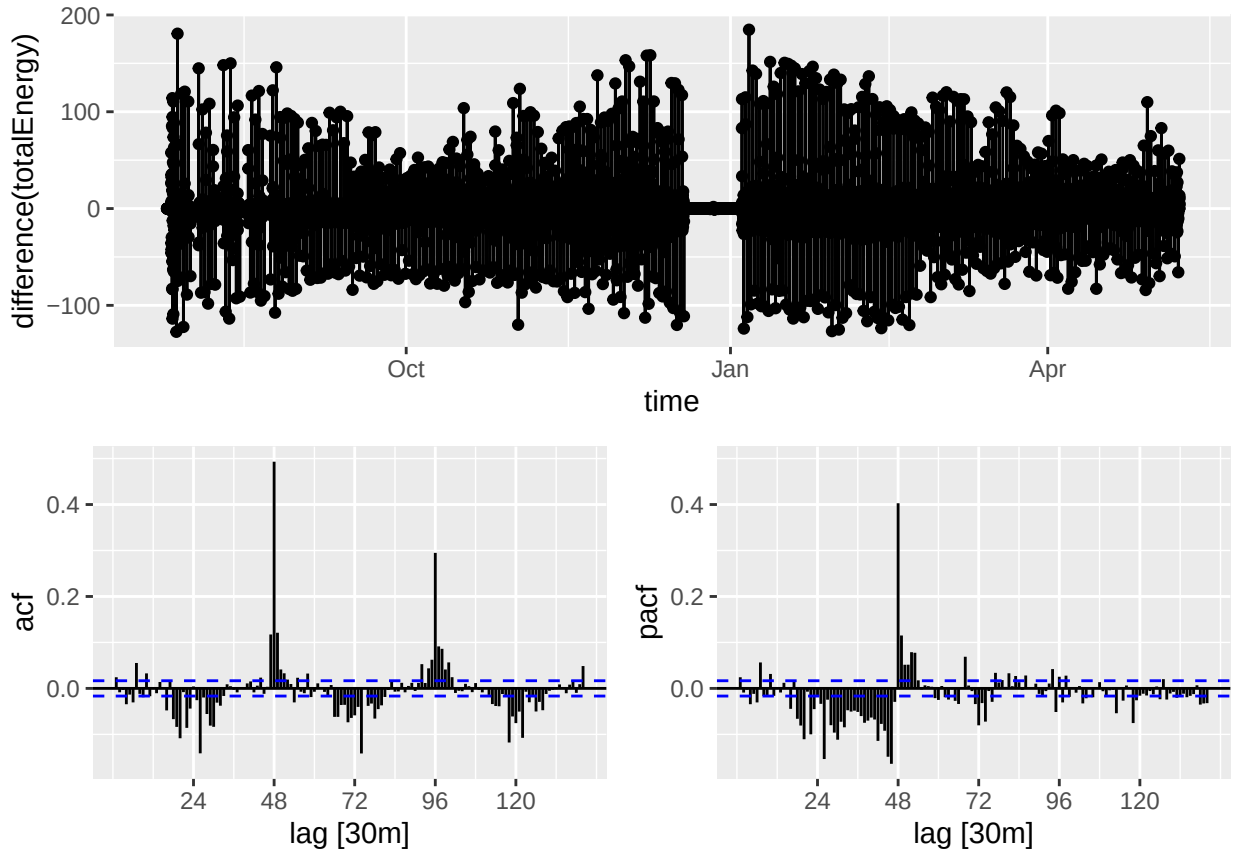


Given a 48 period lag corresponds to one day, the behaviour seen in the ACF plots show exactly what we might expect; ACF is highest at multiples of 24 hours from a given point. PACF plots also show highest PACF values at about the 24 hour mark, though additional 24 hour extensions provide significantly less information.

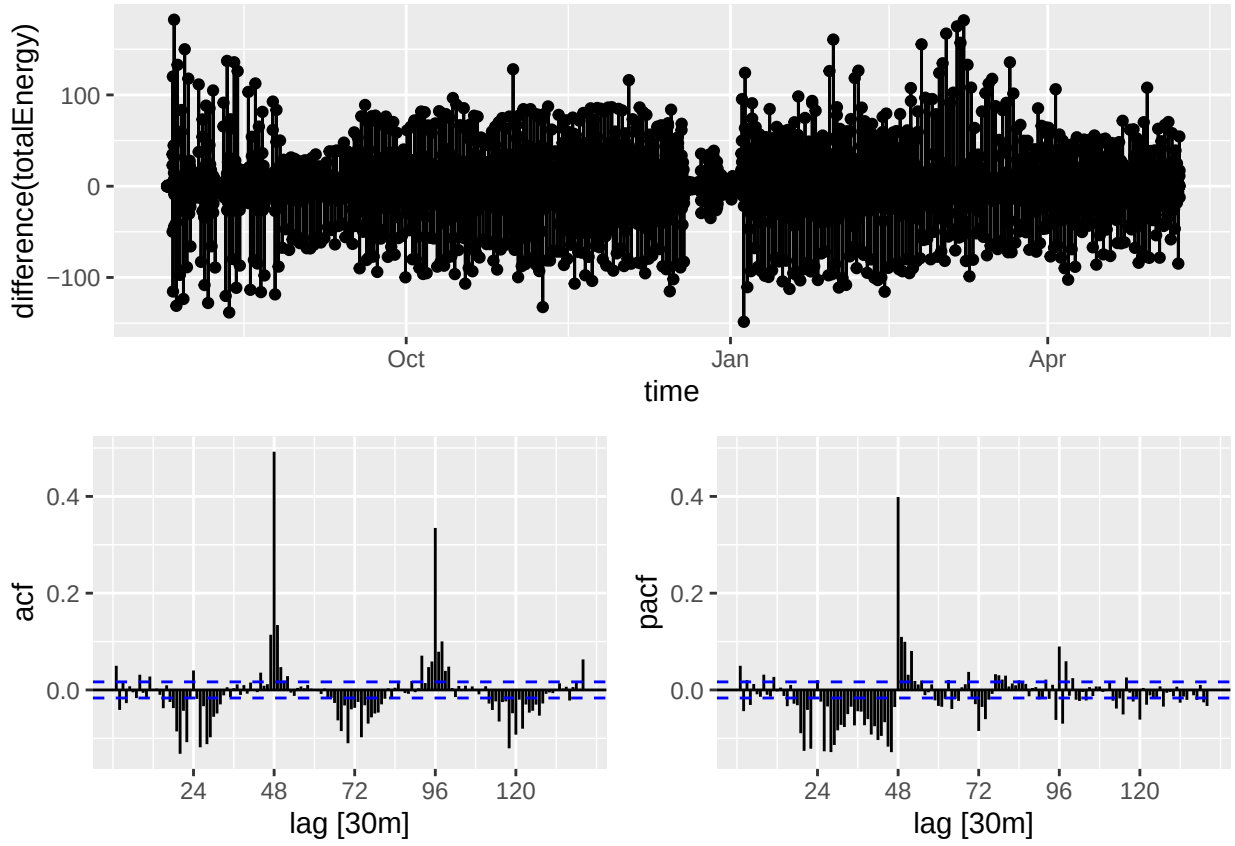
However, given the potential for trends in the data, an investigation into the differences of the data is needed.

Differencing

```
totalEnergyDF %>% filter(equipment == "AHU2E") %>% gg_tsdisplay(difference(totalEnergy), plot_type = "pa
```




```
totalEnergyDF %>% filter(equipment == "AHU2W") %>% gg_tsdisplay(difference(totalEnergy), plot_type = "pa
```

