CWRU DSCI451: Semester Project 4

Alan Curran, Graduate Student, CWRU

December 20th, 2016

Contents

Introduction

Solar power is one of the most prominent types of renewable energy in use today. Recent improvements to the efficiency of photovoltaic (PV) modules as well as massive reductions in the cost to produce the necessary silicon have helped pave the way for the spread of useful and low cost PV power. However, because this is a relatively new technology, there has not been a large amount of research into how well PV modules maintain their power production in a real world outdoor setting. Accelerated lab testing has been fairly standard for some time, where modules are subjected to extreme conditions and their estimated performance lifetime is then extapolated from the results of these tests. It is not necessarily true that these accelerated test provide an adequate estimate for how well a PV module will perform over long periods of time under real world conditions. This uncertainy in the actual performance of PV power has lead to some unexpectedly poor performances of power plants and makes it difficult to reliably invest in the expansion of solar power. To this extent, the Solar Durability and Lifetime Extention Center (SDLE) at Case Western Reserve University (CWRU) has begun to create a database consisting of power data from solar power plants over long periods of time, with the goal of performing statistical modeling to determine the degradation of these systems and the importance of the factors that affect this degradation, such as module brand or climate conditions. This project is a statistical analysis using the month by month method developed by Dr. Yang Hu of the SDLE of a set of PV plants dating back upto 16 years. In the subsequent sections, the process of data collection, cleaing, and analysis will be shown, as well as the preliminary results of the study.

Data Collection

The data in this study was obtained from a partner in the PV industry. It consists of 541 total inverters across 237 sites. An inverter is the device that converts the DC power from PV modules into AC power and feeds it into a power grid. A string of several PV modules are connected in series to a single inverter. The number of modules and size of each module can vary between inverters. The inverter is also usually the location for the power output measurement. For these reasons, the power data for each individual inverter is provided and each inverter will be analyzed as a separate power plant. The site refers to the location of these inverters, in many cases there are multiple inverters at a single site.

The weather data for each inverter is provided with the power data, as the weather is a strong factor in the output of a PV module. The weather conditions are given by the wind spped, ambient temperature, and the global horizontal irradiance (GHI). Future analysis will attempt to isolate the performance of the inverter from the ambient weather conditions in order to determine the degradation occuring over time.

In addition to the inverter data, metadata for the inverters was also provided. The metadata includes information about each inverter such as the brand of the modules, the description of the site, of the latitde and longitiude of the site.

Data Cleaning

There were several problems with the data as it was given. Each .csv data file was for a given site and many included several inverters. There were also some mismatches between the metadata and the actual data, is several cases there was no metadata for an inverter or there was metadata for an inverter with no data. As this data is from industry, the data for each site had to be encrypted. Each inverter in the metadata was given a corresponding 7 digit alpha-numeric key, and the metadata information, such as the module brand, was encrypted with a 32 digit alpha-numeric string. The keys for the encrpytion are stored in a RedCAP database.

The script 1610-master-cleaning-curran.R as well as the function 1610-column-renaming-curran.R were created to isolate each inverter, match it to the metadata, standardize the column names, isolate inverters with missing data or metadata, and rewrite them as individual inverters with all the necessary information encrypted.

Once this process was complete, there were 373 inverters with complete data sets and matching metadata. In serveral cases there was a mismatch between the data and metadata, or the data was missing variables. Many of the inverters did not have any data of the ambient wind speed.

