Renewable energy generator forecast

Outline

1. About the research

The data from Electronics Testing Center, Taiwan. Due to the limited dimensions, date from solar generator included DOWNWARD SHORT WAVE FLUX AT GROUND SURFACE(SWDOWN) and electricity production (PAC), and data from wind drive generator included 10 meter high of wind speed(WS10m), 65 meter high of wind speed (WS65m) and electricity production(PAC).

本次研究案電子檢驗中心提供兩種發電數據:風力發電及太陽能發電，由於本次提供數據維度有限，太陽能發電的數據僅能從地表輻射(SWDOWN)和發電量(PAC)觀察;風力發電僅能參考發電機10公尺觀測到的風速(WS10m)、65公尺觀測到的風速(WS65m)及發電量(PAC)三個變數。

2. Data preparation for solar generator

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.4.4
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(dygraphs)
```

```
## Warning: package 'dygraphs' was built under R version 3.4.4
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

```
## Loading required package: lattice
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.4.4
```

```
## corrplot 0.84 loaded
```

```
library(Metrics)
```

```
## Warning: package 'Metrics' was built under R version 3.4.4
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.4.4
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.4
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(xts)
```

```
## Warning: package 'xts' was built under R version 3.4.4
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.4.4
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

2.1 loading data

```
chunghwa <- read.csv("./Data_1105/14.csv")
chunghwa$location <- "chunghwa"
dreamhouse <- read.csv("./Data_1105/19.csv")
dreamhouse$location <- "dreamhouse"
futurehouse <- read.csv("./Data_1105/18.csv")
futurehouse$location <- "futurehouse"
xinglongmarket <- read.csv("./Data_1105/39.csv")
xinglongmarket$location <- "xinglongmarket"
nuclearresearch <- read.csv("./Data_1105/40.csv")
nuclearresearch$location <- "nuclearresearch"
cbso <- rbind(chunghwa,dreamhouse,futurehouse,xinglongmarket,nuclearresearch)
rm(chunghwa,dreamhouse,futurehouse,xinglongmarket,nuclearresearch)

names(cbso)
```

```
##  [1] "SESSIONS"      "RANK"          "MYTIME"        "MYHOUR"
##  [5] "INITIAL_DATE"  "FCST"          "STNNO"         "FCST_DATETIME"
##  [9] "WS10M"         "WS65M"         "SWDOWN"        "USER_ID"
## [13] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [17] "RAIN"          "PAC"           "VAC"           "IAC"
## [21] "EAC"           "location"
```

```
names(cbso)[c(1:2,4:8,12:17)]
```

```
##  [1] "SESSIONS"      "RANK"          "MYHOUR"        "INITIAL_DATE"
##  [5] "FCST"          "STNNO"         "FCST_DATETIME" "USER_ID"
##  [9] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [13] "RAIN"
```

```
cbso <- cbso[,-c(1:2,4:8,12:17)]
summary(cbso)
```

```
##            MYTIME         WS10M              WS65M
##  2018/09/01 00:   5   Min.   :   0.043   Min.   :-999.0000
##  2018/09/01 01:   5   1st Qu.:   1.232   1st Qu.:   1.7948
##  2018/09/01 02:   5   Median :   2.197   Median :   3.0540
##  2018/09/01 03:   5   Mean   :   5.789   Mean   :   0.7302
##  2018/09/01 04:   5   3rd Qu.:   3.125   3rd Qu.:   4.2880
##  2018/09/01 05:   5   Max.   :1412.799   Max.   :  10.3460
##  (Other)      :6418
##      SWDOWN            PAC                VAC             IAC
##  Min.   :-999.0   Min.   :    0.00   Min.   :191.1   Min.   :  0.000
##  1st Qu.:   0.0   1st Qu.:    0.00   1st Qu.:221.2   1st Qu.:  0.400
##  Median :   0.0   Median :   53.33   Median :226.6   Median :  0.825
##  Mean   : 165.5   Mean   : 4779.09   Mean   :274.1   Mean   : 12.608
##  3rd Qu.: 253.4   3rd Qu.: 5401.49   3rd Qu.:230.9   3rd Qu.: 13.963
##  Max.   : 988.9   Max.   :48605.86   Max.   :447.6   Max.   :126.870
##
##       EAC          location
##  Min.   : 5317   Length:6448
##  1st Qu.:17941   Class :character
##  Median :32655   Mode  :character
##  Mean   :29537
##  3rd Qu.:37748
##  Max.   :49195
##
```

2.2 preparing time series

```
head(cbso$MYTIME)
```

```
## [1] 2018/09/01 00 2018/09/01 01 2018/09/01 02 2018/09/01 03 2018/09/01 04
## [6] 2018/09/01 05
## 1503 Levels: 2018/09/01 00 2018/09/01 01 2018/09/01 02 ... 2018/11/05 10
```

```
date <- substr(cbso$MYTIME, 1,10)
time <- paste(substr(cbso$MYTIME,12, 13),":00:00",sep = "")
cbso$MYTIME <- as.POSIXct(paste(date,time, sep = " "))

cbso$location <- factor(cbso$location)

summary(cbso)
```

```
##      MYTIME                        WS10M             WS65M
##  Min.   :2018-09-01 00:00:00   Min.   :   0.043   Min.   :-999.0000
##  1st Qu.:2018-09-15 11:00:00   1st Qu.:   1.232   1st Qu.:   1.7948
##  Median :2018-10-01 18:00:00   Median :   2.197   Median :   3.0540
##  Mean   :2018-10-02 02:39:42   Mean   :   5.789   Mean   :   0.7302
##  3rd Qu.:2018-10-18 00:00:00   3rd Qu.:   3.125   3rd Qu.:   4.2880
##  Max.   :2018-11-05 10:00:00   Max.   :1412.799   Max.   :  10.3460
##      SWDOWN            PAC                VAC             IAC
##  Min.   :-999.0   Min.   :    0.00   Min.   :191.1   Min.   :  0.000
##  1st Qu.:   0.0   1st Qu.:    0.00   1st Qu.:221.2   1st Qu.:  0.400
##  Median :   0.0   Median :   53.33   Median :226.6   Median :  0.825
##  Mean   : 165.5   Mean   : 4779.09   Mean   :274.1   Mean   : 12.608
##  3rd Qu.: 253.4   3rd Qu.: 5401.49   3rd Qu.:230.9   3rd Qu.: 13.963
##  Max.   : 988.9   Max.   :48605.86   Max.   :447.6   Max.   :126.870
##       EAC                   location
##  Min.   : 5317   chunghwa       :1503
##  1st Qu.:17941   dreamhouse     :1503
##  Median :32655   futurehouse    :1503
##  Mean   :29537   nuclearresearch: 436
##  3rd Qu.:37748   xinglongmarket :1503
##  Max.   :49195
```

2.3 removing missing data

```
table(cbso$WS65M=="-999")
```

```
##
## FALSE  TRUE
##  6432    16
```

```
cbso <- subset(cbso,WS65M!="-999")
summary(cbso)
```

```
##      MYTIME                        WS10M           WS65M
##  Min.   :2018-09-01 00:00:00   Min.   :0.043   Min.   : 0.086
##  1st Qu.:2018-09-15 10:00:00   1st Qu.:1.229   1st Qu.: 1.803
##  Median :2018-10-01 16:00:00   Median :2.192   Median : 3.058
##  Mean   :2018-10-02 01:19:24   Mean   :2.289   Mean   : 3.217
##  3rd Qu.:2018-10-17 22:00:00   3rd Qu.:3.119   3rd Qu.: 4.290
##  Max.   :2018-11-05 10:00:00   Max.   :9.142   Max.   :10.346
##      SWDOWN           PAC             VAC             IAC
##  Min.   :  0.0   Min.   :    0   Min.   :191.1   Min.   :  0.00
##  1st Qu.:  0.0   1st Qu.:    0   1st Qu.:221.2   1st Qu.:  0.40
##  Median :  0.0   Median :   50   Median :226.6   Median :  0.81
##  Mean   :168.4   Mean   : 4784   Mean   :274.1   Mean   : 12.62
##  3rd Qu.:254.2   3rd Qu.: 5408   3rd Qu.:230.9   3rd Qu.: 13.97
##  Max.   :988.9   Max.   :48606   Max.   :447.6   Max.   :126.87
##       EAC                   location
##  Min.   : 5317   chunghwa       :1499
##  1st Qu.:17929   dreamhouse     :1499
##  Median :32651   futurehouse    :1499
##  Mean   :29523   nuclearresearch: 436
##  3rd Qu.:37696   xinglongmarket :1499
##  Max.   :49195
```

2.4 standardizing data

```
cbso_sc <- cbso[,c(2:5)]
cbso_sc <- scale(cbso_sc)
cbso_z <- as.data.frame(cbso_sc)
cbso_z <- cbind(cbso$MYTIME,cbso$location,cbso_z)
colnames(cbso_z) <- c("mytime","location","ws10m","ws65m","swdown","pac")
summary(cbso_z)
```

```
##      mytime                              location        ws10m
##  Min.   :2018-09-01 00:00:00   chunghwa       :1499   Min.   :-1.70944
##  1st Qu.:2018-09-15 10:00:00   dreamhouse     :1499   1st Qu.:-0.80698
##  Median :2018-10-01 16:00:00   futurehouse    :1499   Median :-0.07387
##  Mean   :2018-10-02 01:19:24   nuclearresearch: 436   Mean   : 0.00000
##  3rd Qu.:2018-10-17 22:00:00   xinglongmarket :1499   3rd Qu.: 0.63185
##  Max.   :2018-11-05 10:00:00                          Max.   : 5.21567
##      ws65m              swdown             pac
##  Min.   :-1.77558   Min.   :-0.6335   Min.   :-0.55077
##  1st Qu.:-0.80190   1st Qu.:-0.6335   1st Qu.:-0.55077
##  Median :-0.09021   Median :-0.6335   Median :-0.54502
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.60872   3rd Qu.: 0.3228   3rd Qu.: 0.07185
##  Max.   : 4.04270   Max.   : 3.0864   Max.   : 5.04569
```

3. Data visualization for solar generator

```
cbso_sc <- cbso_z[,-2]
#cbso_location <- cbso_z[,c(1:2)]
lcbso_sc <- melt(cbso_sc, id.vars='mytime', variable.name = "variable")
#lcbso_sc <- merge(cbso_location,lcbso_sc,by="mytime")
lcbso_sc$month <- factor(month(lcbso_sc$mytime))
lcbso_sc$day <- factor(day(lcbso_sc$mytime))
lcbso_sc$hour <- factor(hour(lcbso_sc$mytime))

DF_hour <- aggregate(value ~ hour*variable, data = lcbso_sc, FUN = mean)
DF_month <- aggregate(value ~ month*variable, data = lcbso_sc, FUN = mean)
day <- subset(DF_hour,DF_hour$hour %in% c(6:18))
night <- subset(DF_hour,DF_hour$hour %in% c(0:5,19:23))
```

3.1 all solar data for different time range

```
mytheme <- theme_grey(base_family="STKaiti")
ggplot(data = DF_hour, mapping = aes(x = hour, y = value, color = variable, group = variabl
e)) + geom_line()  + mytheme + theme(axis.text.x=element_text(angle=45, vjust=0.5)) + labs(x
 = "Time(hour)") + labs(y = "ws,swdown and pac") + ggtitle("solar generator for every hour (b
y mean)")
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