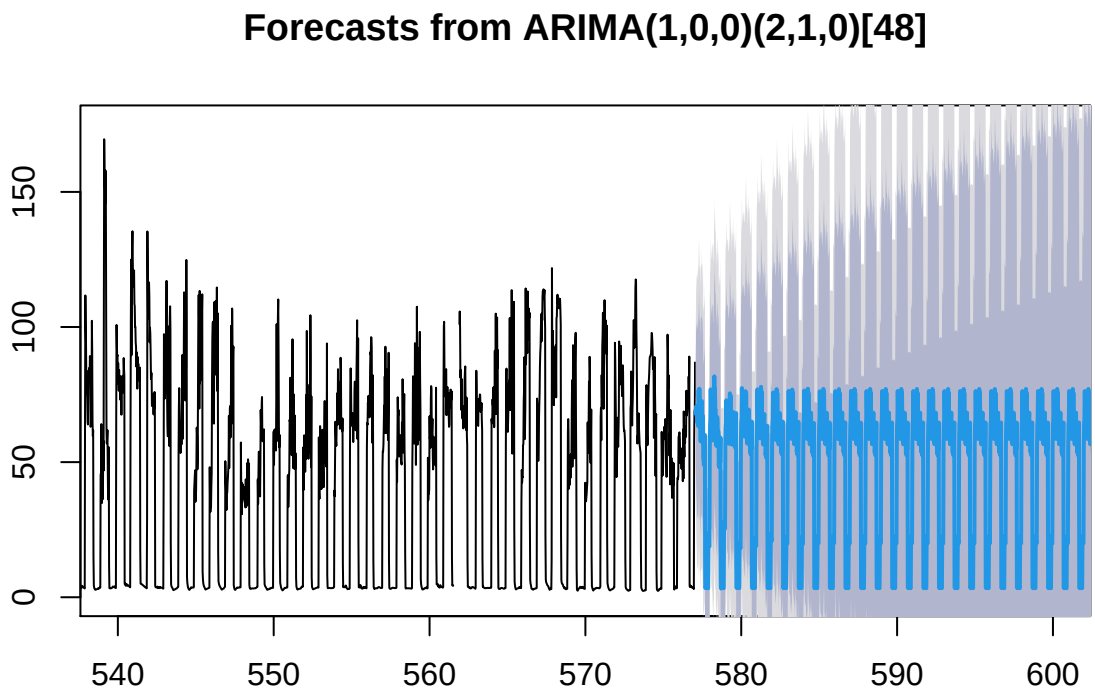


Note on Modeling On my machine, the models in the following sections take a while to train (about 20-30 minutes). Saved versions of the models are available in the "models" folder in this repository and can be loaded using "load('models/[modelname].RData')" for faster computing.

Preliminary ARIMA Modeling

The ACF is suggestive of an MA(52) model while the PACF suggests an AR(54) model, so ARIMA(0, 1, 52) and ARIMA(54, 1, 0) might be a good place to start for initially fit models. However, a more appropriate approach would likely be setting the seasonality of the data to daily (48 periods) and attempting ARIMA from there. This may not accurately address yearlong seasonal trends, but weather data may account for this in a later dynamic regression model.

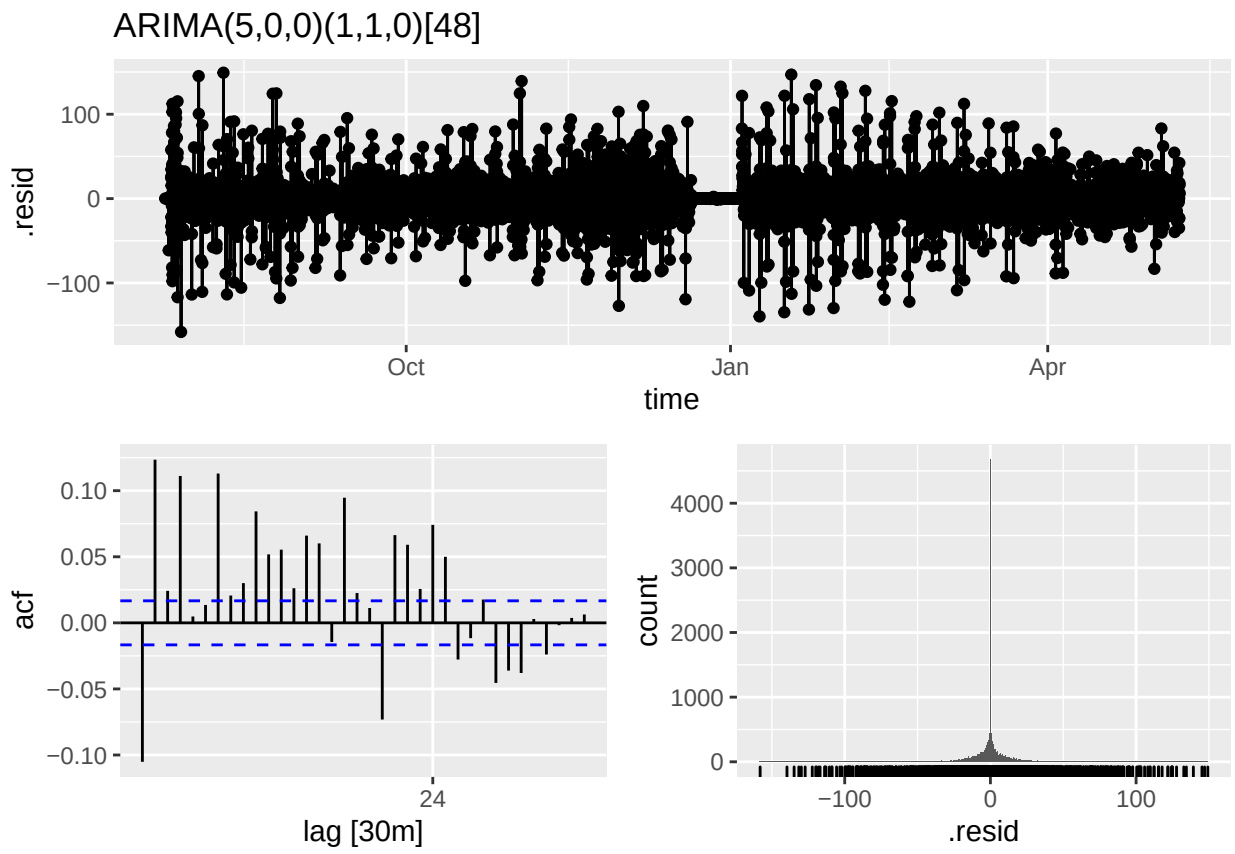
```
# create univariate TS object and inform it of 48 period seasonality
univar = ts(totalEnergyDF, frequency = 48)[,7]

# create model and forecast for 100 days
load("models/univarARIMA.RData")
# auto.arima(univar) -> univarARIMA

UAfc <- forecast(univarARIMA, h=4800)

# plot forecast
plot(UAfc, xlim=c(540, 600), ylim=c(0,175))
```



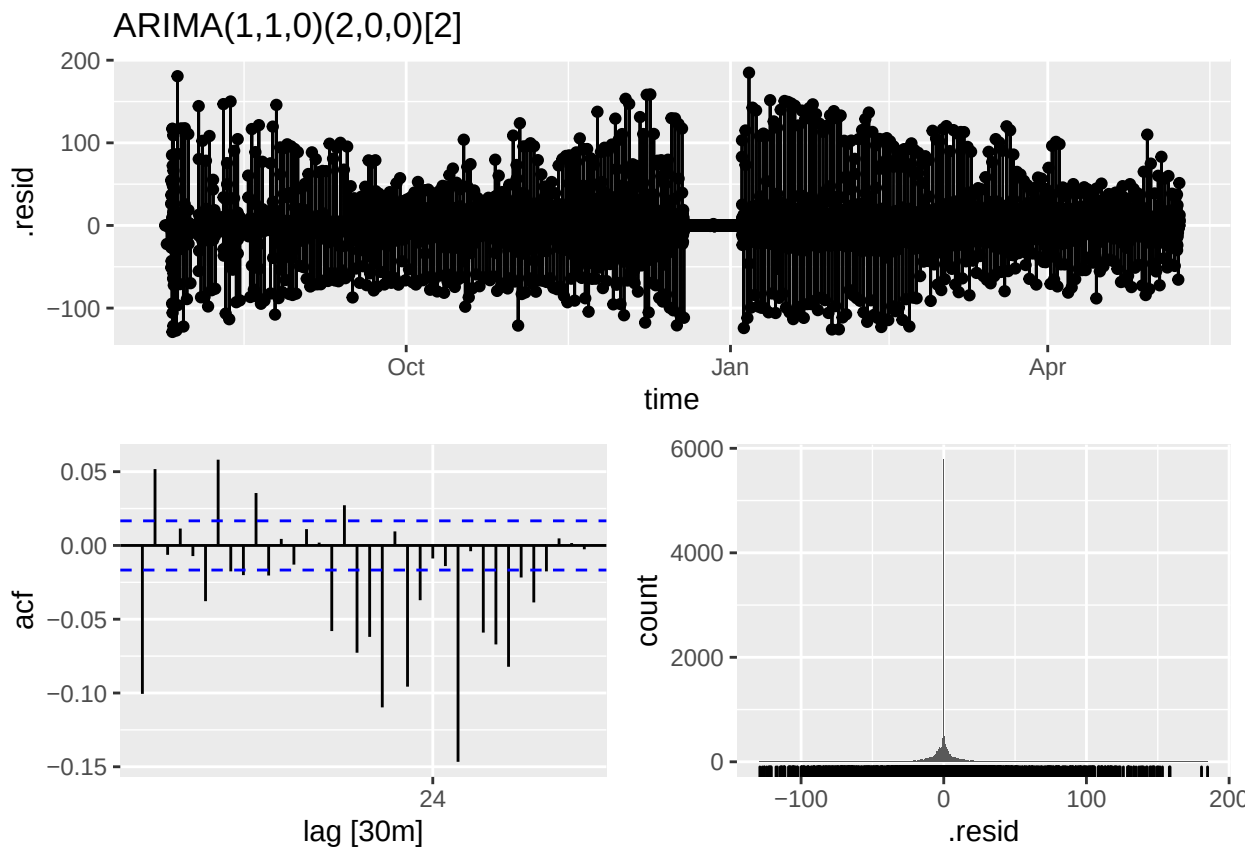
This model seems to capture the daily cycle of energy consumption quite well, but doesn't well account for general variation in the data, which could likely be attributed to changing outside temperatures. To determine if another ARIMA model would perform better, fable::ARIMA can be used to investigate additional ARIMA models before attempting dynamic regression with ARIMA errors.