Column Renaming

The following function was created to identify and rename weather columns as well as convert energy data into power data. Most of the inverters provided energy data but not the desired power data. It also calculates the age of the system as a catagorical variable "Month_age" which is the number of calendar months the system has been active.

```
#Alan Curran
#10/2016
#11/4/2016 Alan Curran updated the function for averaging over calendar months instead of 30 days

#function to rename and add new columns in the cleaned single inverter SunPower data
#this function takes in a data file and over writes it with the edits

library(dplyr)

col_cleaning <- function(data) {

  #first rename weather columns
  temp_patt <- "AMB\\.TE"
  wind_patt <- "WSP\\.VAL"
  ghi_patt <- "GLB\\.IR\\.VAL\\_PHOR|PHOR\\.AVG\\.PI|TRK\\.IR\\.VAL\\_HPOA|PPOA\\.AVG\\.PI|IRR\\.CALC\\

  if (TRUE %in% grepl(temp_patt, names(data))) {
    colnames(data)[grepl(temp_patt, colnames(data))] <- "Ambient_temp"
  } else {
     print("There is no temperature data")
}

  if (TRUE %in% grepl(wind_patt, names(data))) {
    colnames(data)[grepl(wind_patt, colnames(data))] <- "Wind_speed"
  } else {
    print("There is no wind speed data")
}

  if (TRUE %in% grepl(ghi_patt, names(data))) {
    colnames(data)[grepl(ghi_patt, colnames(data))] <- "GHI"
  } else {
    print("There is no irradiance data")
}

  #convert timestamp into specific time deliniations
  #this portion was adapted from code written by Yang Hu
  #thanks Yang

  data[,1] <- as.POSIXct(data[,1], format = "%Y-%m-%d %H:%M:%S")
  data$Date <- as.Date(data[,1])
  data$Year <- format(data[,1], format = "%Y")
  data$Month <- format(data[,1], format = "%m")
  data$Day <- format(data[,1], format = "%d")
  data$Time <- format(data[,1], format = "%H:%M")

  #add age into dataframe
  data$origin <- rep(as.Date(data[1,1],format = "%Y/%m/%d"),nrow(data))

    #this age standard has been altered from 30 days to one calendar month
      #day_diff <- as.data.frame(data$Date - data$origin)
      #days_working <- as.numeric(sapply(day_diff,as.character))

      ##the data will be combined later into an average over one month to make analysis easier
      ##one month is regarded as 30 days in operation
      #months_working <- ceiling(days_working/30L)
      #data$Age <- months_working

  data$Month_year <- format(data[,1], format = "%Y/%m")
  elapsed_months <- function(end_date, start_date) {
    ed <- as.POSIXlt(end_date)
    sd <- as.POSIXlt(start_date)
    12 * (ed$year - sd$year) + (ed$mon - sd$mon)
  }

  data$Month_age <- elapsed_months(data$Date, data$origin)

  #change power or energy name
  power_patt <- "KW\\.|InverterPower|KW\\_"
  energy_patt <- "KWHINT\\."

  #inverters with names like X2 or X3 have been attatched to inverter names with
  #proper unit of output in the initial cleaning script
  #the above patterns will be in their names

  if (TRUE %in% grepl(power_patt, names(data))) {
    colnames(data)[grepl(power_patt, colnames(data))] <- "AC_power"
  } else {
    print("There is no power data")
  }

  if (TRUE %in% grepl(energy_patt, names(data))) {
    colnames(data)[grepl(energy_patt, colnames(data))] <- "AC_energy"

    #add in a calculated power column knowing the energy produced over the timestep
    #15 mins timestamp steps, kW(during 15 mins) = 4*kWh(produced over 15 mins) because there are 4 15 
    data$AC_power <- data$AC_energy/0.25
  } else {
    print("There is no energy data")
  }

  return(data)

}
```