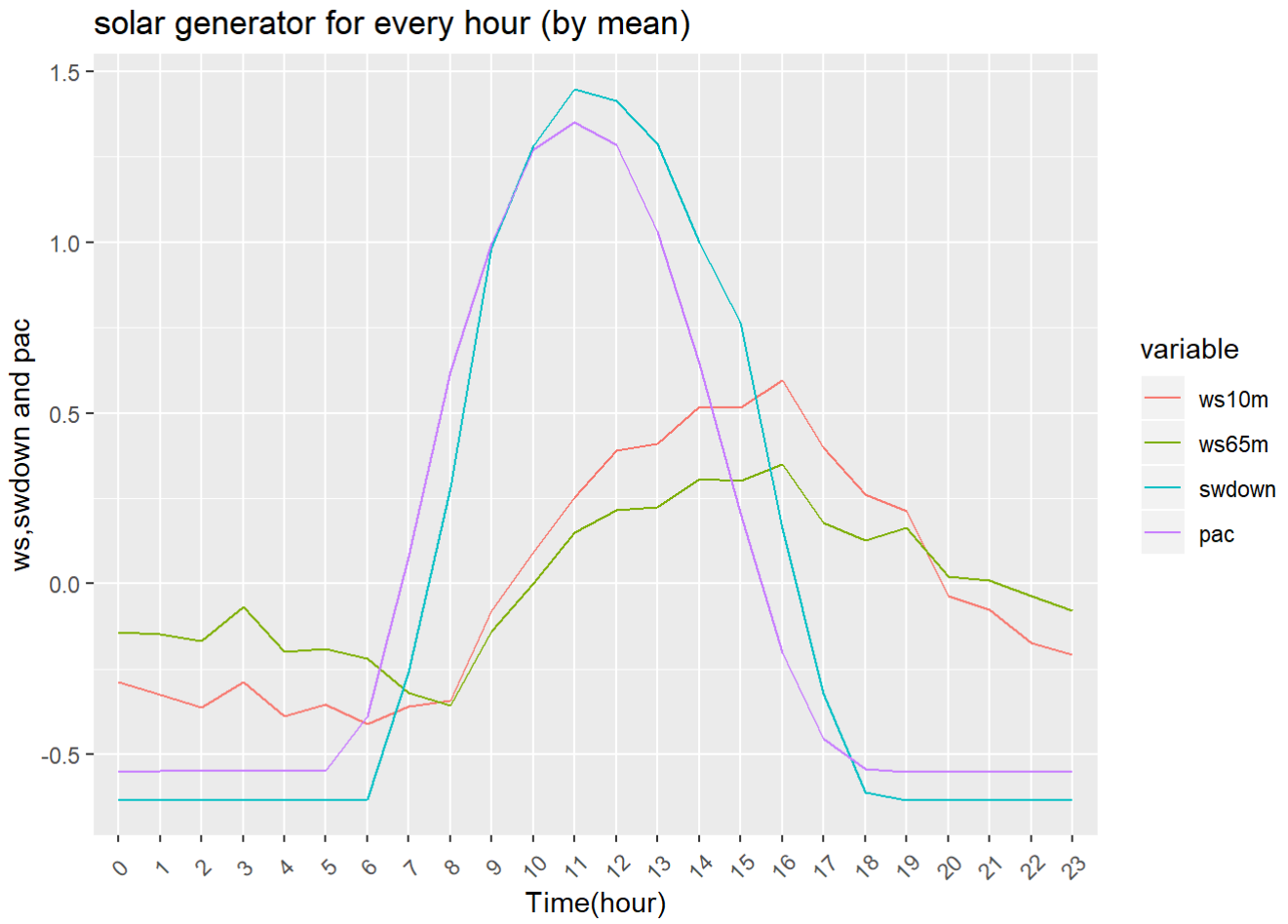
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
```

```
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

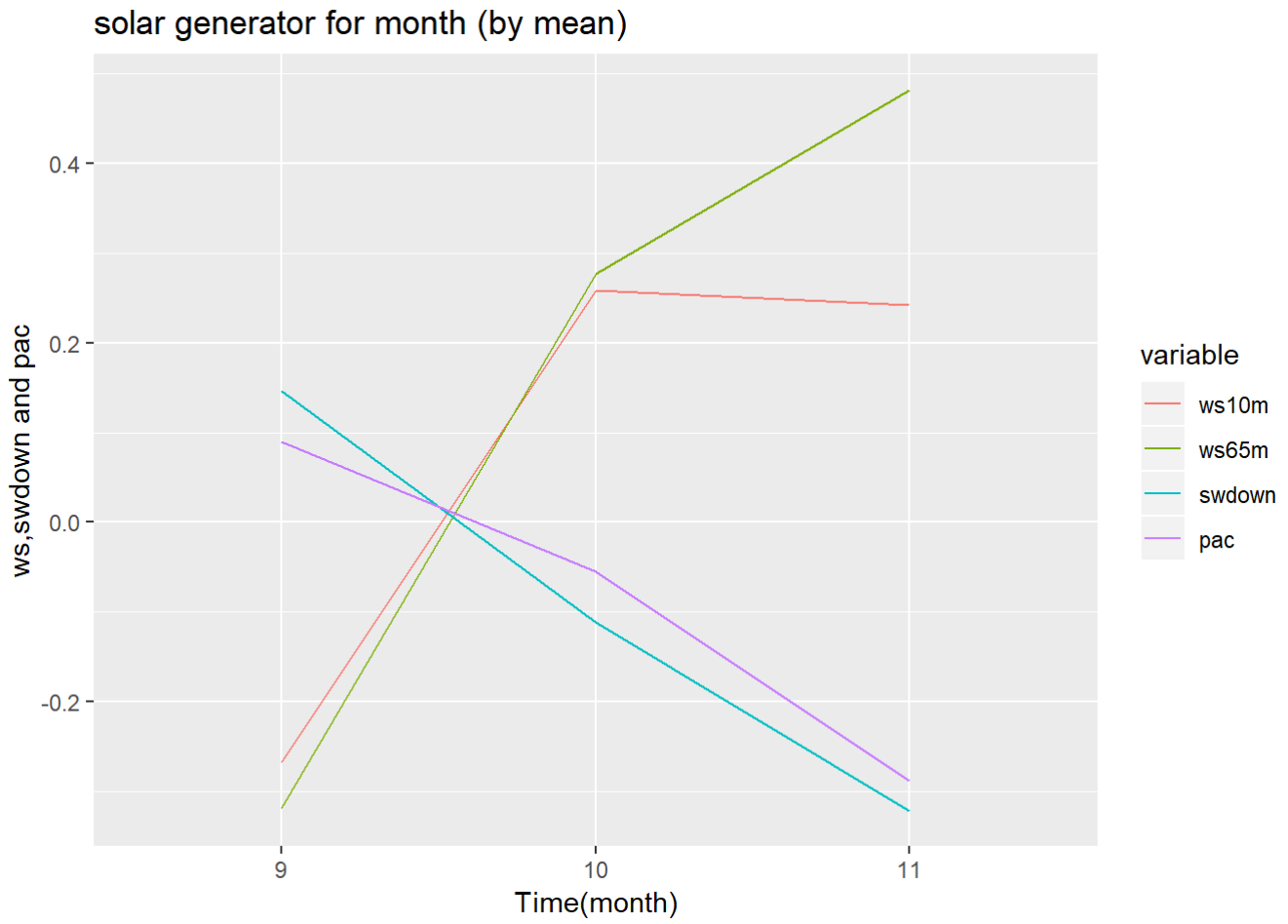



```
mytheme <- theme_grey(base_family="STKaiti")
ggplot(data = DF_month, mapping = aes(x = month, y = value, color = variable, group = variabl
e)) + geom_line()  + mytheme +  labs(x = "Time(month)") + labs(y = "ws,swdown and pac")+ ggti
tle("solar generator for month (by mean)")
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

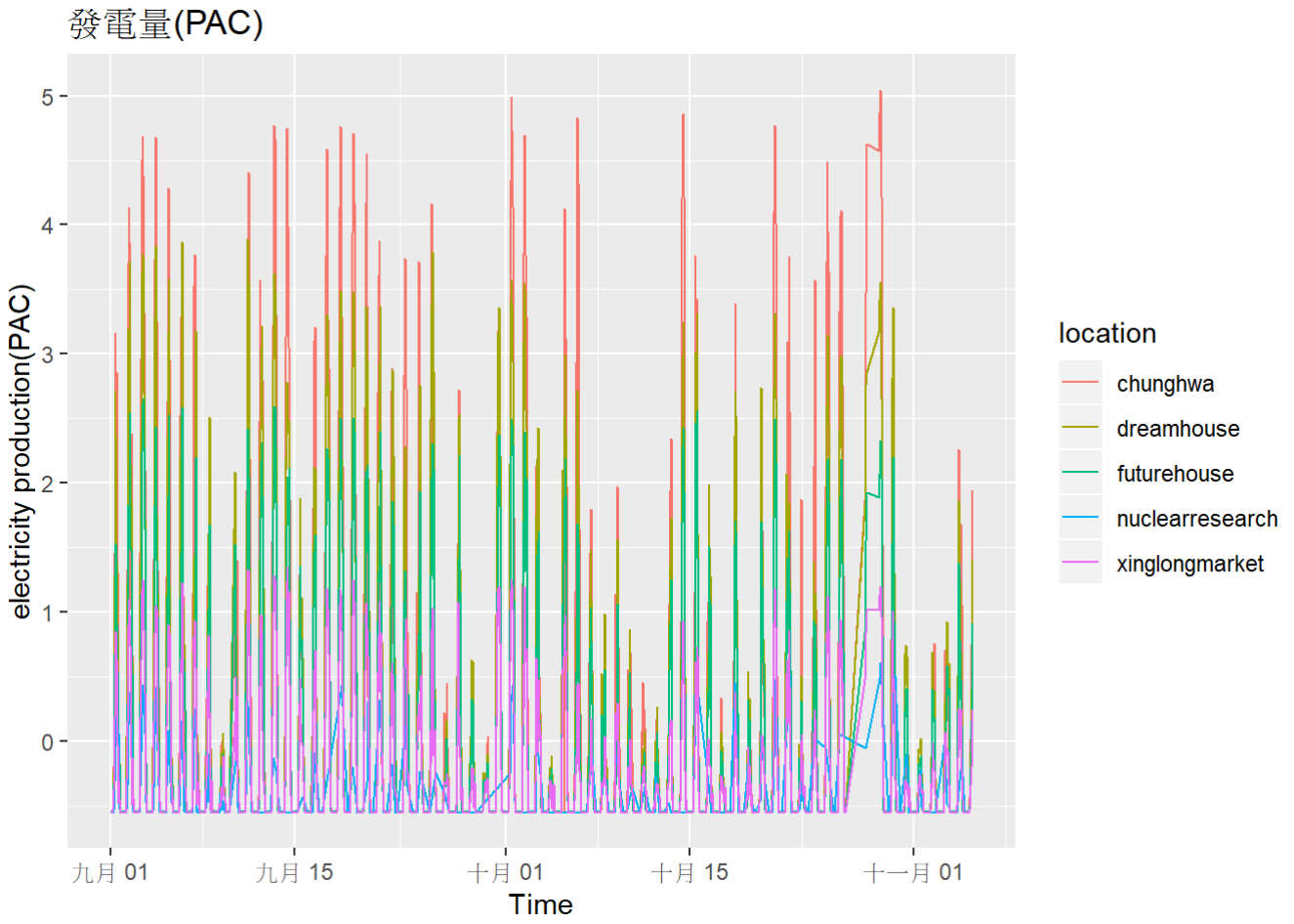
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```



3.2 solar data visualization for different location

```
pac_graph <- ggplot(cbso_z, aes(x=mytime, y=pac, colour = location, group = location)) + geom
_line() + ggtitle("發電量(PAC)")+  labs(x = "Time")+ labs(y = "electricity production(PAC)")
pac_graph
```



different location of the generator

```
chunghwa <- filter(cbso_z, cbso_z$location=="chunghwa")
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```