ARIMA Model Search

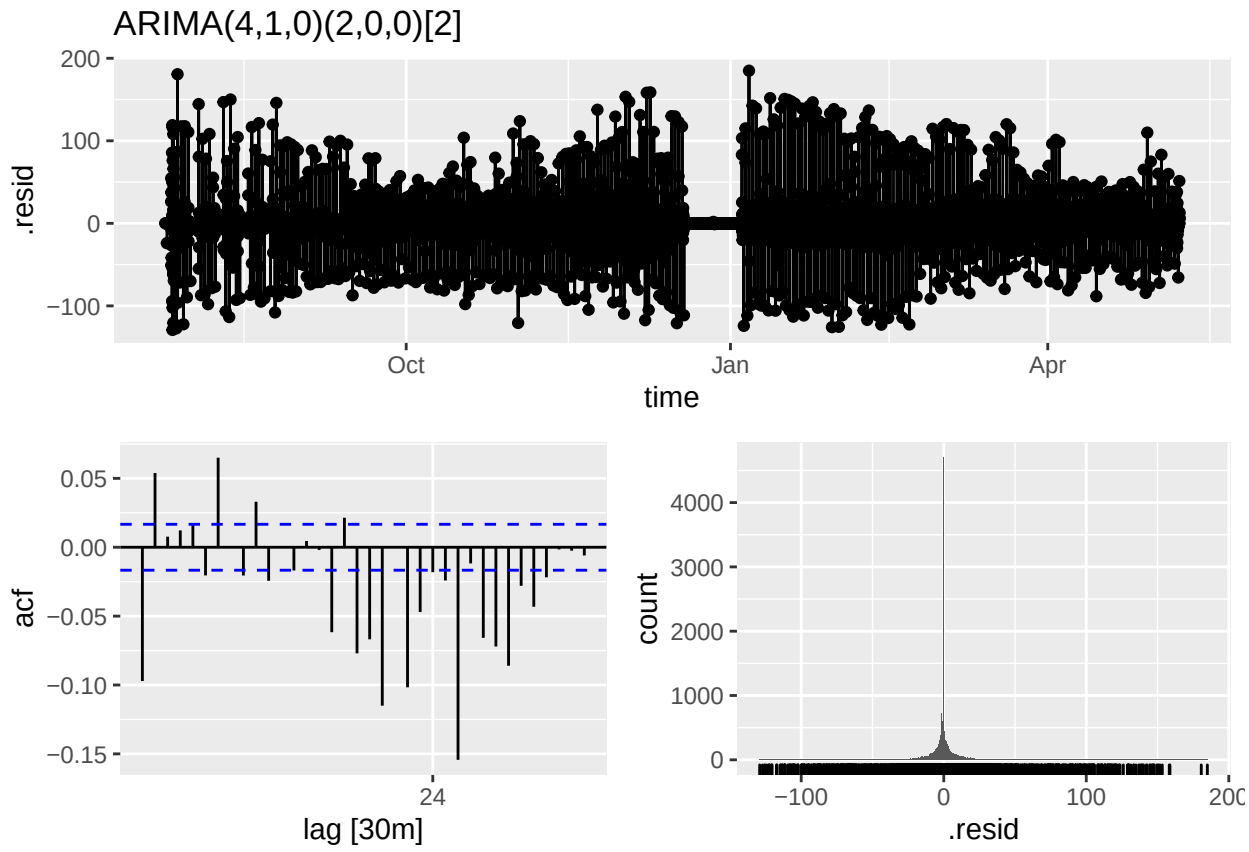
```
load(file ="models/fableModels.RData")

# Create models using fable::ARIMA for better plotting interface
# models <- totalEnergyDF %>%
#   filter(equipment == "AHU2E") %>%
#   model(stepwise = ARIMA(totalEnergy),
#         search = ARIMA(totalEnergy, stepwise = FALSE),
#         seasonal = ARIMA(totalEnergy ~ PDQ(period=48)))

models %>% select(seasonal) -> seasonal
models %>% select(stepwise) -> stepwise
models %>% select(search) -> search

# plots residuals
seasonal %>% gg_tsresiduals() + ggtitle("ARIMA(5,0,0)(1,1,0)[48]")
```



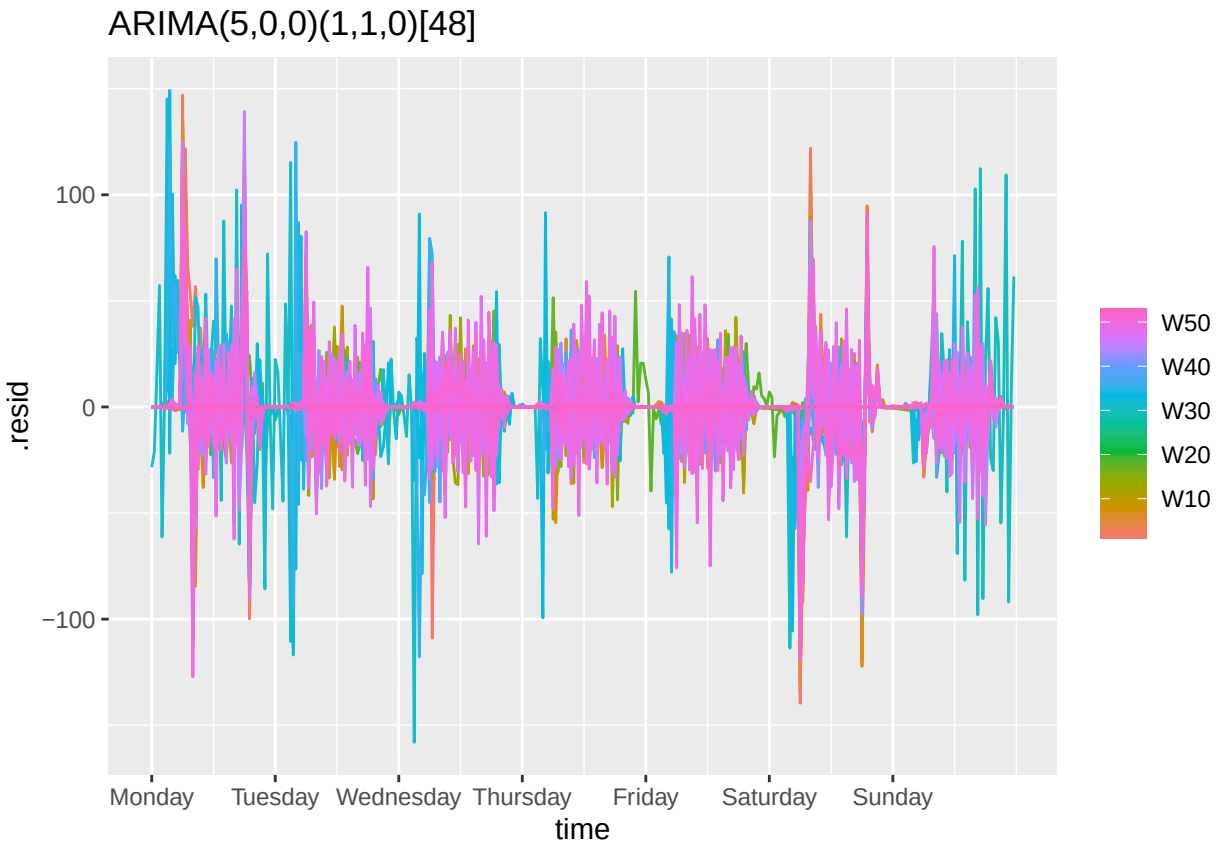

```
stepwise %>% gg_tsresiduals() + ggtitle("ARIMA(1,1,0)(2,0,0)[2]")
```