Master Cleaning Script

The following script took each site file in a folder, isolated the inverters,properly named them, and saved them to a separate folder as .csv files. This script utilizes the column renaming funciton.

```
#Alan Curran
#10/2016

##################################################################################################
##
##  serves as a master cleaning script for the BAPVC SunPower data
##  this is run on a data set where files missing columns have already been moved to "Weird data" folde
##  this script runs initial-cleaning-curran with the addition of column editing using the col_cleaning
##  in addition to removing duplicate columns for better data matching
##  and using the encrpyted file names when writing
##  this the output of this will be ready for direct ingestion
##
###################################################################################################

library(dplyr)
source('H:/Git/dsci451alancurran/code/1610-column-renaming-curran.R', echo = FALSE)

setwd("V:/vuv-data/proj/BAPVC-TSA/SunPower")

metadata <- read.csv("SunPower_modules_subset_modified.csv", colClasses = c(NA, NA, NA, rep("NULL", 16)

metadata$Project.. <- gsub("$", ".csv", metadata$Project..)

files <- list.files(path = "copy/data-csv", pattern = ".csv")

inverter_tracker <- NULL
no_inverter_data <- NULL
data_meta_mismatch <- NULL

#loop over every data file
for (i in files) {
```

```
#select inverter names in metadata for the given file
meta_names <- filter(metadata, Project.. == i)

#read in data and remove duplicate columns
data <- read.csv(paste("copy/data-csv/", i, sep = ""))
data <- data[!duplicated(lapply(data, summary))]

data_names <- names(data)[-c(1:4)]

inverter_tracker <- rbind(inverter_tracker, c(i, length(data_names), nrow(meta_names)))

#same count of inverter data in meta and real data
if (length(data_names) == nrow(meta_names)) {

  for (j in 1:nrow(meta_names)) {

    #first 4 columns are not inverter data
    data_col <- data[(4 + j)]

    #catch any inverters that have names like X2
    #attatch the first inverter name to this name so they can later be identified as power or energy
    if (grepl("X\\d+$", colnames(data_col)) == TRUE) {
      colnames(data_col) <- paste(names(data)[5], names(data_col), sep = "_")
    }

    #create a data set with climate info and only one inverter
    data_combine <- cbind(data[(1:4)],data_col)

    #prewritten function to clean, rename and sort columns
    #if this returns an error, the data has an irregular timestamp
    try(data_combine <- col_cleaning(data_combine), silent = FALSE)

    write.csv(data_combine, file = paste("copy/data-cleaned/", as.character(meta_names[j,1]), ".csv",

  }
}
#if there are no real data columns where there should be inverter data
 else if (length(data_names) == 0) {

    no_inverter_data <- rbind(no_inverter_data,i)
 }

# remainer are files with inverter counts different than in the metadata
 else {

    for (k in 1:(length(data_names))) {

      #split the real data name and recombine to match the metadata format
      data_split <- strsplit(data_names[k],"\\.")
      data_split <- unlist(data_split)
      data_meta_match <- paste(data_split[1:3], collapse = "_")

      #if the real data and metadata share a matching inverter pull and write that specific column
```

```
        if (data_meta_match %in% metadata$Project.Name) {

          data_col <- data[(4 + k)]
          data_combine <- cbind(data[(1:4)],data_col)
          #print(paste(i,"is good at k =", k, sep = " "))

          try(data_combine <- col_cleaning(data_combine), silent = FALSE)

          write.csv(data_combine, file = paste("copy/data-cleaned/", as.character(metadata$record_id[me
        }

        #if there is a column in the data that is not named in the metadata do not write it but track i
        else{

          print(paste("there is a data mismatch in", i, "at k =", k, sep = " "))
          data_meta_mismatch <- rbind(data_meta_mismatch, paste(i, data_names[k], sep = "_"))

        }

      }

   }

  print(i)

}

inverter_tracker <- as.data.frame(inverter_tracker)
colnames(inverter_tracker) <- c("file", "inverters_in_data", "inverters_in_meta")
inverter_tracker$difference <- as.numeric(as.character(inverter_tracker[,2])) - as.numeric(as.character
write.csv(inverter_tracker, file = "inverter_tracking.csv", row.names = FALSE)
write.csv(data_meta_mismatch, file = "data_meta_mismatch.csv", row.names = FALSE)
```

Data Book

The result of this script is 373 .csv files, one for each inverter. The name of each file is the 7 digit alpha-numeric encrypted name of the inverter. Each vaiable in the file is a column, with the rows being each measurement, taken every 15 minutes. Below of the data book for this data set.

- All data follows the following naming convention
  - Timestamp - time and date when reading was taken
  - Ambient_temp - outside temperature (C)
  - Wind_speed - speed of wind (m/s)
  - GHI - global horizontal irradiance (W/m^2)
  - AC_power - power production (kW) (if there was no power data this was calculated from the energy data)
  - AC_energy - energy produced (kWh) (if the energy was given)
  - Month_age - time in calendar months since the start of opertion (integer value)

There are additional columns with different formats of the system age, however these are not used in the analysis and are mearly there in case an alternate time deliniation is desired.