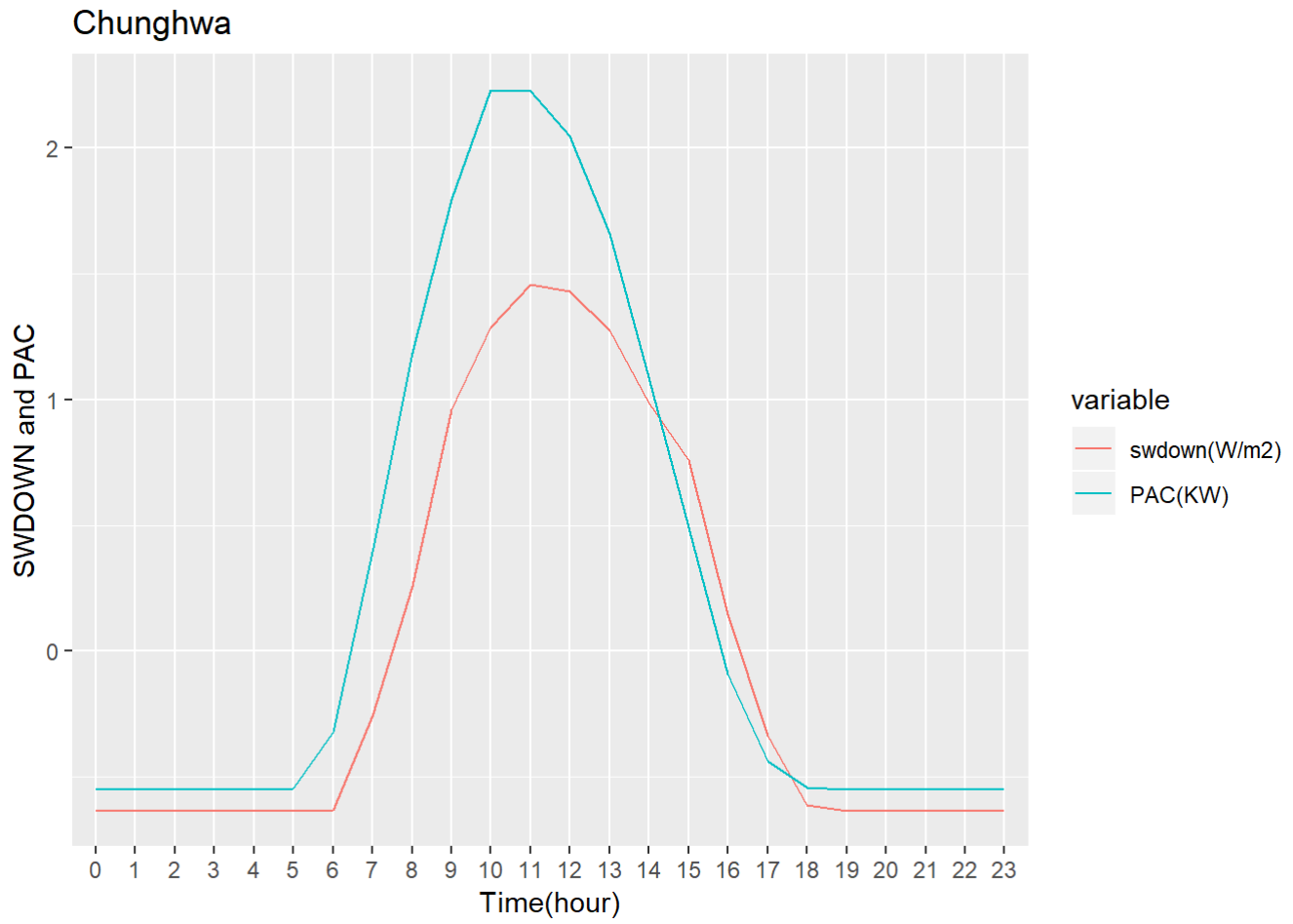
```
chunghwa_graph <- chunghwa[,c(1,5,6)]
lchunghwa <- melt(chunghwa_graph,id.vars = "mytime",measure.vars = colnames(chunghwa_graph[-1
]))
lchunghwa$hour <- factor(hour(lchunghwa$mytime))
DF_chunghwa <- aggregate(value ~ hour*variable, data = lchunghwa, FUN = mean)
mytheme <- theme_grey(base_family="STKaiti")
graph_chunghwa <- ggplot(data = DF_chunghwa, mapping = aes(x = hour, y = value, color = varia
ble, group = variable)) + geom_line()  + mytheme  + labs(x = "Time(hour)") + labs(y = "SWDOWN
and PAC")+ ggtitle("Chunghwa") + scale_color_discrete(labels=c("swdown(W/m2)","PAC(KW)"))
graph_chunghwa
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

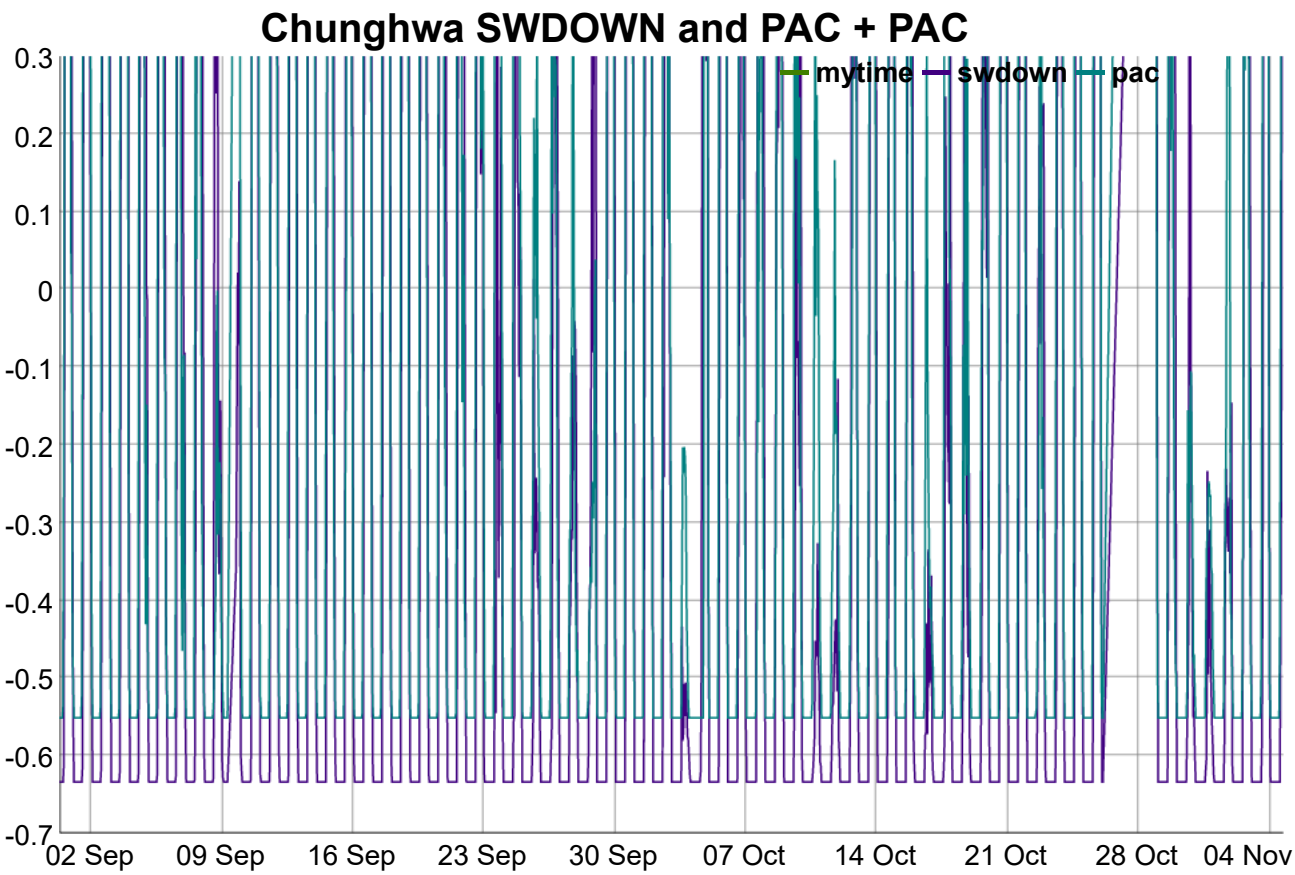
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```




```
chunghwa_xts <-  xts(chunghwa_graph, order.by = chunghwa_graph$mytime)

dygraph(chunghwa_xts,main = " Chunghwa SWDOWN and PAC + PAC") %>%
  dyAxis("y",valueRange = c(-0.7:5)) %>%
  dyRangeSelector()
```

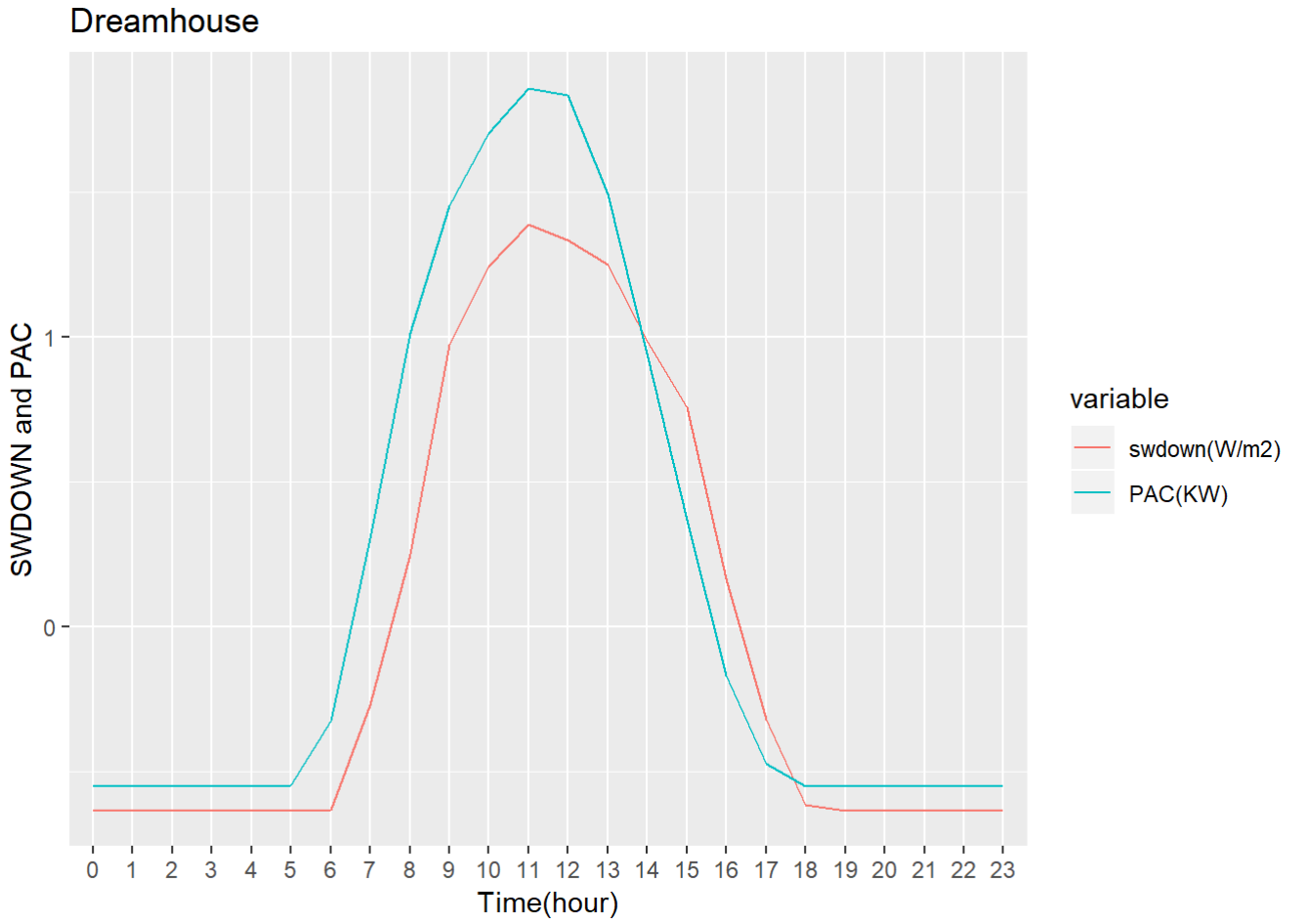


```
dreamhouse <- filter(cbso_z, cbso_z$location=="dreamhouse")
dreamhouse_graph <- dreamhouse[,c(1,5,6)]
ldreamhouse <- melt(dreamhouse_graph,id.vars = "mytime",measure.vars = colnames(chunghwa_grap
h[-1]))
ldreamhouse$hour <- factor(hour(ldreamhouse$mytime))
DF_dreamhouse <- aggregate(value ~ hour*variable, data = ldreamhouse, FUN = mean)
graph_dreamhouse <- ggplot(data = DF_dreamhouse, mapping = aes(x = hour, y = value, color = v
ariable, group = variable)) + geom_line()  + mytheme+ labs(x = "Time(hour)") + labs(y = "SWDO
WN and PAC")+ ggtitle("Dreamhouse")+ scale_color_discrete(labels=c("swdown(W/m2)","PAC(KW)"))
graph_dreamhouse
```

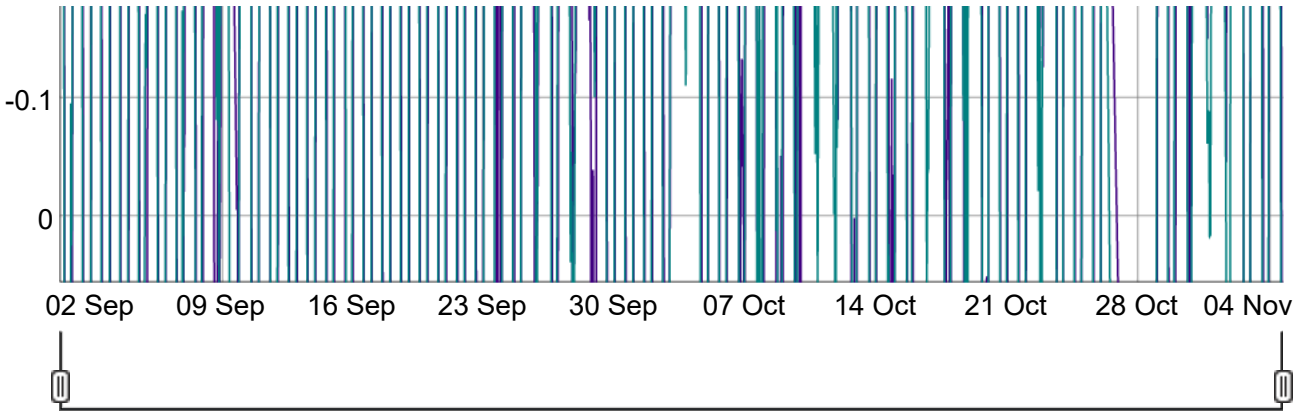
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```