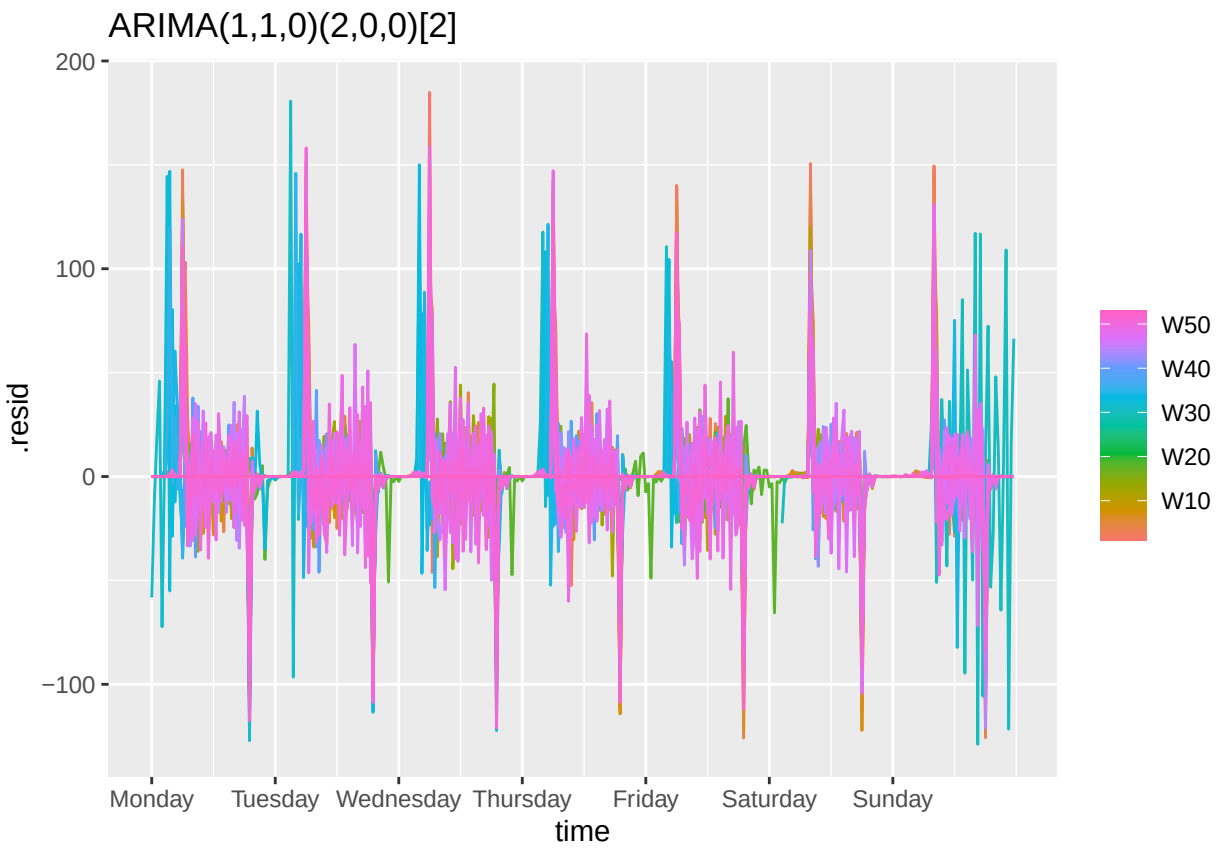

```
search %>% gg_tsresiduals() + ggtitle("ARIMA(4,1,0)(2,0,0)[2]")
```



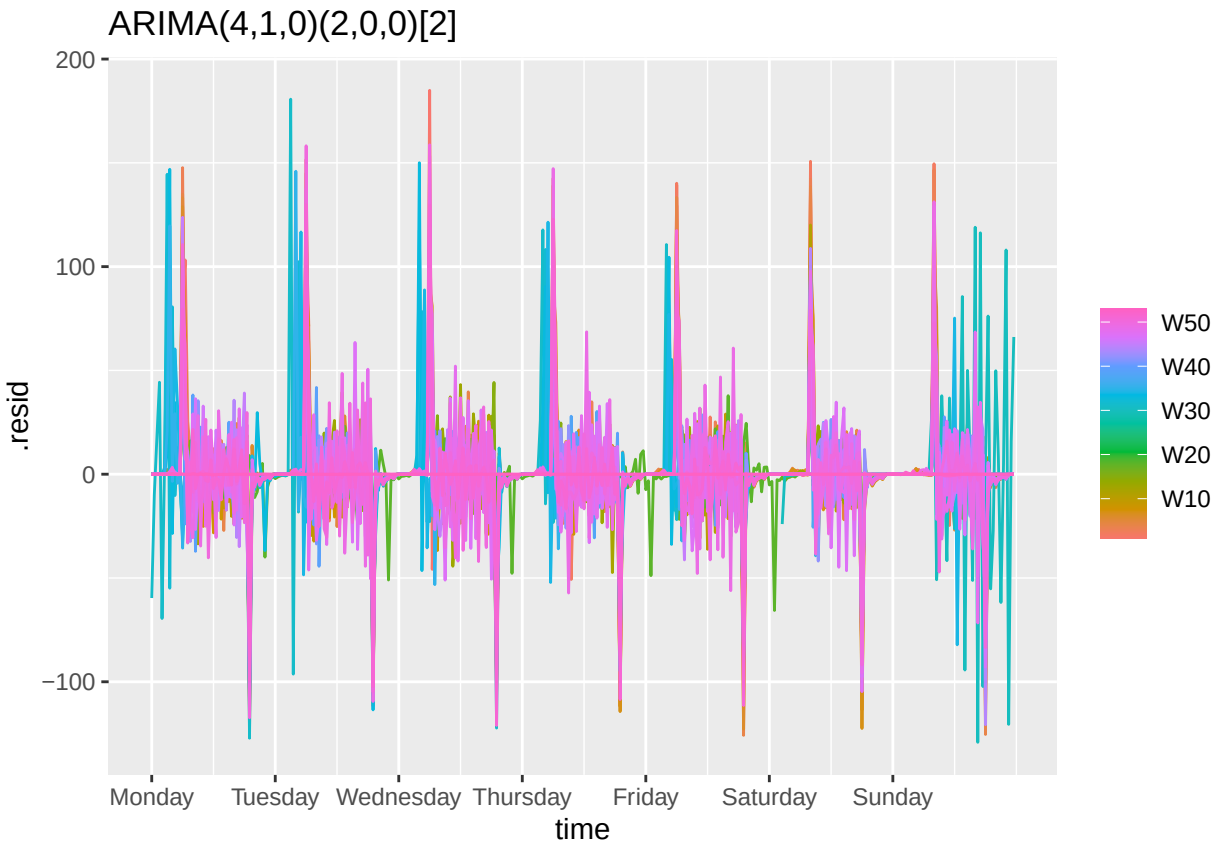

```
# weekly seasonal plot of residuals
residuals(seasonal) %>%
  as_tsibble(key=.model, index=time) %>%
  gg_season(y=.resid, period = "week") + ggtitle("ARIMA(5,0,0)(1,1,0)[48]")
```




```
residuals(stepwise) %>%
  as_tsibble(key=.model, index=time) %>%
  gg_season(y=.resid, period = "week") + ggtitle("ARIMA(1,1,0)(2,0,0)[2]")
```

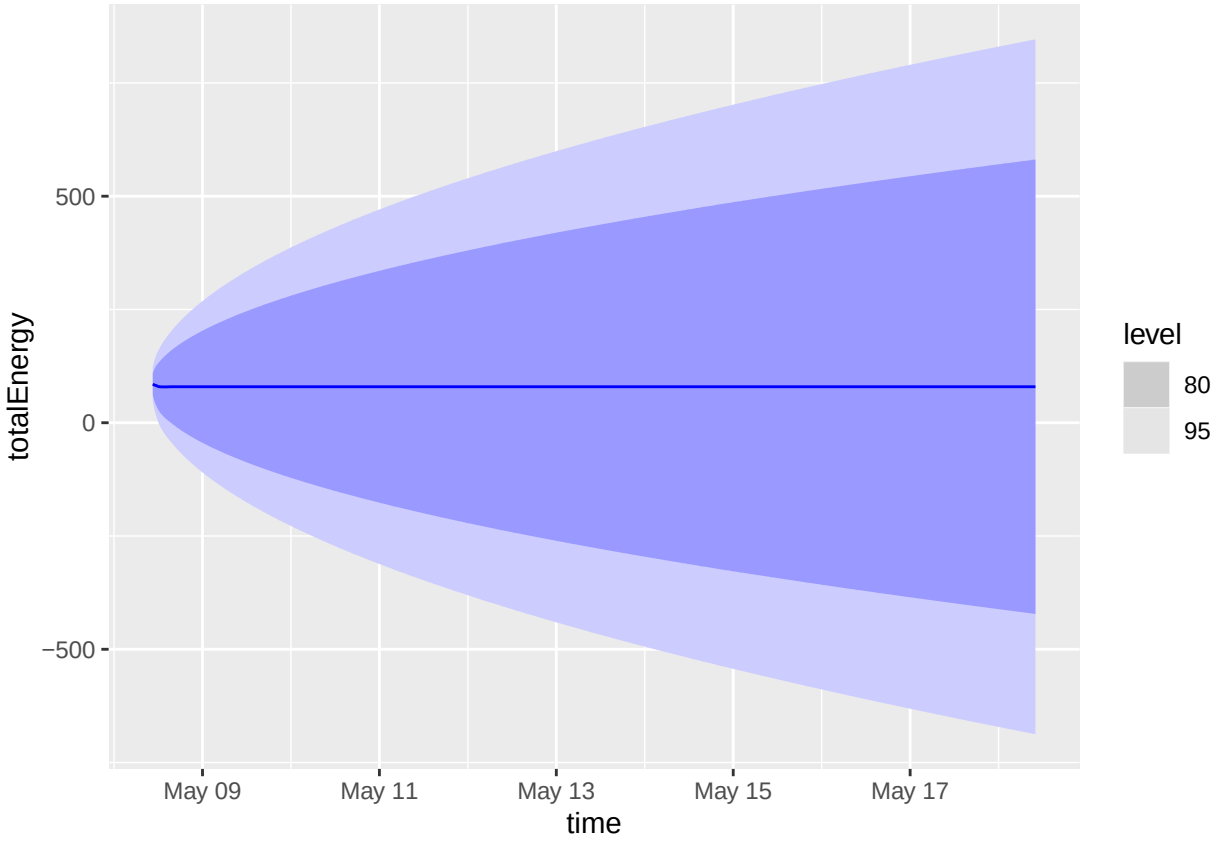



```
residuals(search) %>%
  as_tsibble(key=.model, index=time) %>%
  gg_season(y=.resid, period = "week") + ggtitle("ARIMA(4,1,0)(2,0,0)[2]")
```