Analysis

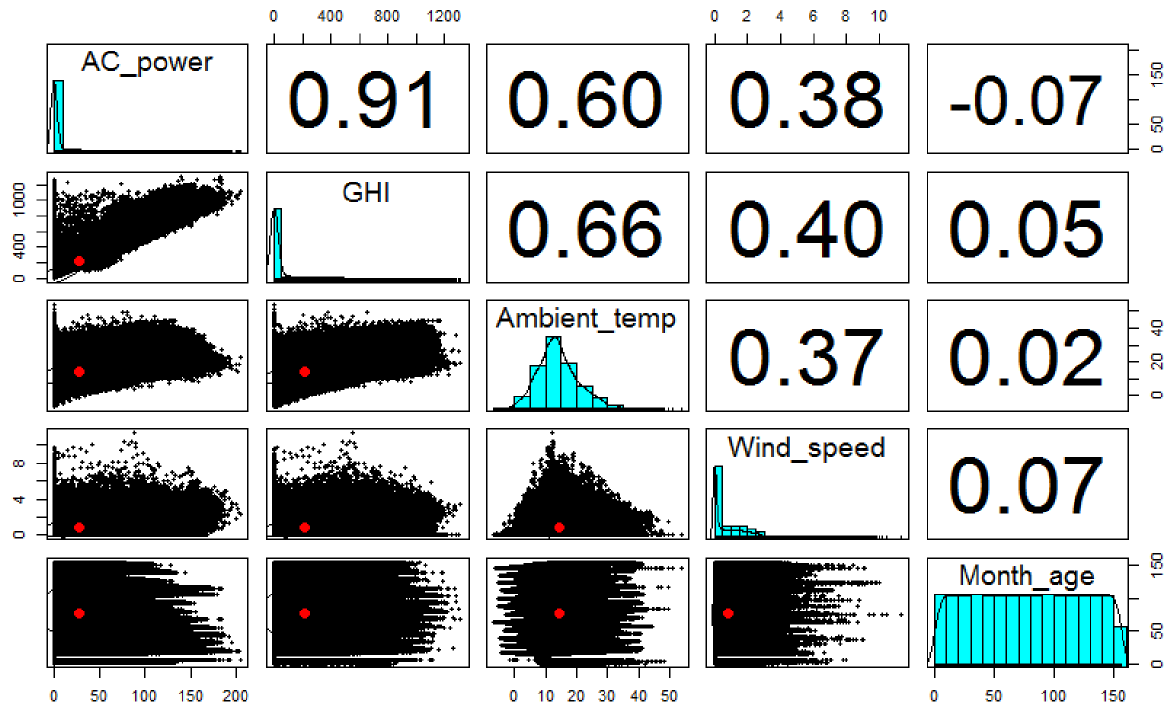


Figure 1: Pairs plot of the raw data for inverter 4fmk53x.

This pairs plot shows the distribution and correlation coefficients for one of the inverters (4fmk53x). As is expected, the strongest correlation is between the power output and the GHI. There is a lesser correlation between power and temperature and wind speed. Temperature and wind speed have very similar correlations to GHI as they do with power. This is most likely coincidental given the close relationship between GHI and power. The pairs plot also shows some additional cleaning will be required, as there are some zero values across all the variables and serveral times when power is zero when GHI is nonzero.

Additional Data Cleaning

The following conditions were removed from the data before analysis was performed - All rows with NA values in the GHI, Ambient_temp, or timestamp - All rows where the GHI was less than 50 W/m^2, these are determined to be nighttime readings - All duplicated rows - All rows where GHI is low while power output is high

The code for this is included in the analysis function but is shown below.

```r
#remove night time readings (GHI < 0) and extreme GHI readings
data <- data[!(data$GHI < 50) & !(data$GHI > 2000),]

#subset and remove the observations where GHI is "high" while DC or AC is "low"
sub1 <- data[which((data$AC_power < 0.01*max(data$AC_power, na.rm = TRUE)) & (data$GHI > 50)),]
data <- data[!(data$Date %in% unique(sub1$Date)),]
```

```r
#remove NA values
data <- data[!is.na(data$Timestamp),]
data <- data[!is.na(data$AC_power),]
data <- data[!is.na(data$Ambient_temp),]

#remove duplicate rows
data <- data[!duplicated(data[,-c(1,6:13)]),]
```

The Month by Month Method

The performance of the system must be isolated from the weather conditions if the degradation over time is to be determined. The month by month method (MbyM) has been developed to do this. First, an assumption is made that there is no significant degradation over the course of 1 calendar month. Second, the data is sutset by the age the the system in calendar months. Third, the linear corrolation between the weather data (GHI, temperature, and wind speed) and the power data is calculated for each month, as shown by the equation below.

$$PredictedPower = \beta_0 * (GHI) + \beta_1 * (Temperature) + \beta_2 * (WindSpeed) + \varepsilon$$

Fourth, standard weather conditions for each inverter are calculated as the average temperature, the average wind speed, and the minimum value of the maximum irradiance across each month. Lastly, the standard weather conditions are fed into each monthly model for the weather and power. This produces a predicted power value that would correspond to the system remaining at constant weather conditions over its lifetime.