```
dreamhouse_xts <-  xts(dreamhouse_graph, order.by = dreamhouse_graph$mytime)

dygraph(dreamhouse_xts, main = " Dreamhouse SWDOWN and PAC") %>%
  dyRangeSelector()
```

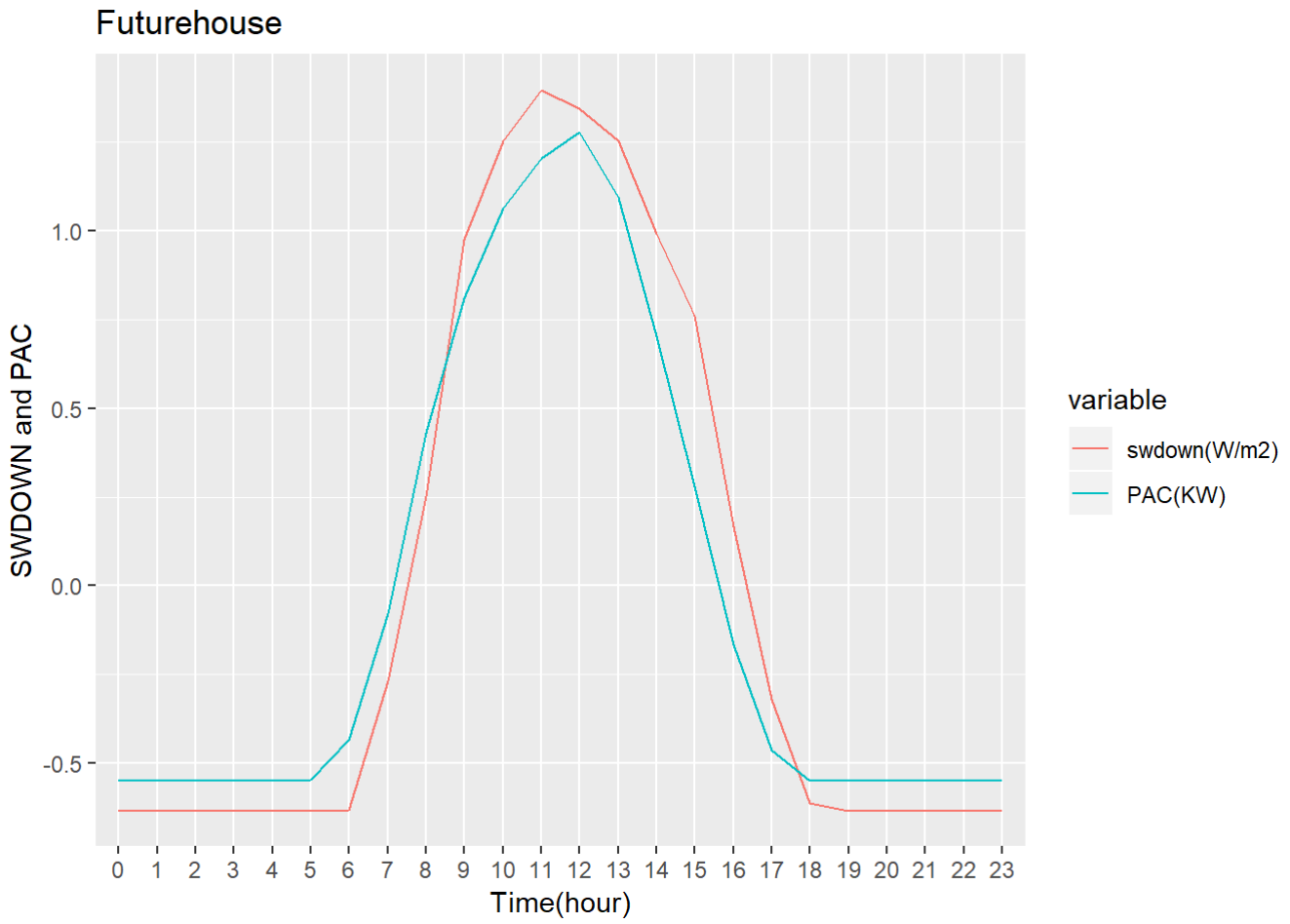



```
futurehouse <- filter(cbso_z, cbso_z$location=="futurehouse")
futurehouse_graph <- futurehouse[,c(1,5,6)]
lfuturehouse <- melt(futurehouse_graph,id.vars = "mytime",measure.vars = colnames(futurehouse
_graph[-1]))
lfuturehouse$hour <- factor(hour(lfuturehouse$mytime))
DF_futurehouse <- aggregate(value ~ hour*variable, data = lfuturehouse, FUN = mean)
graph_futurehouse <- ggplot(data = DF_futurehouse, mapping = aes(x = hour, y = value, color =
variable, group = variable)) + geom_line()  + mytheme + labs(x = "Time(hour)") + labs(y = "SW
DOWN and PAC")+ ggtitle("Futurehouse")+ scale_color_discrete(labels=c("swdown(W/m2)","PAC(K
W)"))
graph_futurehouse
```

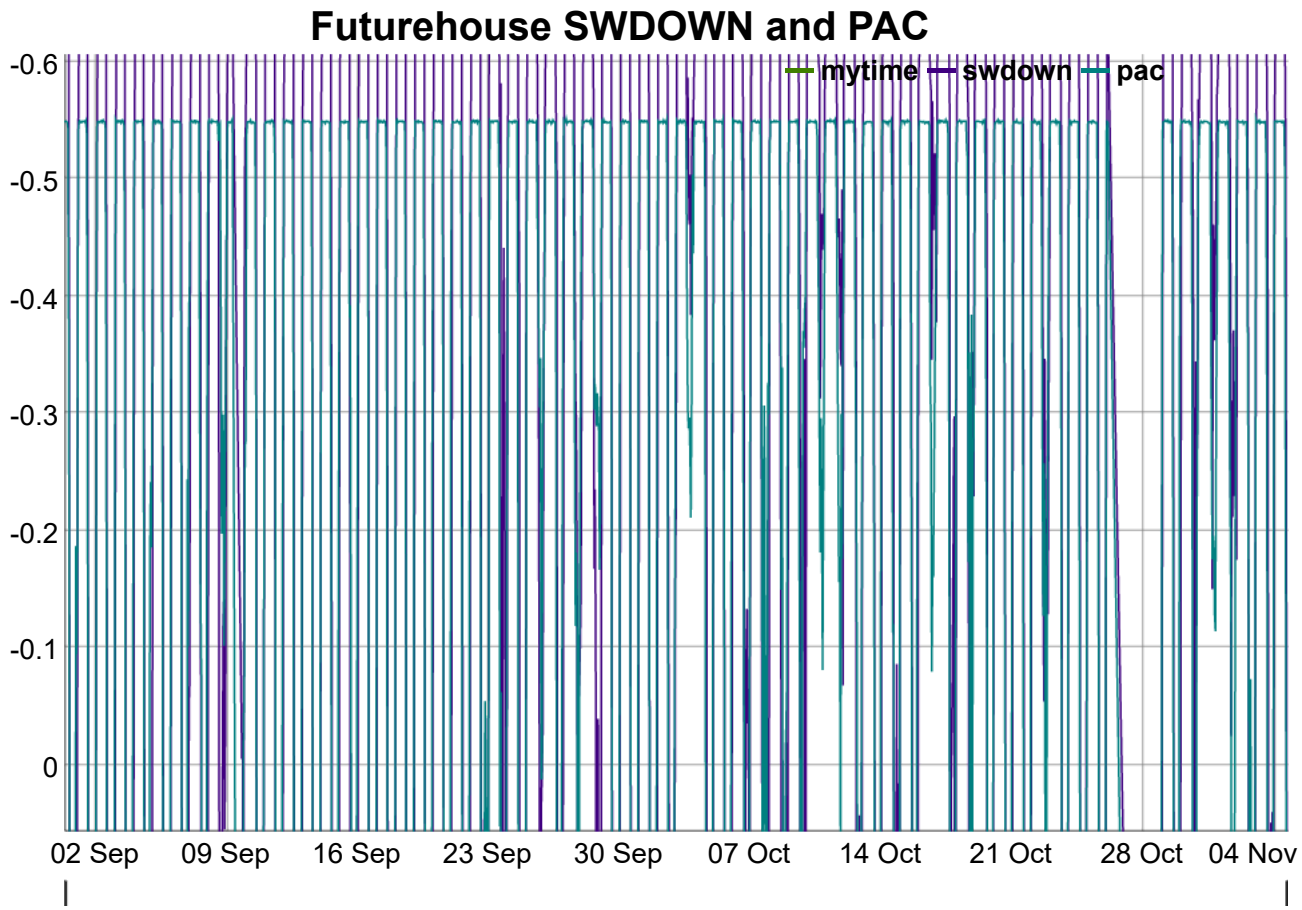
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```




```
futurehouse_xts <-  xts(futurehouse_graph, order.by = futurehouse_graph$mytime)

dygraph(futurehouse_xts, main = "Futurehouse SWDOWN and PAC") %>%
  dyRangeSelector()
```

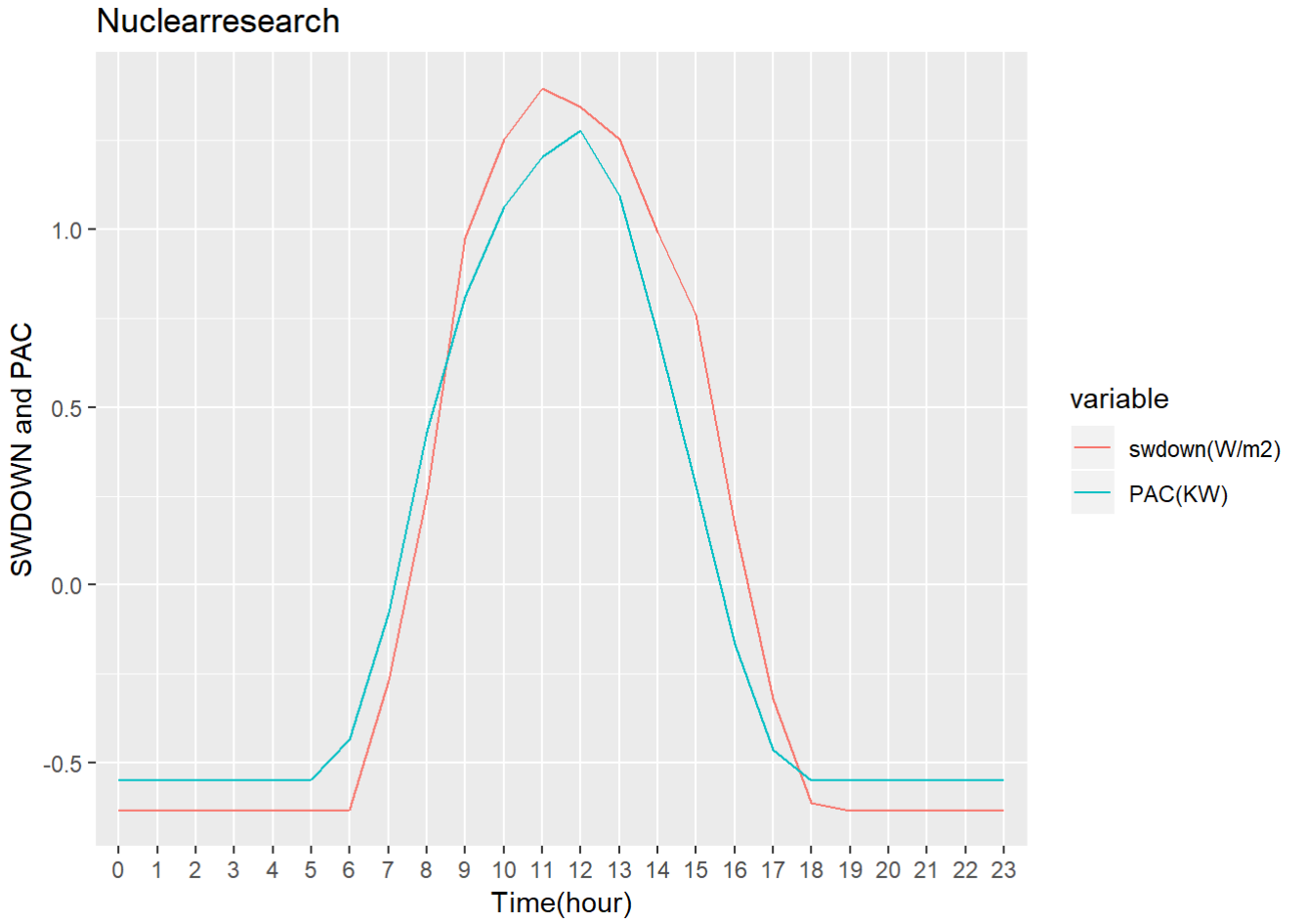