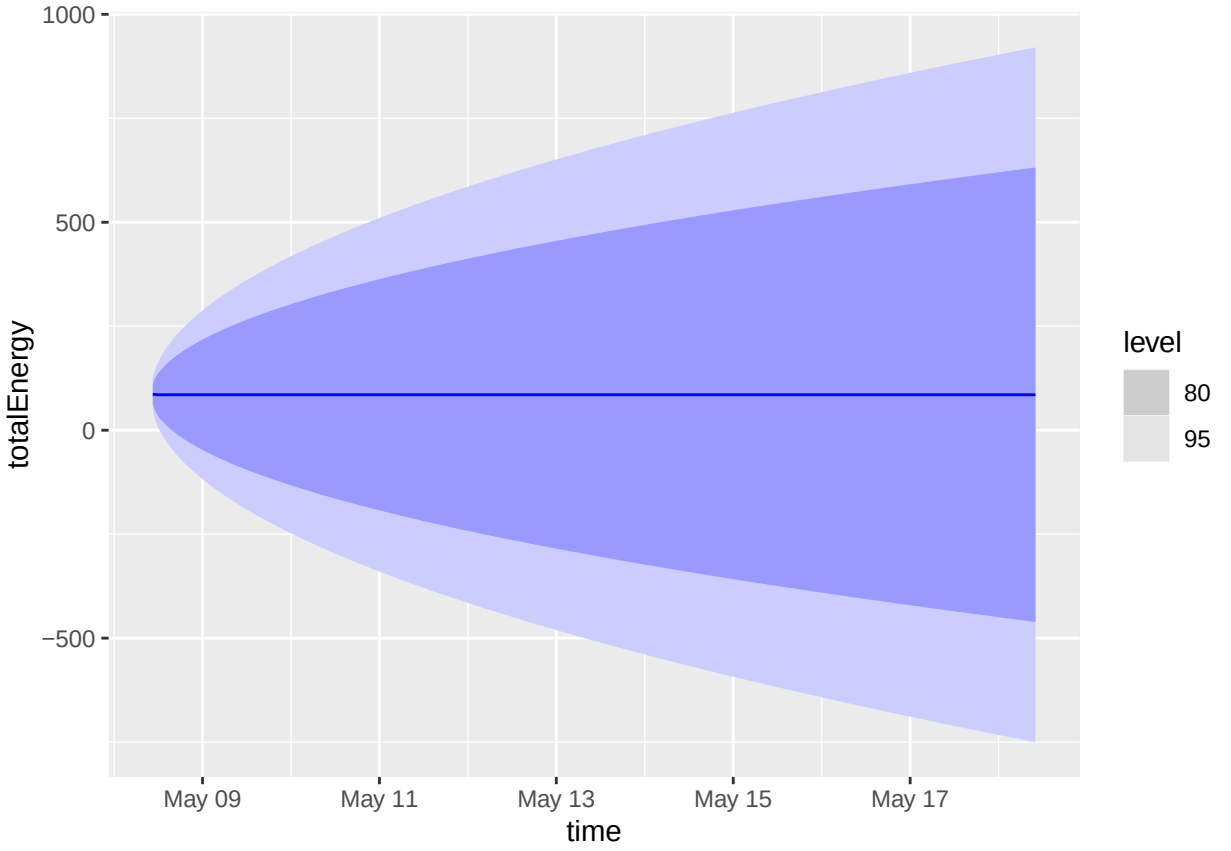


It appears the seasonal model has residuals which are the most normally distributed and which most closely approach white noise. The drop in residuals in the month of January can be attributed to system inactvity as the University was on winter break. Seasonal plots also show more balance in the seasonally adjusted model compared to the large spikes at the beginning and end of each day in the search and stepwise models. Evaluating forecasts gives a clearer picture.

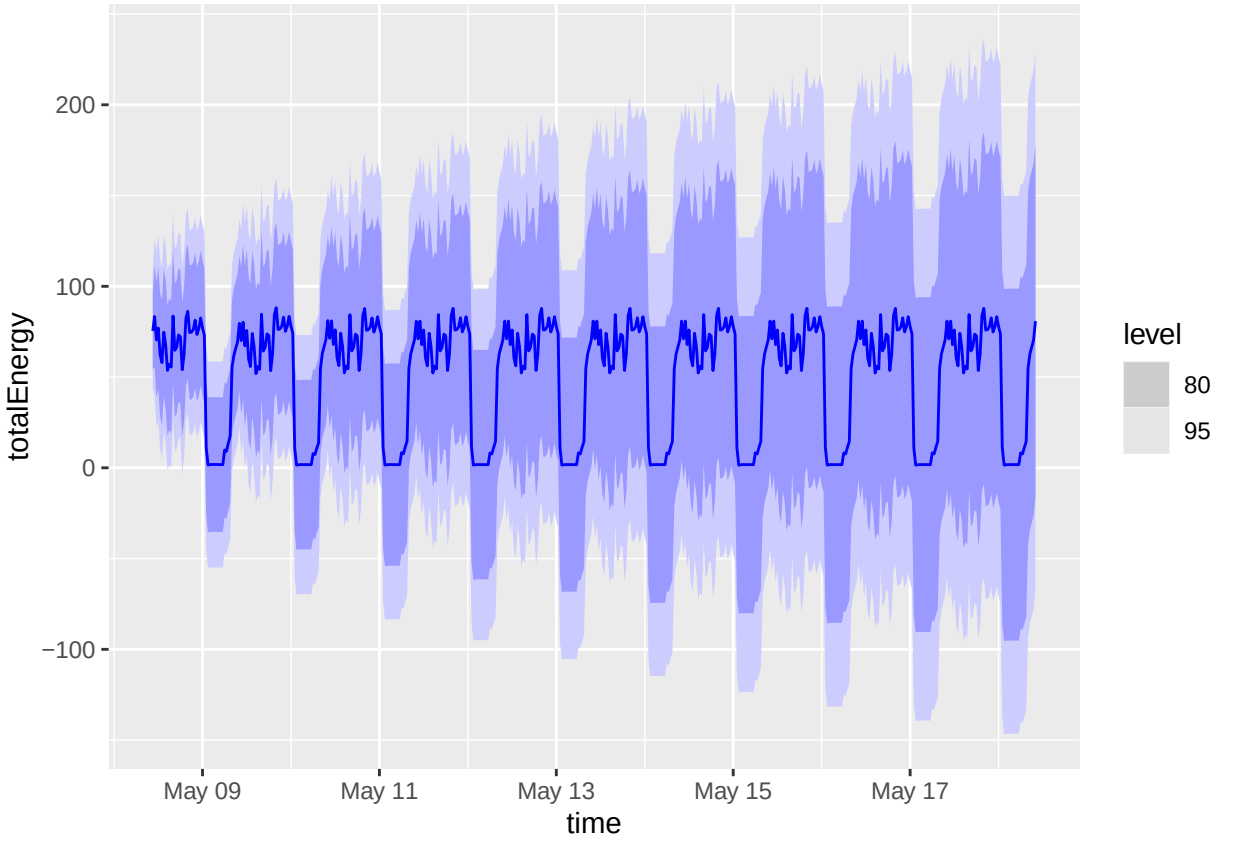
```
search %>% forecast(h=480) -> searchFC
stepwise %>% forecast(h=480) -> stepwiseFC
seasonal %>% forecast(h=480) -> seasonalFC

searchFC %>% autoplot()
```




```
stepwiseFC %>% autoplot()
```




```
seasonalFC %>% autoplot()
```



Only the model with a 48 seasonal term captures any of the daily cycle in the data, whereas the other two models seems to take a blind guess at the mean value of the process.