Because this predicted power value was determined at constant weather conditions, any change over time is attributed to the degradation of the system.

predictive_model Function

Below shows the code for the MbyM analysis. It was written as a function with the input being a data file and the name of the data file (used to organize the results). This funciton performs the MbyM method on a data set, then fits a linear trend to the resulting predictive power model to determine the linear degradation. It then saves the linear degradation fit, as well at the statistics relating to the goodness of the fit as an output. The value of the slope of the linear fit is used to determine the degradation of the system.

```r
#Alan Curran
#this function imports data for an individual inverter and the name of the inverter
#it cleans the data and performs month by month modeling on the data

predictive_model <- function(data, inverter_name){


  #initial data filtering, removeing power values that do not properly correlate to GHI values

  #data_raw <- data

  #remove NA timestamp readings
  data <- data[!is.na(data$Timestamp),]
  data <- data[!is.na(data$AC_power),]

  #remove night time readings (GHI < 0) and extreme GHI readings
  data <- data[!(data$GHI < 50) & !(data$GHI > 2000),]
  data <- data[!(data$AC_power > 2000),]

  #subset the observation which GHI is "high" while DC or AC is "low"
  sub1 <- data[which((data$AC_power < 0.01*max(data$AC_power, na.rm = TRUE)) & (data$GHI > 50)),]
  data <- data[!(data$Date %in% unique(sub1$Date)),]

  data <- data[!is.na(data$Timestamp),]
  data <- data[!is.na(data$AC_power),]
  data <- data[!is.na(data$Ambient_temp),]

  #remove duplicate rows
  data <- data[!duplicated(data[,-c(1,6:13)]), ]

  # #now split the data into testing and training sets
  # training_data <- NULL
  # for (i in 1:max(data$Month_age, na.rm = TRUE)) {
  #
  #   sub <- subset(data, age == i)
  #   set.seed(n); test <- sample(nrow(sub),floor(nrow(sub)*0.1))
  #   training_data <- rbind(training_data, sub[-test,])
  # }
  #
  # test_data <- NULL
  # for (i in 1:max(data$Month_age, na.rm = TRUE)) {
  #
  #   sub <- subset(data, age == i)
  #   set.seed(n); test <- sample(nrow(sub),floor(nrow(sub)*0.1))
  #   test_data <- rbind(test_data, sub[test,])
  # }

  #define constant values to be applied to each monthly model

  #average wind speed
  Wind_speed <- mean(data$Wind_speed, na.rm = TRUE)

  #average temp
  Ambient_temp <- mean(data$Ambient_temp, na.rm = TRUE)

  #minimum of the max irradiance values for each month
  max_irrad <- NULL
  for (i in unique(data$Month_age)) {
    sub <- subset(data, Month_age == i)
    GHI_max <- max(sub$GHI, na.rm = TRUE)
    max_irrad <- rbind(max_irrad, GHI_max)
  }

  GHI <- min(max_irrad, na.rm = TRUE)

  power_predict <- NULL
  models <- list()
  environ_cond <- data.frame(GHI, Ambient_temp, Wind_speed)

  #make NA wind speed values equal to the average wind speed to prevent errors
```

```r
data$Wind_speed[is.na(data$Wind_speed)] <- Wind_speed

#create a predictive model for each month
for (i in 1:max(data$Month_age, na.rm = TRUE)) {

  combine <- subset(data, Month_age == i)
  if(nrow(combine) > 30) {
  model <- lm(AC_power ~ GHI + Ambient_temp + Wind_speed, data = combine)
  power_predict[i] <- predict(model, environ_cond)
  models[[i]] <- model
  } else
    models[[i]] <- NA

}

#plot constant monthly predictive model with slope being the degradation rate
months <- 1:max(data$Month_age, na.rm = TRUE)

#use this line only if you want to save the plots
png(file = paste("H:/Git/dsci451alancurran/figs/1612_sample1_", inverter_name, ".png", sep = ""))

title = paste(inverter_name, "Irradiance", round(GHI, digits = 1), "w/m2 Ambient Temperature", round(
graph <- plot(power_predict ~ months, ylim = c(0,max(power_predict,na.rm = TRUE)*1.1), xlab = "System

# Esd<-NULL
# for (i in 1:max(train_set$age, na.rm = TRUE)){
#   sub_test<- subset(test_set, age == i)
#   if(!is.na(lms[[i]])){
#     sub_test$predicted<- predict(lms[[i]],sub_test)
#     sub_test$err<-(sub_test$AC_power - sub_test$predicted)
#     Esd<-c(Esd,sd(sub_test$err))
#   }
#   else Esd<-c(Esd, NA)
# }
#
# graph <- graph + arrows(x, power_predict-Esd, x, power_predict+Esd, length=0.05, angle=90, code=3)

Rd <- lm(power_predict ~ months)
graph <- abline(Rd, col = "blue")

#use this line only if you want to save the plots
dev.off()

stats <- data.frame(inverter_name, Rd$coefficients[[2]], summary(Rd)$r.squared, summary(Rd)$adj.r.squa
colnames(stats) <- c("inverter_name", "slope", "r.squared", "adj.r.squared")

return(stats)
```