```
nuclearresearch <- filter(cbso_z, cbso_z$location=="nuclearresearch")
nuclearresearch_graph <- futurehouse[,c(1,5,6)]
lnuclearresearch <- melt(nuclearresearch_graph,id.vars = "mytime",measure.vars = colnames(nuc
learresearch_graph[-1]))
lnuclearresearch$hour <- factor(hour(lnuclearresearch$mytime))
DF_nuclearresearch <- aggregate(value ~ hour*variable, data = lnuclearresearch, FUN = mean)
graph_nuclearresearch <- ggplot(data = DF_nuclearresearch, mapping = aes(x = hour, y = value,
color = variable, group = variable)) + geom_line()  + mytheme  + labs(x = "Time(hour)") + lab
s(y = "SWDOWN and PAC")+ ggtitle("Nuclearresearch")+ scale_color_discrete(labels=c("swdown(W/
m2)","PAC(KW)"))
graph_nuclearresearch
```

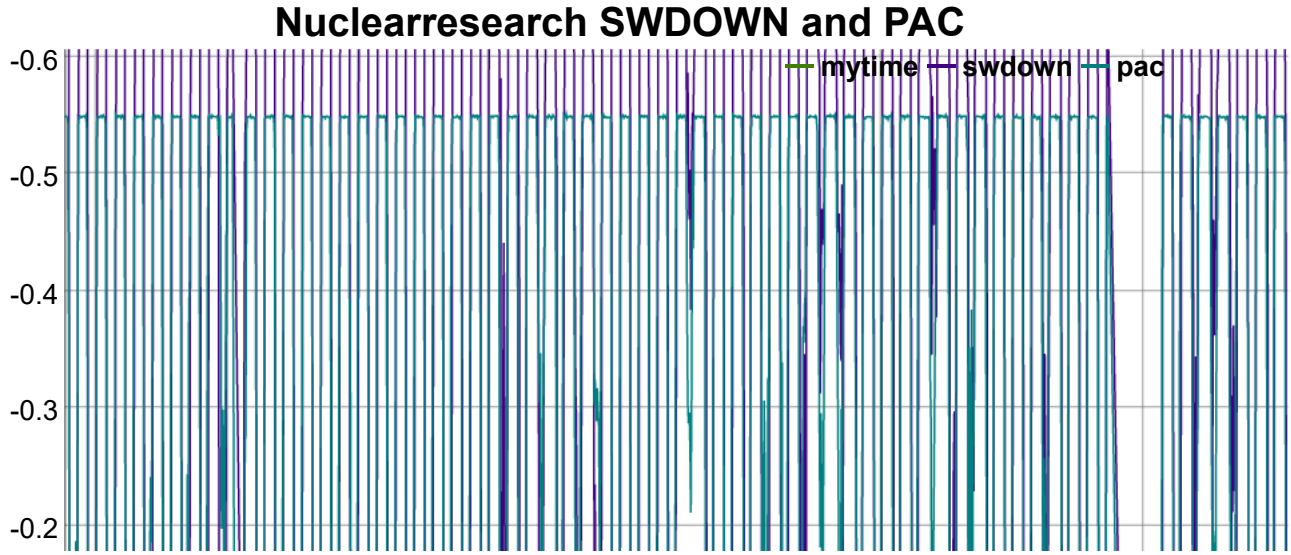
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

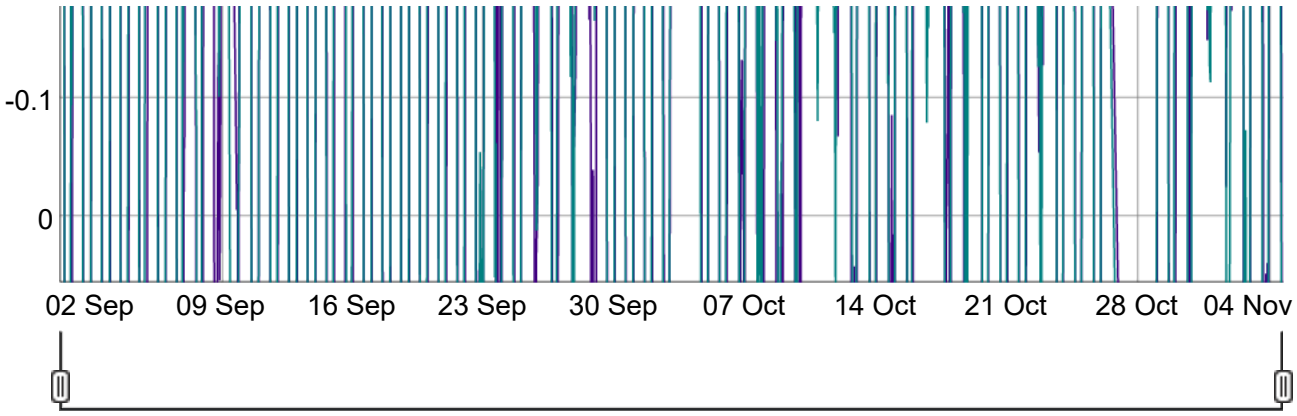
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```




```
nuclearresearch_xts <-  xts(nuclearresearch_graph, order.by = nuclearresearch_graph$mytime)

dygraph(nuclearresearch_xts, main = "Nuclearresearch SWDOWN and PAC") %>%
  dyRangeSelector()
```



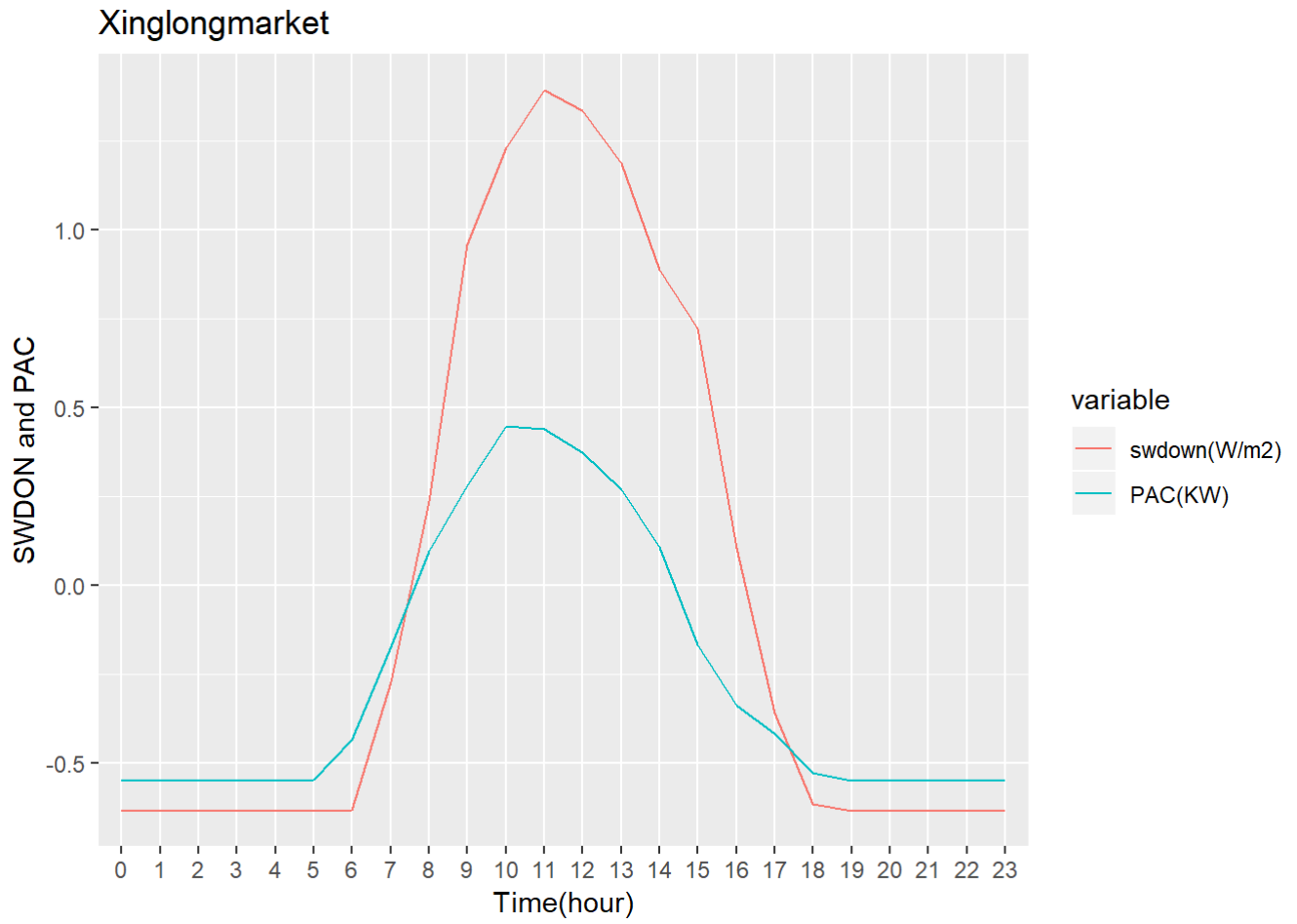



```
xinglongmarket <- filter(cbso_z, cbso_z$location=="xinglongmarket")
xinglongmarket_graph <-  xinglongmarket[,c(1,5,6)]
lxinglongmarket <- melt(xinglongmarket_graph,id.vars = "mytime",measure.vars = colnames(xingl
ongmarket_graph[-1]))
lxinglongmarket$hour <- factor(hour(lxinglongmarket$mytime))
DF_xinglongmarket <- aggregate(value ~ hour*variable, data = lxinglongmarket, FUN = mean)
graph_xinglongmarket <- ggplot(data = DF_xinglongmarket, mapping = aes(x = hour, y = value, c
olor = variable, group = variable)) + geom_line()  + mytheme  + labs(x = "Time(hour)") + labs
(y = "SWDON and PAC")+ ggtitle("Xinglongmarket")+ scale_color_discrete(labels=c("swdown(W/m
2)","PAC(KW)"))
graph_xinglongmarket
```

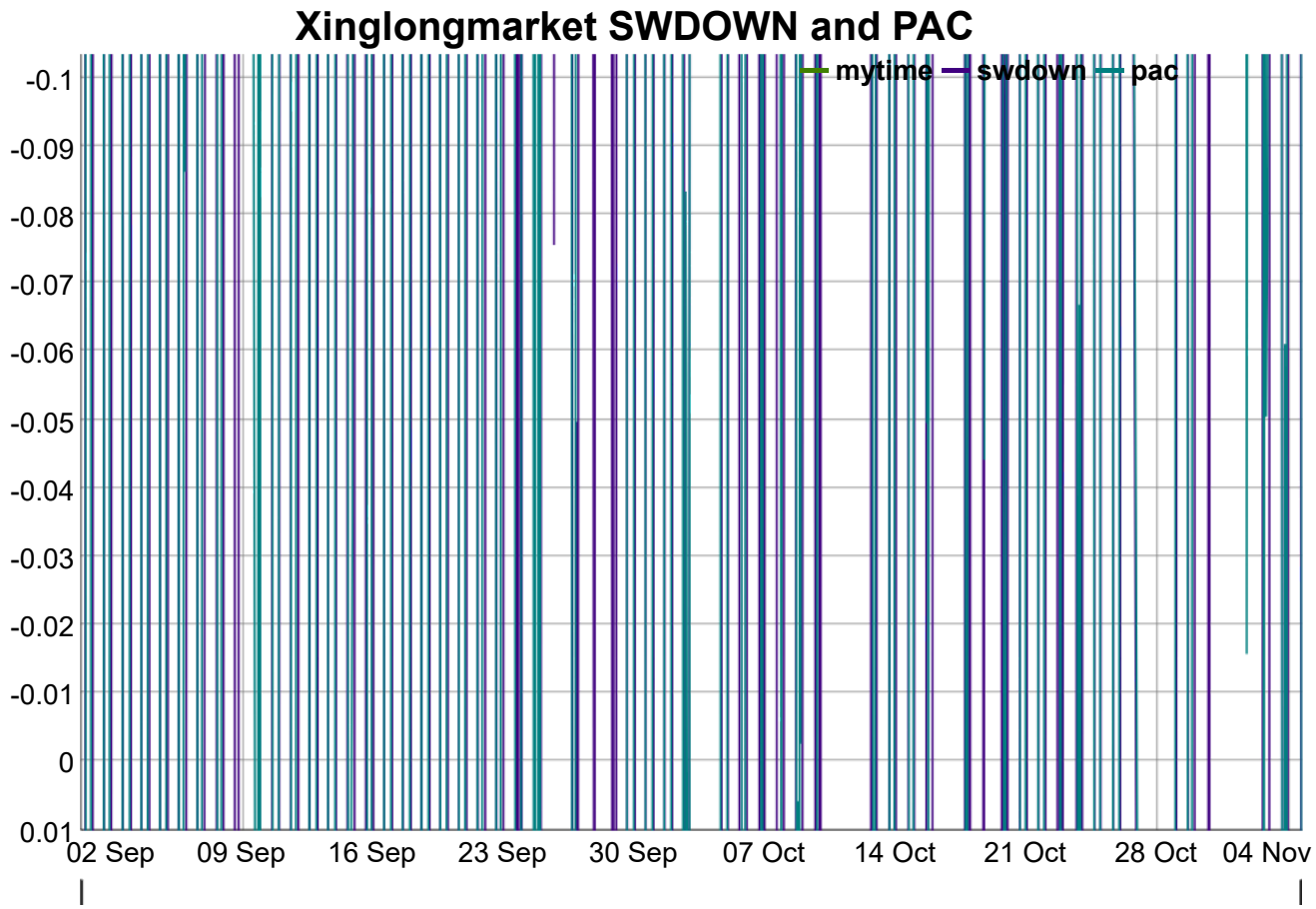
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x
## $y, : font family not found in Windows font database
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## font family not found in Windows font database
```




```
xinglongmarket_xts <-  xts(xinglongmarket_graph, order.by = xinglongmarket_graph$mytime)

dygraph(xinglongmarket_xts, main = "Xinglongmarket SWDOWN and PAC") %>%
  dyRangeSelector()
```