Dynamic Regression with ARIMA Errors

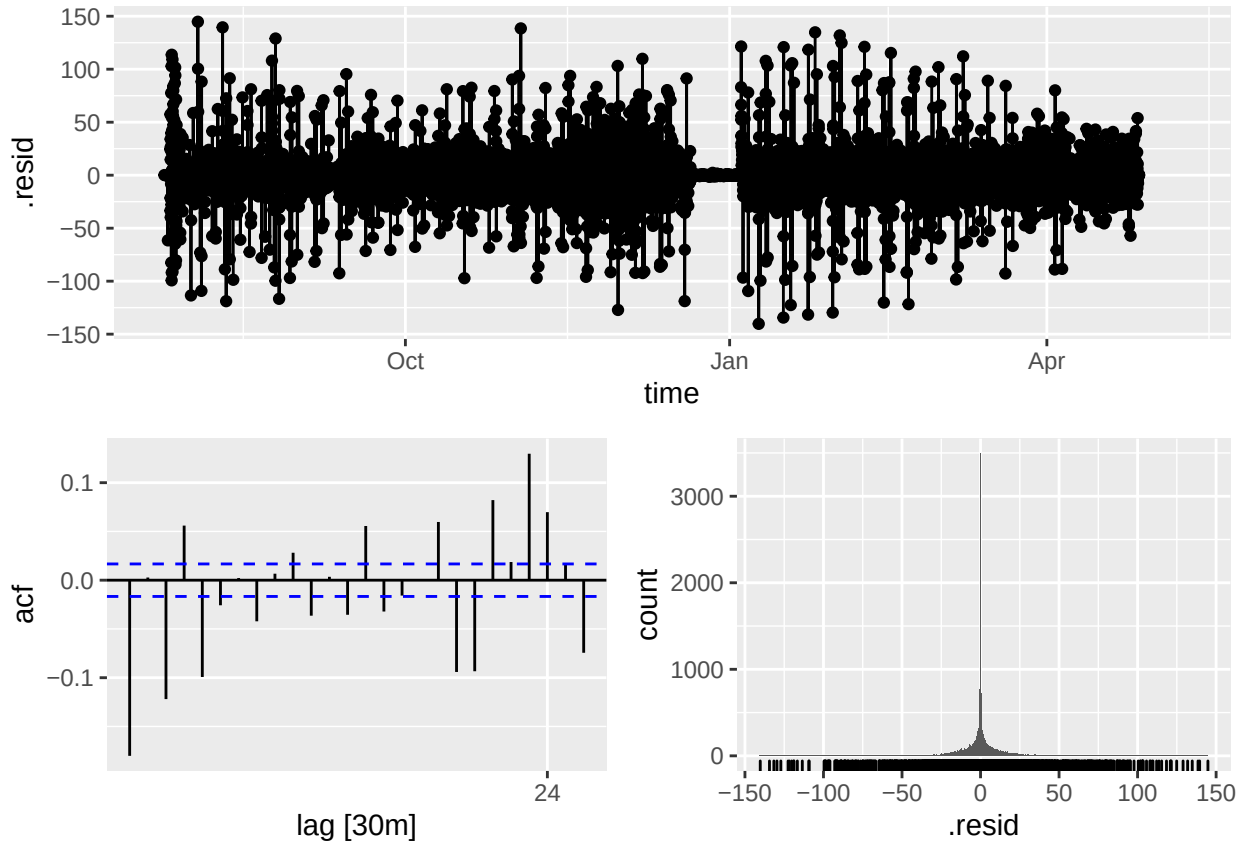
The ARIMA model seems to capture the daily cycle in the data well, but daily and seasonal variation could likely be explained by daily and seasonal changes in outdoor temperatures. To capture this variation, dynamic regression with ARIMA error can be applied.

```
load("models/dynReg.RData")

# Dynamic regression with all available data
# dynReg <- weather_energy %>%
#   fill_gaps() %>%
#   filter(equipment == "AHU2E") %>%
#   model(ARIMA(totalEnergy ~ tempAvg + I(tempAvg^2) + PDQ(period=48)))

report(dynReg)
```

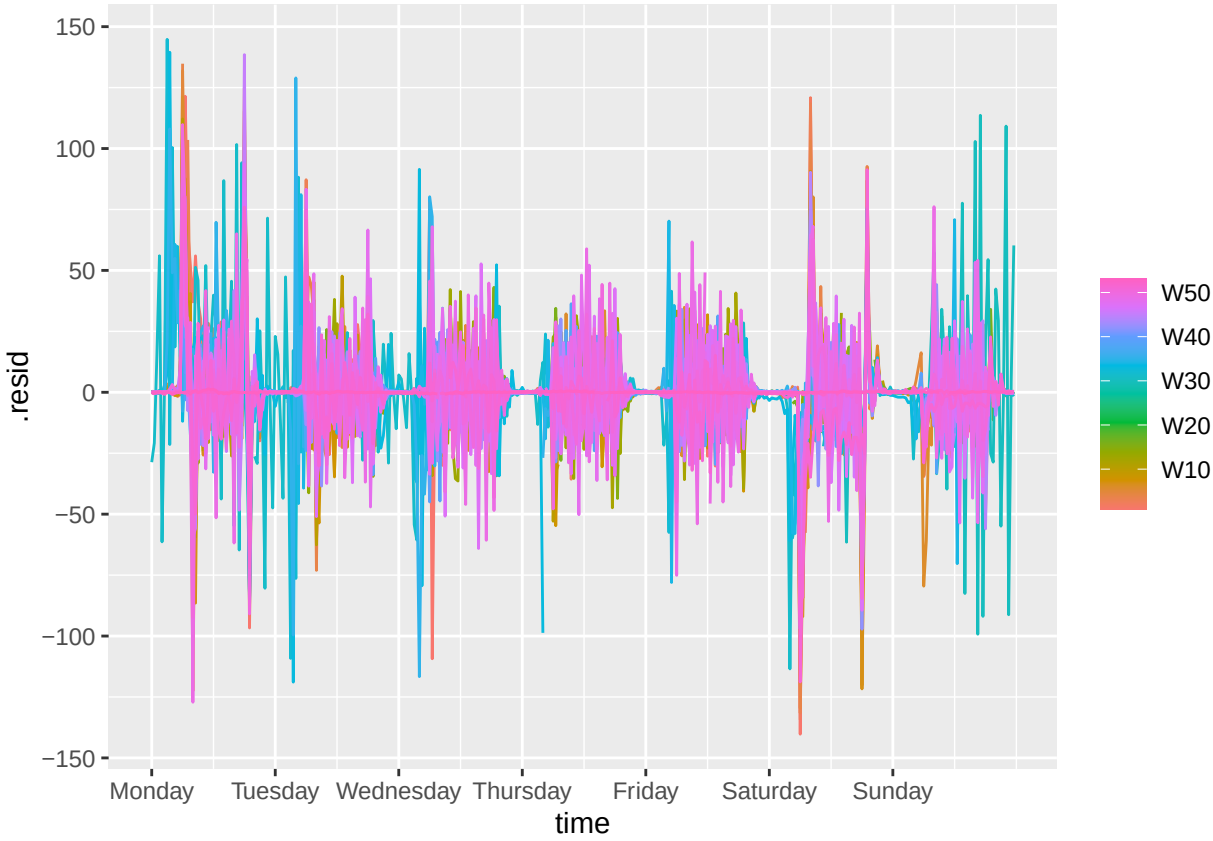
```
## Series: totalEnergy
## Model: LM w/ ARIMA(5,0,0)(1,1,0)[48] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4     ar5     sar1  tempAvg  I(tempAvg^2)
##       0.8993  -0.1631  0.1108  -0.1654  0.1601  -0.2482  -0.3492        0.0162
```

```
## s.e.  0.0090   0.0121  0.0121   0.0121  0.0089   0.0091   0.2344        0.0070
##
## sigma^2 estimated as 249.8:  log likelihood=-53773.75
## AIC=107565.5   AICc=107565.5   BIC=107633.3
```

```
dynReg %>% gg_tsresiduals()
```