Results

Sample Inverter

The inverter 4fmk53x will be shown as an example of the analysis process. As can be seen in the code below, once the scripts and functions have been established, performing an analysis is trivial.

```r
source("H:/Git/dsci451alancurran/code/1611-data-filtering-predictive-fitting.R")
data <- data <- read.csv("V:/vuv-data/proj/BAPVC-TSA/SunPower/copy/data-cleaned/4fmk53x.csv")

predictive_model(data, inverter_name = "4fmk53x")
```

```
##   inverter_name      slope r.squared adj.r.squared
## 1       4fmk53x -0.2399638 0.5451956     0.5420372
```

The graph of the predicted power output shows some seasonality still but there is also a clear decreasing trend over time. This decrease shows reducing performance under the same weather conditions as time goes on. The rate of degradation can be represented by the slope of this linear fit. The adjusted r squared is fairly low, only 0.54, however this can be attributed to the remaining seasonality in the prediction.

Total Data Results

In previous studies, the climate zone was found to be the most significant factor in the degradation of PV plants. The climate zones are catagorized by the Koppen-Geiger climate zone classification system.

The following code separates the results of the analysis on every inverter and sorts the variation of slopes by climate zone. A more negative slope is indiciative of a greater segradation rate. Positive slopes are also possible. If a system is not performing at peak when it first started, or not all of the modules are connected until later, a positive slope at the beginning of the inverters lifetime will be observed. In the analysis, only climate zones with more than 30 inverters will be compared. A table can easily show how many inverters there are in each climate zone.

```
## 
##  As BSh BSk BWh BWk Cfa Cfb Csa Csb Dfa Dfb 
##   2   2  25   3  22  38   1  38  98   2   1
```