3.3 correlation between PAC and SWDOWN (ex. Chunghwa)

```
chunghwa_cor <- chunghwa[,-c(1:2)]
head(nearZeroVar(chunghwa_cor,saveMetrics = TRUE))
```

```
##         freqRatio percentUnique zeroVar   nzv
## ws10m    1.666667      84.32288   FALSE FALSE
## ws65m    1.333333      88.59239   FALSE FALSE
## swdown 772.000000      48.56571   FALSE FALSE
## pac    353.500000      52.70180   FALSE FALSE
```

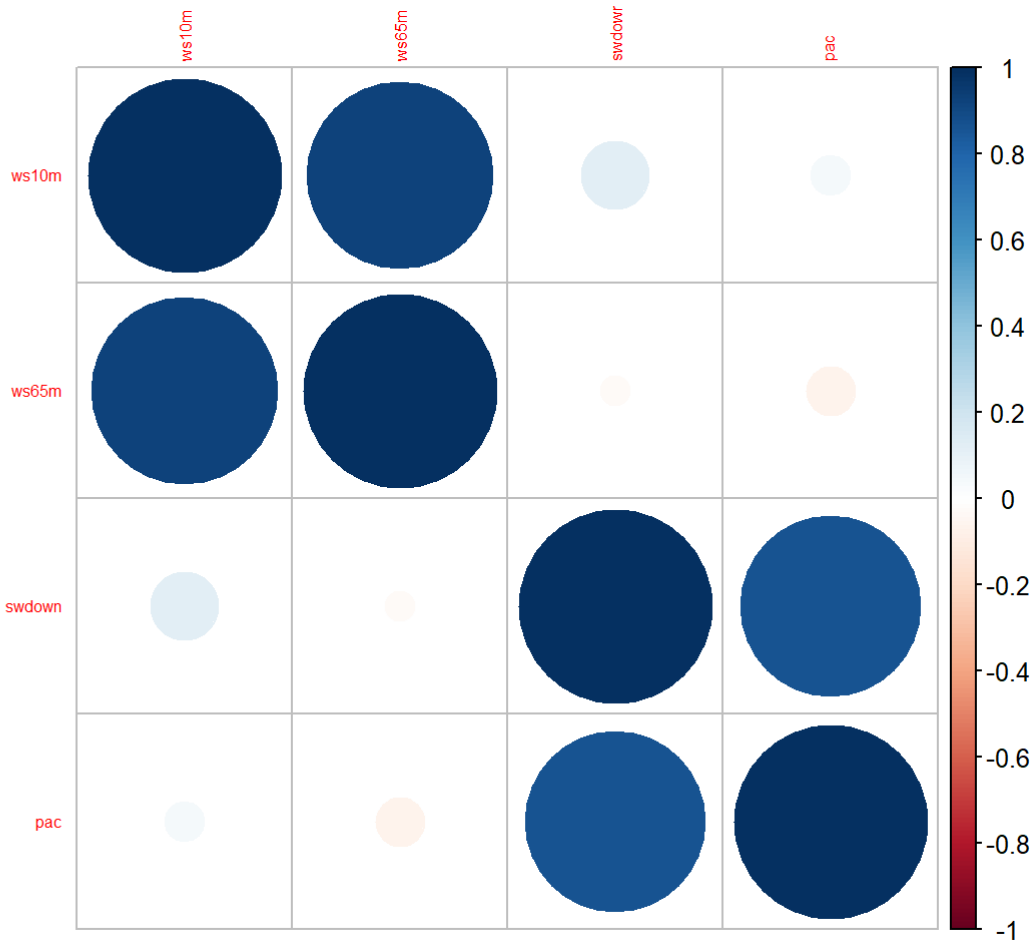
```
names(chunghwa_cor)[nearZeroVar(chunghwa_cor,saveMetrics = TRUE)$zeroVar]
```

```
## character(0)
```

```
correlations <- cor(chunghwa_cor)
dim(correlations)
```

```
## [1] 4 4
```

```
corrplot(correlations,order = "hclust",tl.cex = 0.5)
```



4. Data preparation for wind driven generator

```
formosa <- read.csv("./Data_1105/17.csv")
formosa$location <- "formosa"
chengloong <- read.csv("./Data_1105/47.csv")
chengloong$location <- "chengloong"
cbwd <- rbind(formosa,chengloong)
rm(formosa,chengloong)

names(cbwd)
```

```
##  [1] "SESSIONS"      "RANK"          "MYTIME"        "MYHOUR"
##  [5] "INITIAL_DATE"  "FCST"          "STNNO"         "FCST_DATETIME"
##  [9] "WS10M"         "WS65M"         "SWDOWN"        "USER_ID"
## [13] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [17] "RAIN"          "PAC"           "VAC"           "IAC"
## [21] "EAC"           "location"
```

```
names(cbwd)[c(1:2,4:8,12:17)]
```

```
##  [1] "SESSIONS"      "RANK"          "MYHOUR"        "INITIAL_DATE"
##  [5] "FCST"          "STNNO"         "FCST_DATETIME" "USER_ID"
##  [9] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [13] "RAIN"
```

```
cbwd <- cbwd[,-c(1:2,4:8,12:17)] #刪掉多餘欄位
summary(cbwd)
```

```
##            MYTIME         WS10M             WS65M
##  2018/09/01 00:   2   Min.   :   0.100   Min.   :-999.000
##  2018/09/01 01:   2   1st Qu.:   3.224   1st Qu.:   4.154
##  2018/09/01 02:   2   Median :   5.812   Median :   7.521
##  2018/09/01 03:   2   Mean   :  10.373   Mean   :   5.402
##  2018/09/01 04:   2   3rd Qu.:   9.250   3rd Qu.:  11.321
##  2018/09/01 05:   2   Max.   :1412.799   Max.   :  22.980
##  (Other)      :2982
##      SWDOWN            PAC               VAC             IAC
##  Min.   :-999.0   Min.   : -19048   Min.   :11097   Min.   :  0.21
##  1st Qu.:   0.0   1st Qu.: -10296   1st Qu.:11243   1st Qu.:  0.87
##  Median :   0.0   Median : 316522   Median :11412   Median : 16.11
##  Mean   : 235.2   Mean   : 671181   Mean   :11389   Mean   : 35.23
##  3rd Qu.: 501.4   3rd Qu.:1281052   3rd Qu.:11529   3rd Qu.: 65.36
##  Max.   : 980.4   Max.   :2605056   Max.   :11663   Max.   :138.45
##
##       EAC            location
##  Min.   :3374057   Length:2994
##  1st Qu.:3608466   Class :character
##  Median :4101334   Mode  :character
##  Mean   :4975430
##  3rd Qu.:6265252
##  Max.   :7189438
##
```

```
date <- substr(cbwd$MYTIME, 1,10)
time <- paste(substr(cbwd$MYTIME,12, 13),":00:00",sep = "")
cbwd$MYTIME <- as.POSIXct(paste(date,time, sep = " "))

cbwd$location <- factor(cbwd$location)

table(cbwd$WS65M=="-999")
```

```
##
## FALSE  TRUE
##  2986     8
```

```
table(cbwd$PAC< 0 )
```

```
##
## FALSE  TRUE
##  2075   919
```

```
cbwd <- subset(cbwd,WS65M!="-999")
cbwd <- subset(cbwd,PAC>0)
summary(cbwd)
```

```
##      MYTIME                        WS10M            WS65M
##  Min.   :2018-09-01 12:00:00   Min.   : 0.289   Min.   : 0.760
##  1st Qu.:2018-09-26 06:00:00   1st Qu.: 4.930   1st Qu.: 6.820
##  Median :2018-10-08 10:00:00   Median : 7.569   Median : 9.683
##  Mean   :2018-10-07 13:58:53   Mean   : 8.235   Mean   :10.056
##  3rd Qu.:2018-10-20 18:00:00   3rd Qu.:10.705   3rd Qu.:12.695
##  Max.   :2018-11-05 10:00:00   Max.   :20.998   Max.   :22.980
##      SWDOWN            PAC                 VAC             IAC
##  Min.   :  0.00   Min.   :    109.9   Min.   :11097   Min.   :  0.21
##  1st Qu.:  0.00   1st Qu.: 265608.5   1st Qu.:11242   1st Qu.: 13.69
##  Median : 20.05   Median : 828466.1   Median :11417   Median : 42.30
##  Mean   :262.28   Mean   : 975241.7   Mean   :11388   Mean   : 50.55
##  3rd Qu.:561.47   3rd Qu.:1624053.6   3rd Qu.:11530   3rd Qu.: 82.77
##  Max.   :980.44   Max.   :2605055.7   Max.   :11652   Max.   :138.45
##       EAC                location
##  Min.   :3374100   chengloong:1045
##  1st Qu.:3652177   formosa   :1022
##  Median :4100099
##  Mean   :5046608
##  3rd Qu.:6473596
##  Max.   :7189438
```

```
cbwd_sc <- as.data.frame(scale(cbwd[,c(2:5)]))
cbwd_scframe <- cbind(cbwd$MYTIME,cbwd$location,cbwd_sc)
colnames(cbwd_scframe) <- c("mytime","location","ws10m","ws65m","swdown","pac")
summary(cbwd_scframe)
```

```
##      mytime                          location        ws10m
##  Min.   :2018-09-01 12:00:00   chengloong:1045   Min.   :-1.9297
##  1st Qu.:2018-09-26 06:00:00   formosa   :1022   1st Qu.:-0.8026
##  Median :2018-10-08 10:00:00                     Median :-0.1617
##  Mean   :2018-10-07 13:58:53                     Mean   : 0.0000
##  3rd Qu.:2018-10-20 18:00:00                     3rd Qu.: 0.5999
##  Max.   :2018-11-05 10:00:00                     Max.   : 3.0996
##      ws65m              swdown             pac
##  Min.   :-2.20263   Min.   :-0.7929   Min.   :-1.2697
##  1st Qu.:-0.76681   1st Qu.:-0.7929   1st Qu.:-0.9240
##  Median :-0.08848   Median :-0.7323   Median :-0.1911
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.62528   3rd Qu.: 0.9045   3rd Qu.: 0.8448
##  Max.   : 3.06202   Max.   : 2.1712   Max.   : 2.1222
```

```
cbwd_sc_noloc <- cbwd_scframe[,-2]
```

5. TScluster to ensure the data

TScluster could figure out the correlation of the varibles.