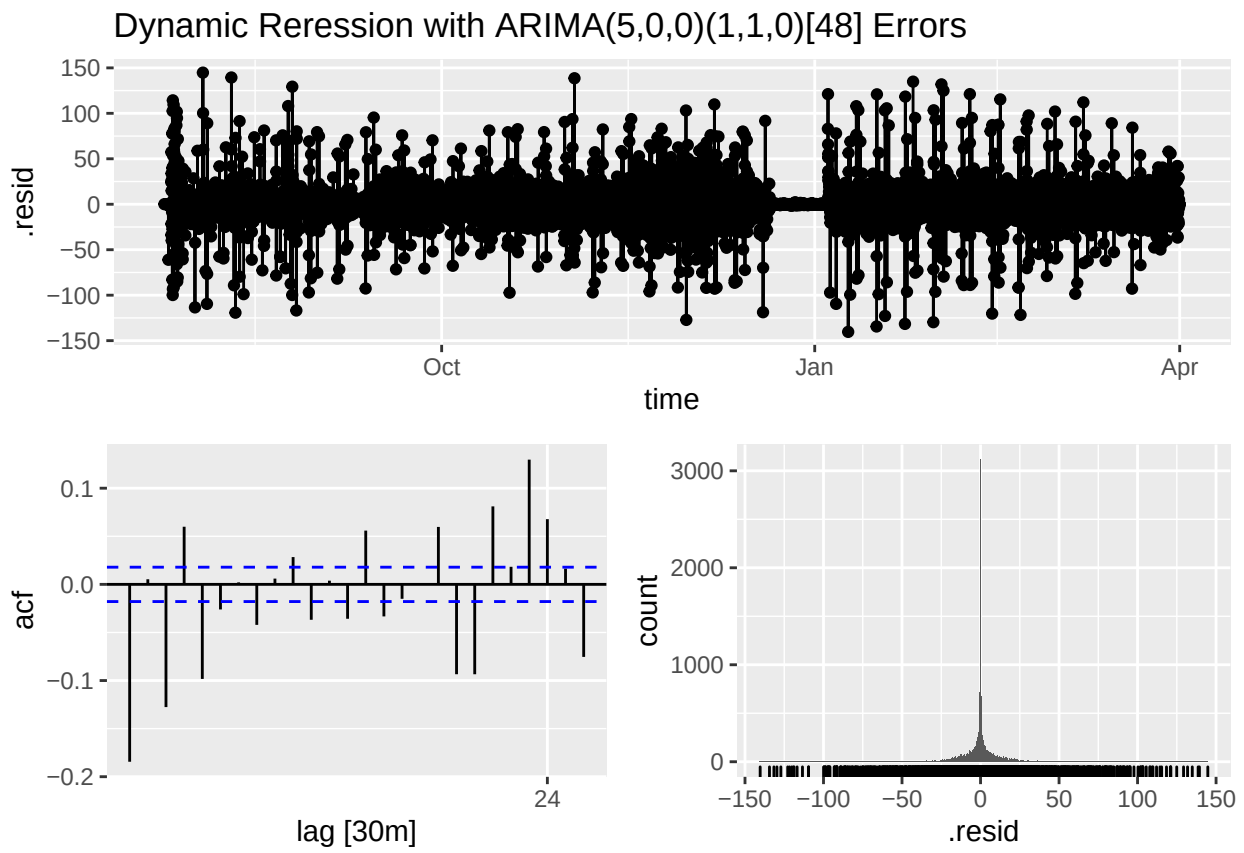

```
residuals(dynReg) %>%
  as_tsibble(key=.model, index=time) %>%
  gg_season(y=.resid, period = "week")
```



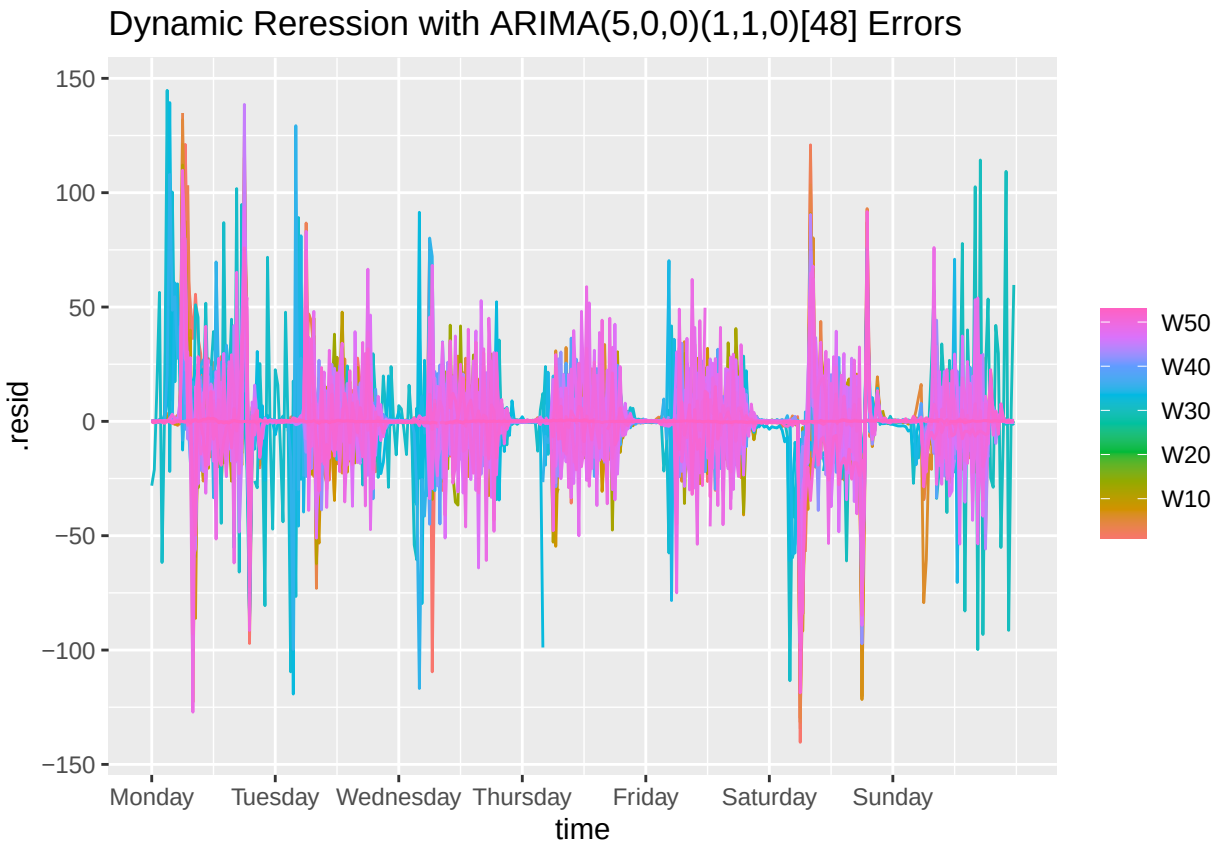

```
# Dynamic regression excluding anything after April 1 to allow for test forecasting
load(file="models/dynRegTrain.Rdata")
# dynRegTrain <- weather_energy %>%
#   fill_gaps() %>%
#   filter(equipment == "AHU2E") %>%
#   filter(time < dmy_h("1-04-2021 00")) %>%
#   model(ARIMA(totalEnergy ~ tempAvg + I(tempAvg^2) + PDQ(period=48)))

dynRegfc <- forecast(dynRegTrain, new_data = weather_energy %>%
  fill_gaps() %>%
  filter(equipment == "AHU2E") %>%
  filter(time >= dmy_h("1-04-2021 00")))

report(dynRegTrain)
```

```
## Series: totalEnergy
## Model: LM w/ ARIMA(5,0,0)(1,1,0)[48] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4     ar5     sar1  tempAvg  I(tempAvg^2)
##       0.9032  -0.1656  0.1168  -0.1735  0.1639  -0.2458  -0.3918        0.0172
## s.e.  0.0094   0.0127  0.0127   0.0128  0.0094   0.0096   0.2496        0.0074
##
## sigma^2 estimated as 272.8:  log likelihood=-49011.45
## AIC=98040.9   AICc=98040.91   BIC=98107.41
```