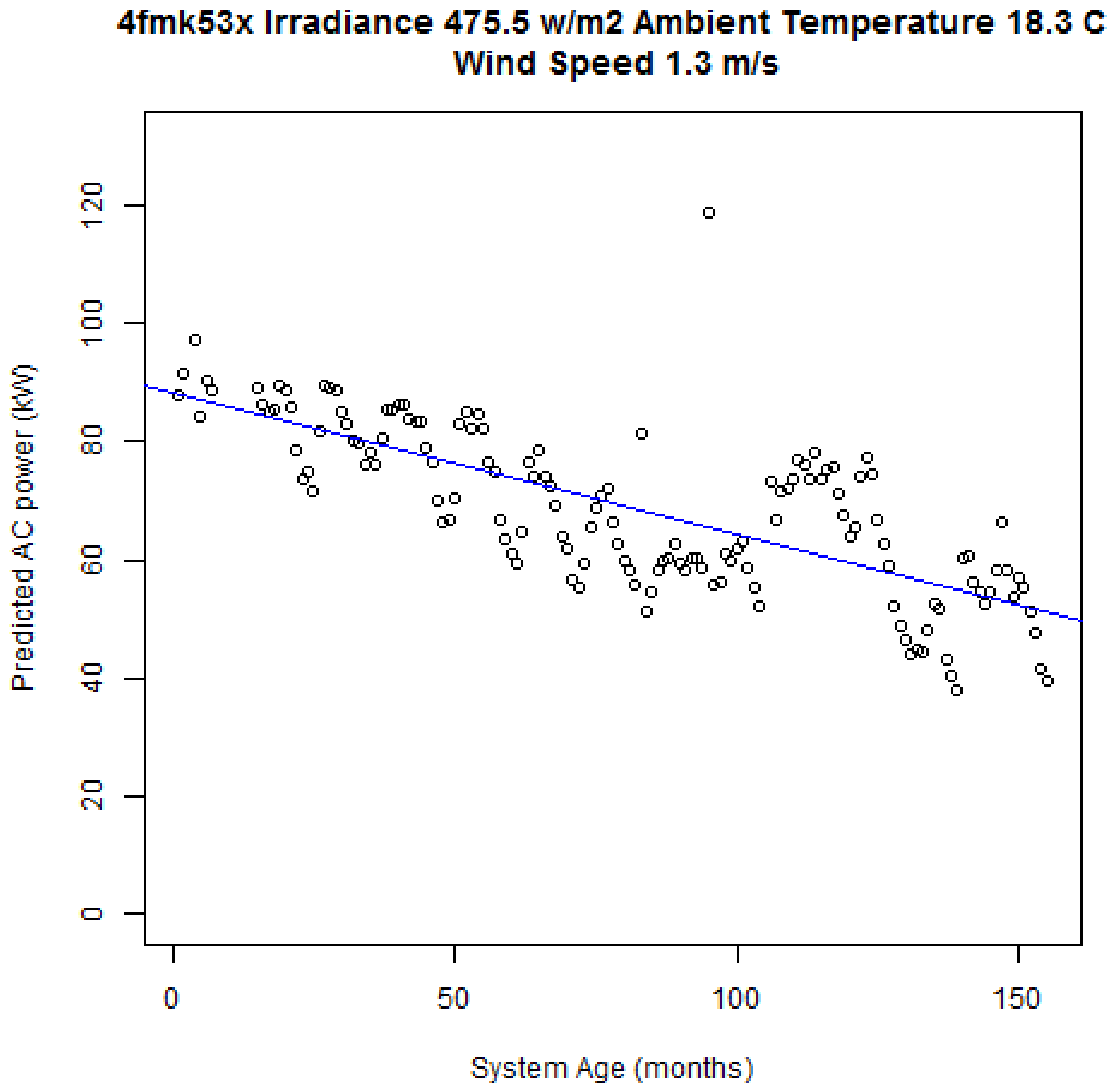
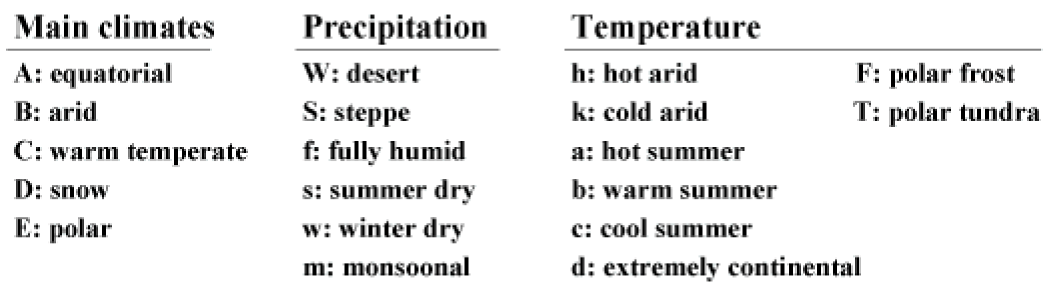
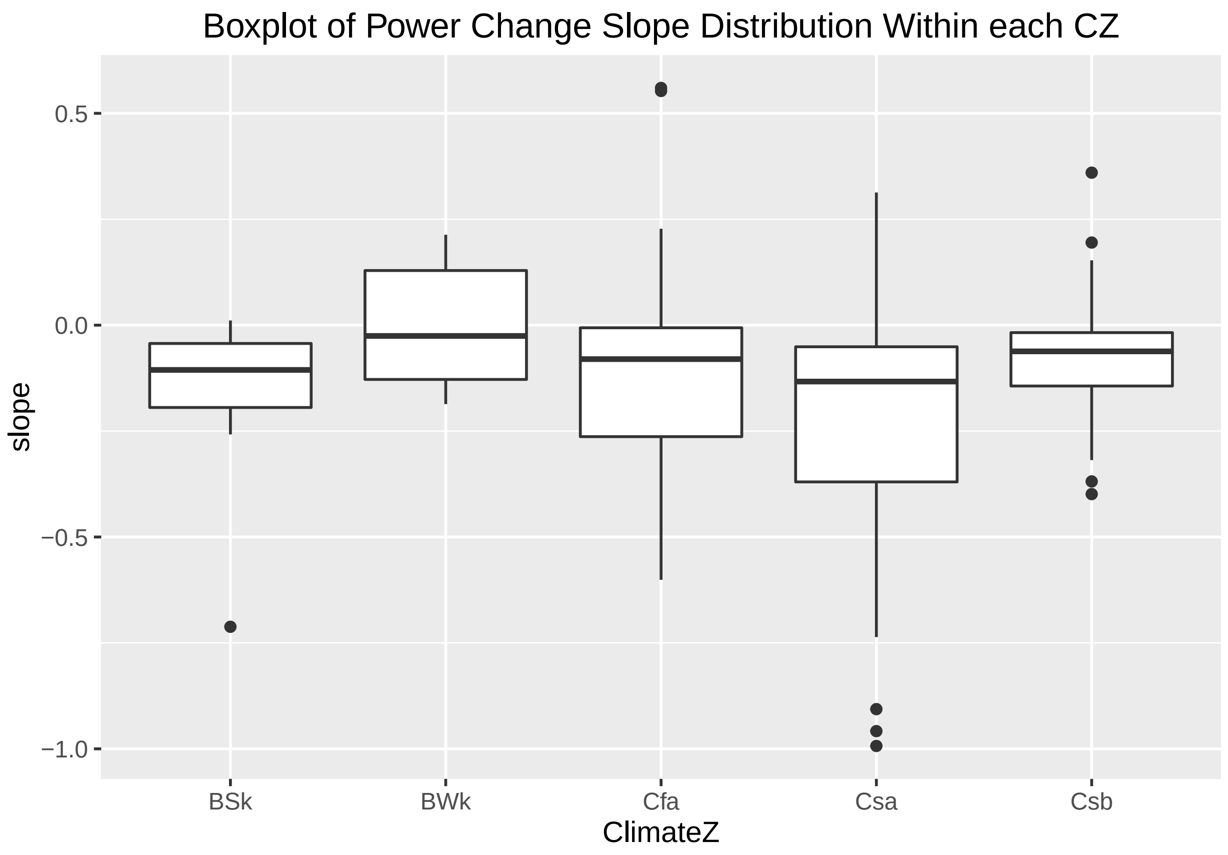