很多變數的時間序列如何找出相關性的變數(上面就已經看的出來了....)

TScluster主要是看時間序列的資料中，有很多筆具時間刻度的資料，例如股票資料，在一段時間不同支股票是否有哪幾支有相似走勢或相關，但本案子已經確定swdown和pac有相關

用TScluster還是可以從一批資料中發掘PAC是與哪個發電來源有關

```
library(TSclust)
```

```
## Warning: package 'TSclust' was built under R version 3.4.4
```

```
## Loading required package: wmtsa
```

```
## Warning: package 'wmtsa' was built under R version 3.4.4
```

```
## Loading required package: pdc
```

```
## Warning: package 'pdc' was built under R version 3.4.4
```

```
## Loading required package: cluster
```

```
summary(cbso_sc)
```

```
##      mytime                        ws10m              ws65m
##  Min.   :2018-09-01 00:00:00   Min.   :-1.70944   Min.   :-1.77558
##  1st Qu.:2018-09-15 10:00:00   1st Qu.:-0.80698   1st Qu.:-0.80190
##  Median :2018-10-01 16:00:00   Median :-0.07387   Median :-0.09021
##  Mean   :2018-10-02 01:19:24   Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.:2018-10-17 22:00:00   3rd Qu.: 0.63185   3rd Qu.: 0.60872
##  Max.   :2018-11-05 10:00:00   Max.   : 5.21567   Max.   : 4.04270
##      swdown             pac
##  Min.   :-0.6335   Min.   :-0.55077
##  1st Qu.:-0.6335   1st Qu.:-0.55077
##  Median :-0.6335   Median :-0.54502
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.3228   3rd Qu.: 0.07185
##  Max.   : 3.0864   Max.   : 5.04569
```

```
cbso_mat <- xts(cbso_sc[-1], order.by = cbso_sc$mytime)
# head(cbso_mat)
cbso_mat = t(as.matrix(cbso_mat)) #TScluster一定要轉成matrix且以rows為觀測值
D_mat <- diss(cbso_mat, "COR")
summary(D_mat)
```

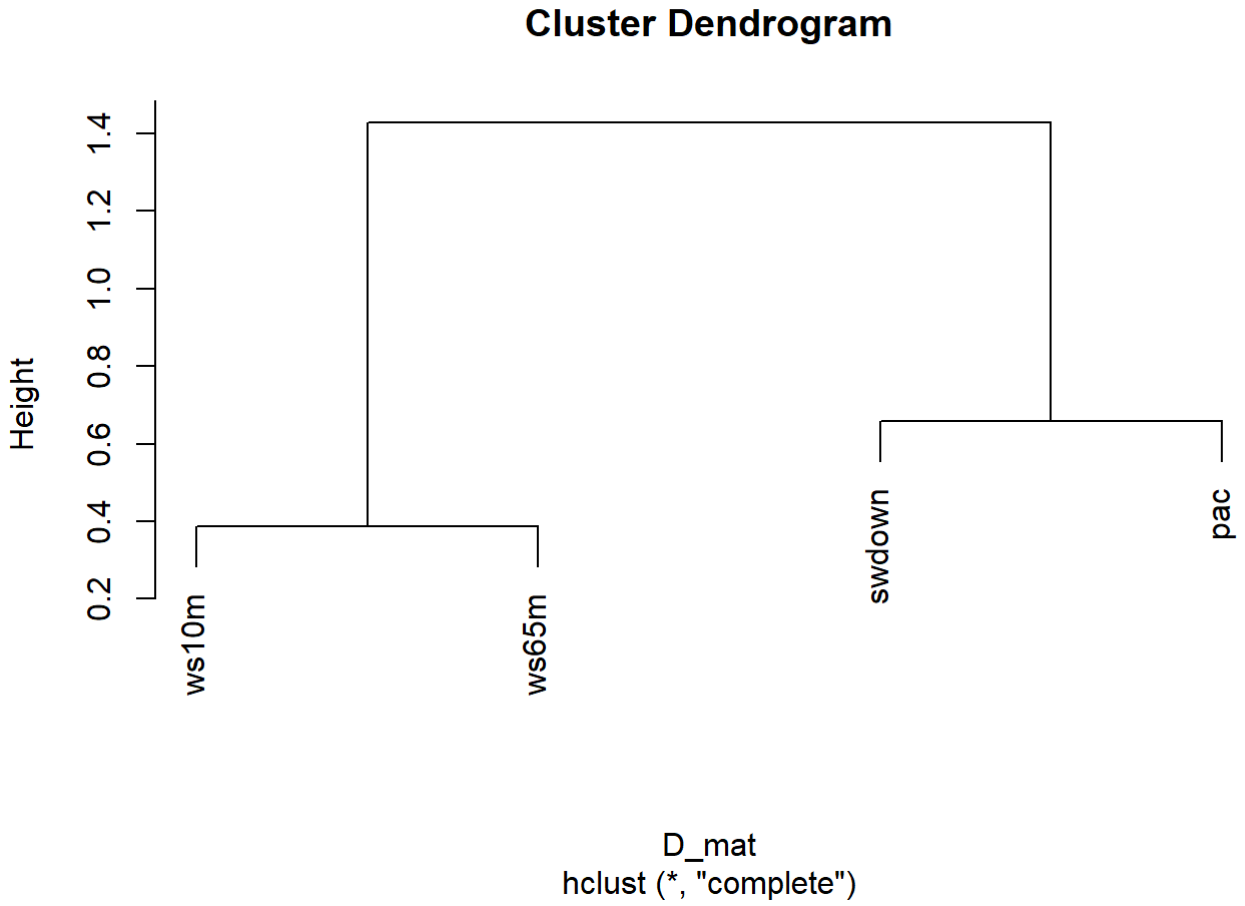
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3882  0.8271  1.3536  1.1005  1.4101  1.4271
```

```
sort(rowMeans(as.matrix(D_mat)))
```

```
##     ws10m     ws65m     swdown       pac
## 0.7738485 0.8092577 0.8531175 0.8652824
```

```
C_mat <- hclust(D_mat)
```

```
plot(C_mat)
```



According to the above graph, we know that the data is solar generator. Because the SWDOWN and PAC are much closer.

```
cbwd_mat <- xts(cbwd_sc_noloc[-1], order.by = cbwd_sc_noloc$mytime)
head(cbwd_mat)
```

```
##                         ws10m     ws65m    swdown       pac
## 2018-09-01 12:00:00 -1.189476 -1.595369 2.1503327 -1.225589
## 2018-09-01 13:00:00 -1.293905 -1.628302 2.0854043 -1.222471
## 2018-09-01 14:00:00 -1.288805 -1.598923 1.8161732 -1.225179
## 2018-09-01 15:00:00 -1.128519 -1.384262 1.3501403 -1.231423
## 2018-09-02 08:00:00 -1.629046 -1.848176 0.5008071 -1.246321
## 2018-09-02 09:00:00 -1.241205 -1.605794 1.1420525 -1.159902
```

```
cbwd_mat = t(as.matrix(cbwd_mat)) #TScluster一定要轉成matrix且以rows為觀測值
D_mat <- diss(cbwd_mat, "COR")
summary(D_mat)
```

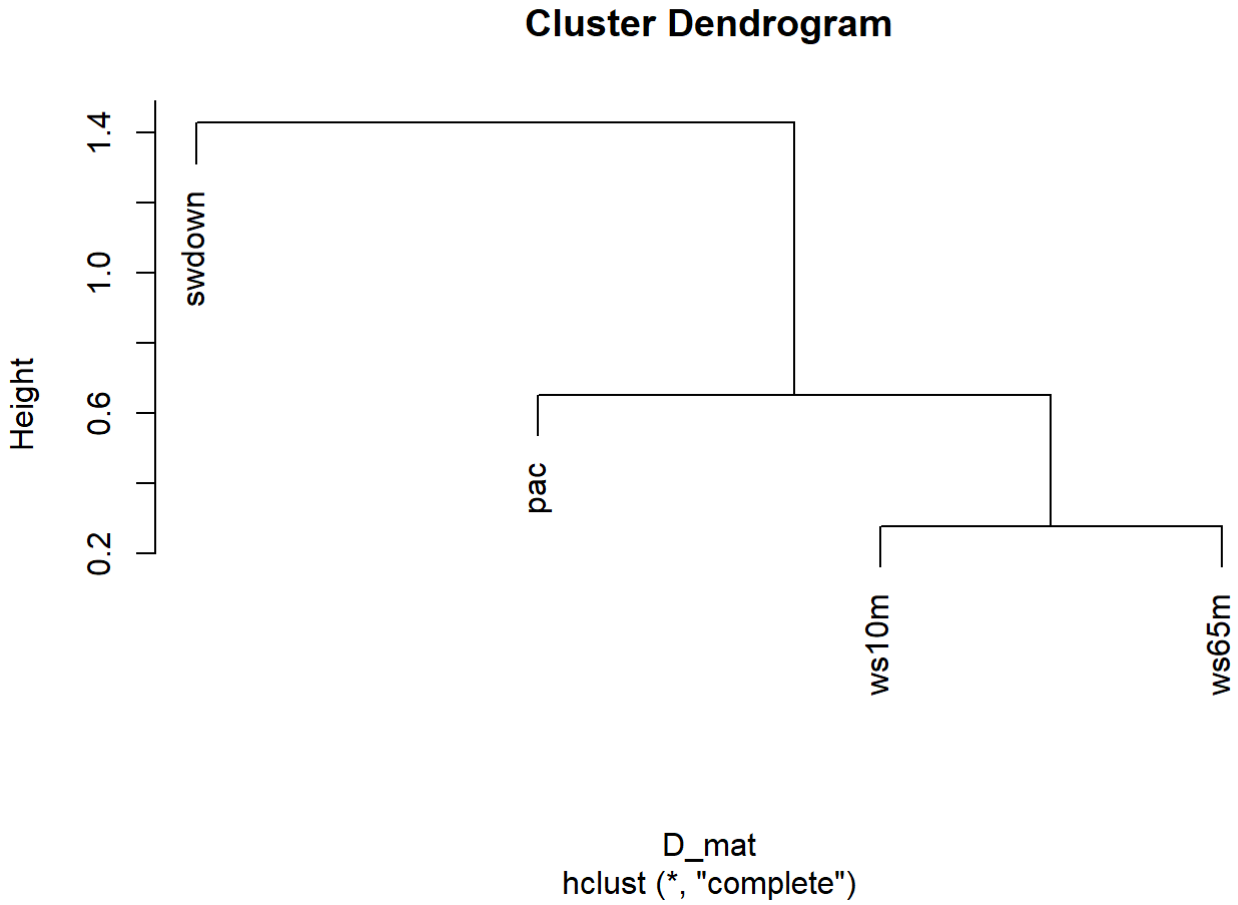
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2779  0.6278  1.0164  0.9600  1.3970  1.4269
```

```
sort(rowMeans(as.matrix(D_mat)))
```

```
##     ws10m     ws65m       pac    swdown
## 0.5776808 0.5812040 0.6683550 1.0526629
```

```
C_mat <- hclust(D_mat)
```

```
plot(C_mat)
```



The graph is wind driven generator, because the two different high of wind speed are more closer to PAC than SWDOWN.