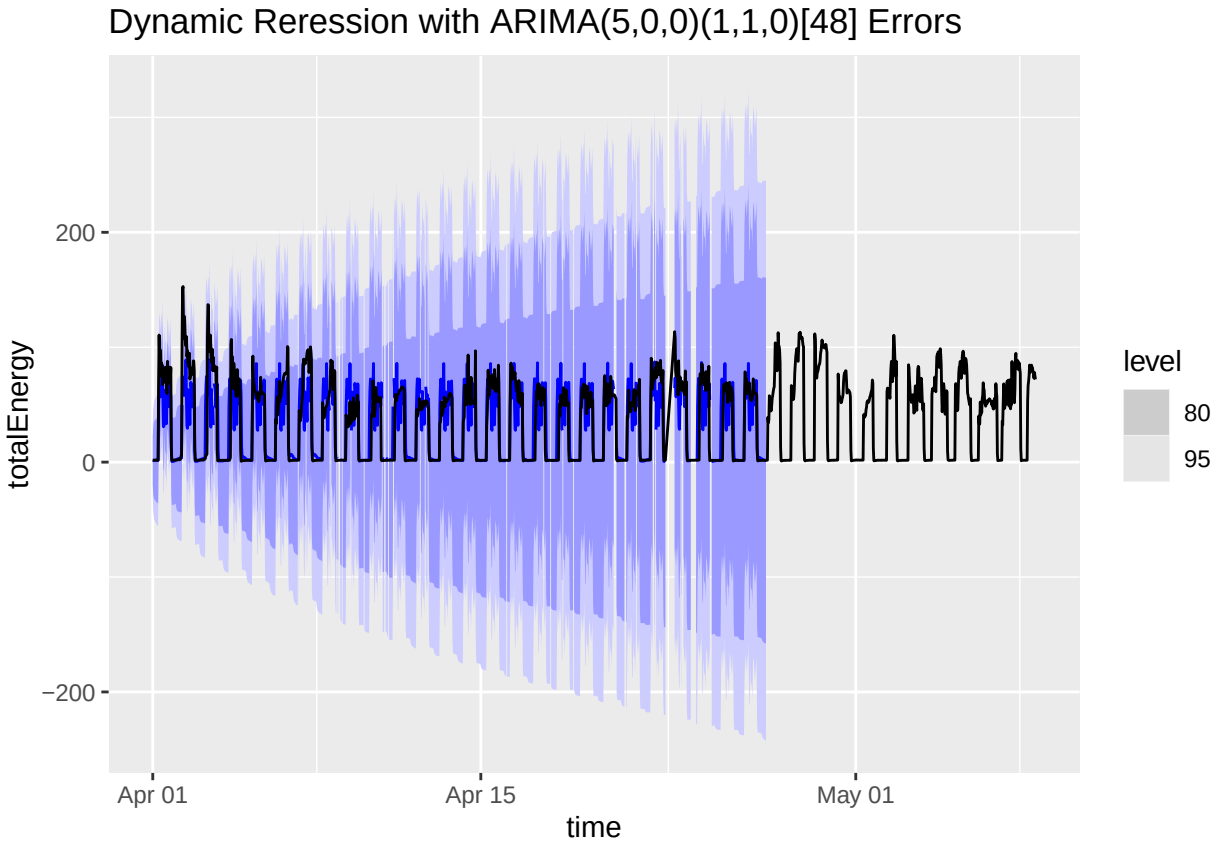
```
dynRegTrain %>% gg_tsresiduals() + ggtitle("Dynamic Reression with ARIMA(5,0,0)(1,1,0)[48] Errors")
```




```
residuals(dynRegTrain) %>%
  as_tsibble(key=.model, index=time) %>%
  gg_season(y=.resid, period = "week") + ggtitle("Dynamic Reression with ARIMA(5,0,0)(1,1,0)[48] Errors
```




```
autoplot(dynRegfc, weather_energy %>% filter(time >= dmy_h("1-04-2021 00"))) + ggtitle("Dynamic Reressi
```



This dynamic regression model seems to be the best at capturing additional variation outside of the daily cycle. Looking at the residuals on a weekly basis over the year, it can be seen that the yearly seasonal component is not being completely accounted for, especially in the changing startup time of the system over the course of the year. Further work in decomposing the daily and yearly trends, potentially with a GAMM or prophet model, could improve performance.

Model of the Data Generating Process

The data generating process for system energy consumption can be modeled as a dynamic regression with ARIMA errors in the form:

y = B_0 + B_1*tempAvg + B_2*tempAvg^2 + e

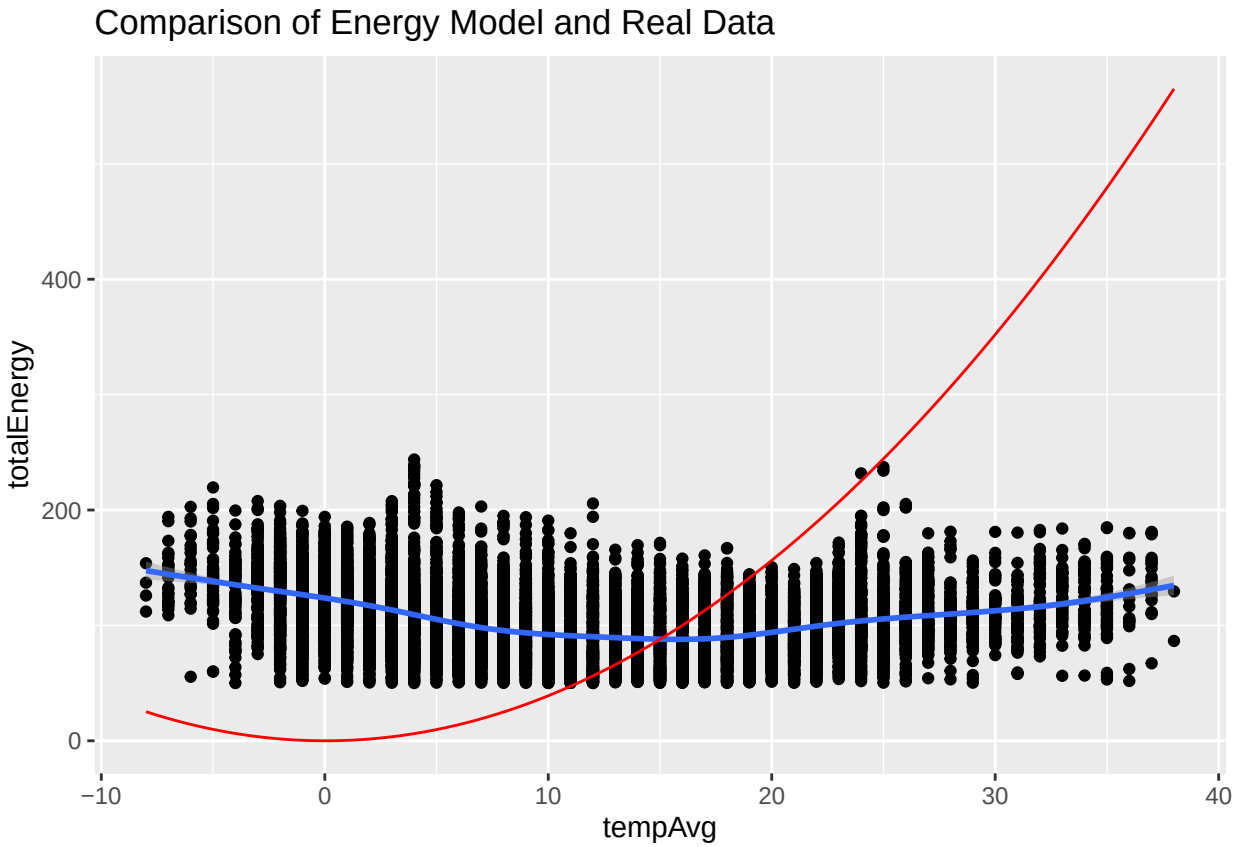
where e is ARIMA (5,0,0,)(1,1,0)[48]

Discussion of the Statistical Model

The provided ARIMA model for errors fits a reasonable understanding of the data generating process. 5 Lag terms in the non-seasonal component fits with the physical limitations of the system which prevent rapid changes in energy consumption. A seasonal lag of 1 with 1 difference fits with the clearly shown daily cycle in the data and differencing creates stationary data out of a potential yearly trend. The linear model component of the dynamic regression can be investigated visually.

Temperature in Model and Acutality

```
tempFunc <- function(x){
  return(0.3917655*x^2 -0.0172378*x)
}

ggplot(data = weather_energy %>% filter(totalEnergy > 50), aes(x=tempAvg)) + geom_point(aes(y=totalEnerg
  geom_function(fun = tempFunc, color ="red") +
  ggtitle("Comparison of Energy Model and Real Data")
```



It seems like the model would likely perform significantly better if the quadratic model were to be shifted such that the minimum were in the 13-18 degree range, as the current model doesn't appear to fit the data as well as possible.

"Chrras/Climateeng Documentation." n.d. https://rdrr.io/github/chrras/climateeng/man/.

"Cooling and Heating Equations." n.d. https://www.engineeringtoolbox.com/cooling-heating-equations-d_747.html.

"Fans - Efficiency and Power Consumption." n.d. https://www.engineeringtoolbox.com/fans-efficiency-power-consumption-d_197.html.

Goetzler, William, Richard Shandross, Jim Young, Oxana Petritchenko, Decker Ringo, Sam McClive, and Building Technologies Office Corporate. 2017. "Energy Savings Potential and RD&D Opportunities for Commercial Building HVAC Systems." DOE/EE–1703, 1419622. https://doi.org/10.2172/1419622.