Figure 2: Graph of the predicted power for inverter 4fmk53x with the linear regression included.



Main climates	Precipitation	Temperature	
A: equatorial	W: desert	h: hot arid	F: polar frost
B: arid	S: steppe	k: cold arid	T: polar tundra
C: warm temperate	f: fully humid	a: hot summer	
D: snow	s: summer dry	b: warm summer	
E: polar	w: winter dry	c: cool summer	
	m: monsoonal	d: extremely continental	

Figure 3: Koppen-Geiger climate zone classification.



The results appear to show different distribution in the changing rate based on the climate zones. There are still advances that need to be made in the statistical side of these results. Firstly, because the predicted power outputs still retain some of their seasonality, the adjusted r squared values are low for these curves. It is also possible than some of these inverters would be better fit with a multiple linear regression, as they did not degrade consistently over their lifetimes. This could be caused by a module or inverter being switched out or repaired. The aim of this project was largely to organize this data set and show to MbyM method as a proof of concept. The next step will be to refine the statistical analysis of the results to better determine the changing rate of these inverters.

Code book

Three scripts were used in this project - 1610-column-renaming-curran.R is a function that controls the naming of the variables and the month_age introduction - 1610-master-cleaning-curran.R is a script used to sort and order the inverters, specifically designed for this data set - 1611-data-filtering-predictive-fitting.R is a function that removes unwanted rows and performs the MbyM analysis on an inverter