6. Deep learning for LSTM

6.1 solar data

We use solar generator data observed from Chunghwa to build the model.

```
library(keras)
```

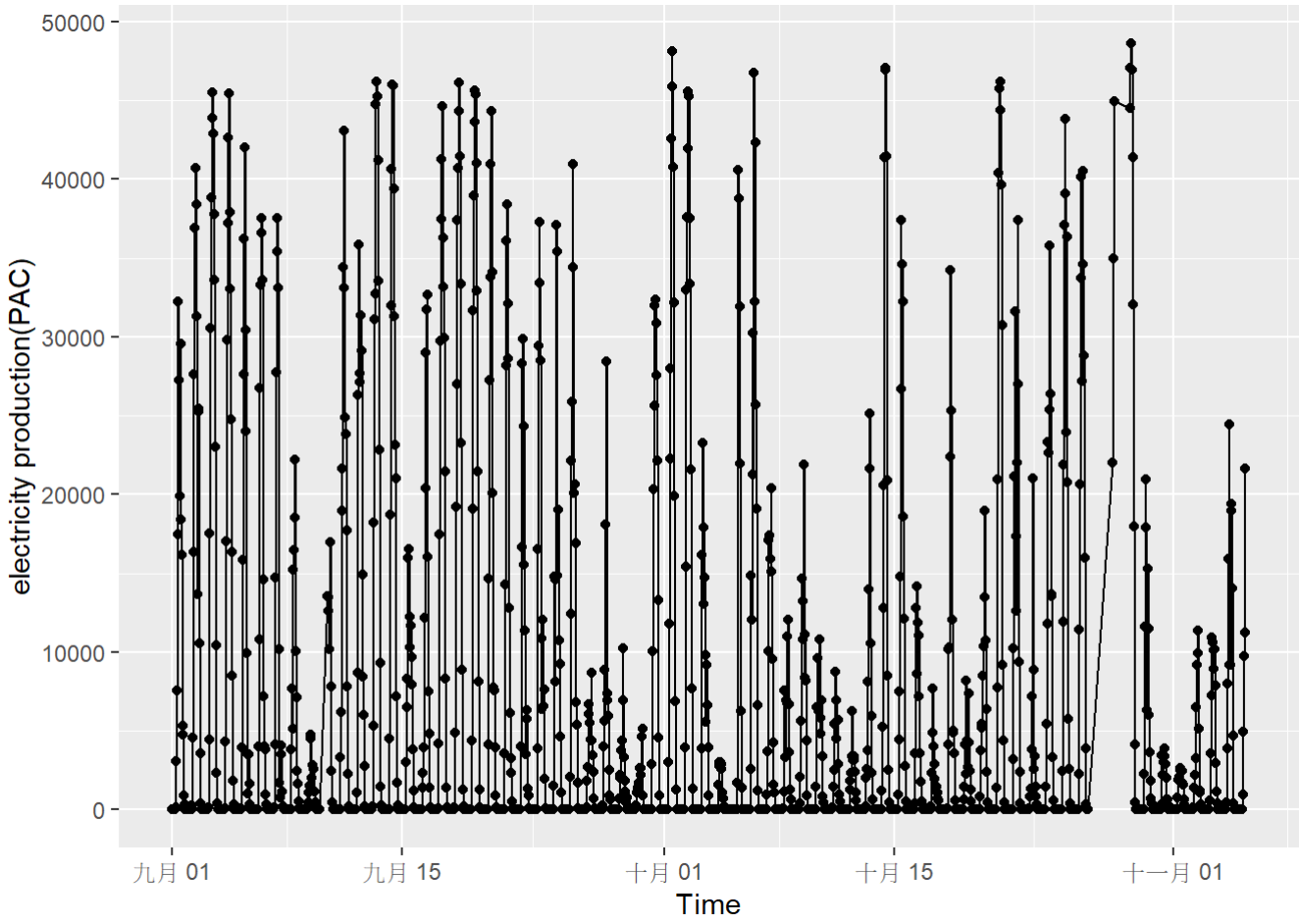
```
## Warning: package 'keras' was built under R version 3.4.4
```

```
library(dplyr)
library(ggplot2)
library(ggthemes)
```

```
## Warning: package 'ggthemes' was built under R version 3.4.4
```

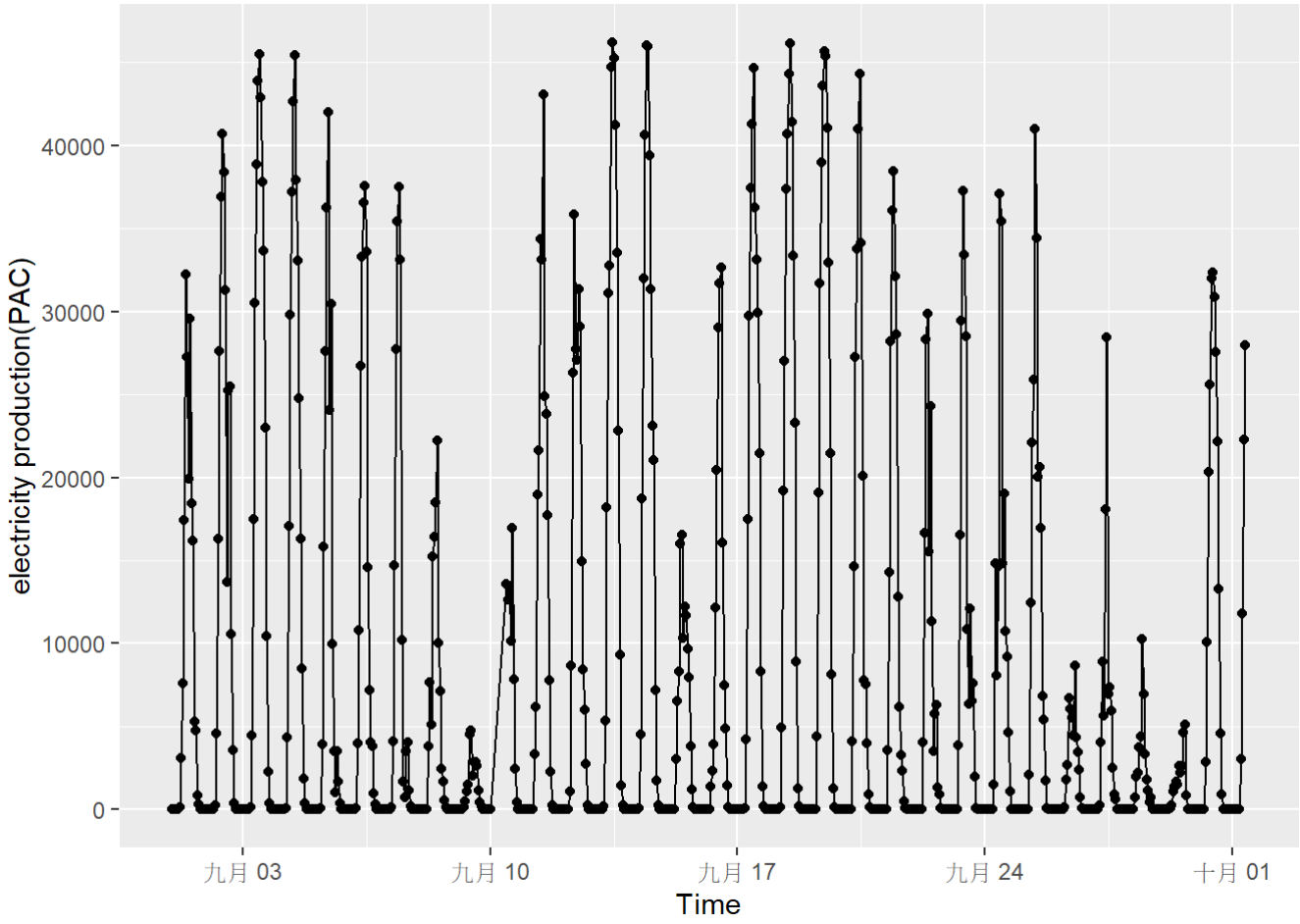
```
library(lubridate)
library(xts)
set.seed(7)

dataframe <- read.csv(
  './Data_1105/14.csv')

date <- substr(dataframe$MYTIME, 1,10)
time <- paste(substr(dataframe$MYTIME,12, 13),":00:00",sep = "")
dataframe$MYTIME <- as.POSIXct(paste(date,time, sep = " "))


dataframe <- dataframe[order(dataframe$MYTIME),]
```

```
ggplot(
  data = dataframe,
  mapping = aes(
    x = MYTIME,
    y = PAC)) +
  geom_line() +
  geom_point() +
  labs(x = "Time") +
  labs(y = "electricity production(PAC)")
```



September from Chunghwa

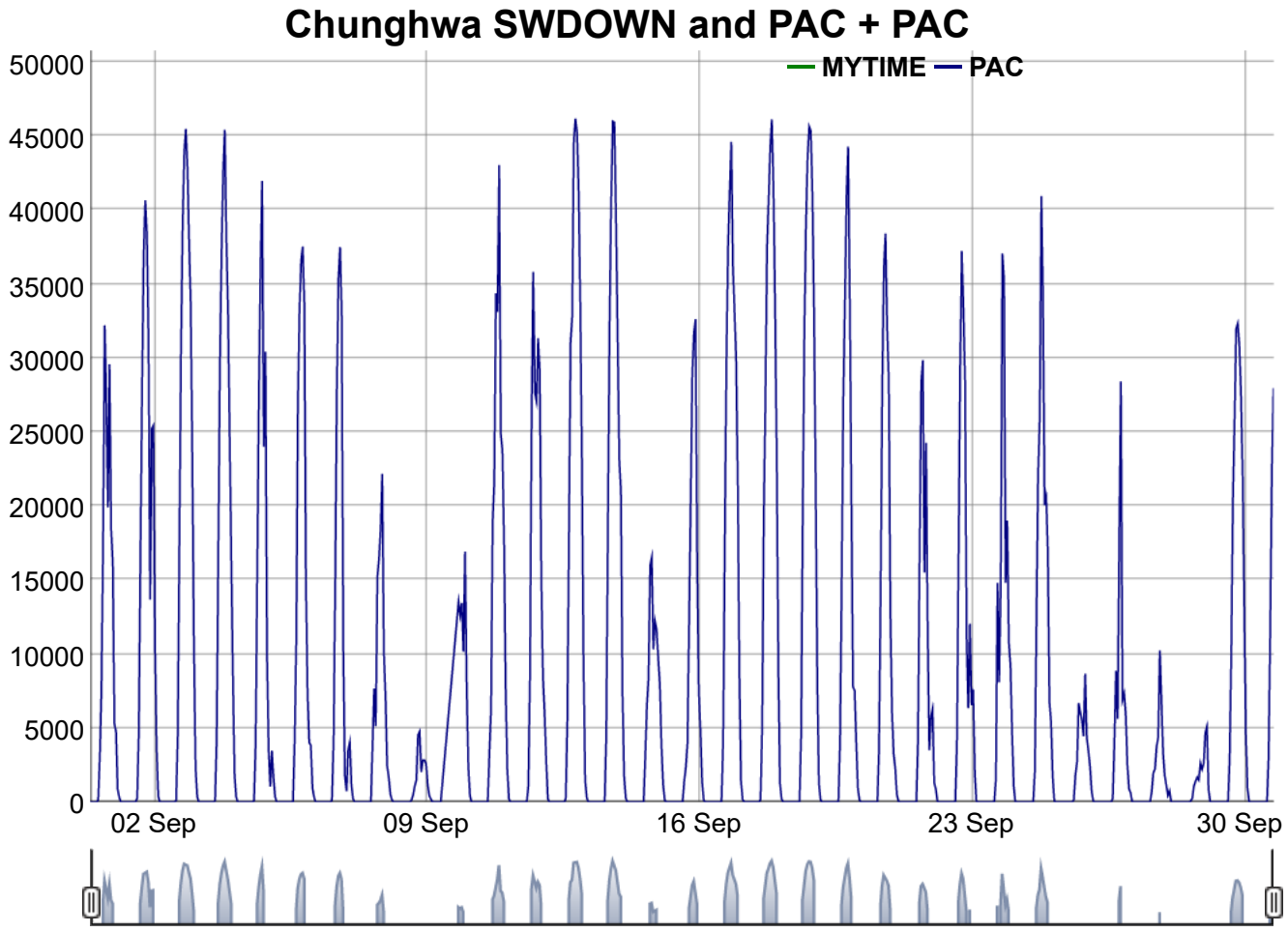
```
sept <- dataframe[1:720,]

ggplot(
  data = sept,
  mapping = aes(
    x = MYTIME,
    y = PAC)) +
  geom_line() +
  geom_point() +
  labs(x = "Time") +
  labs(y = "electricity production(PAC)")
```




```
names(sept)
```

```
##  [1] "SESSIONS"      "RANK"          "MYTIME"        "MYHOUR"
##  [5] "INITIAL_DATE"  "FCST"          "STNNO"         "FCST_DATETIME"
##  [9] "WS10M"         "WS65M"         "SWDOWN"        "USER_ID"
## [13] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [17] "RAIN"          "PAC"           "VAC"           "IAC"
## [21] "EAC"
```

```
sept <- sept[,c(3,18)]

library(dygraphs)
sept_xts <-  xts(sept, order.by = sept$MYTIME)

dygraph(sept_xts, main = " Chunghwa SWDOWN and PAC + PAC") %>%
  dyRangeSelector()
```



data normalization

```
max_value <- max(sept$PAC)
min_value <- min(sept$PAC)
spread <- max_value - min_value

dataset <- (sept$PAC - min_value) / spread #正規化資料
range(dataset)
```

```
## [1] 0 1
```

build the model

```
create_dataset <- function(dataset,
                           look_back = 1)
{
  l <- length(dataset)
  dataX <- array(dim = c(l - look_back, look_back))

  for (i in 1:ncol(dataX))
  {
    dataX[, i] <- dataset[i:(l - look_back + i - 1)]
  }

  dataY <- array(
    data = dataset[(look_back + 1):l],
    dim = c(l - look_back, 1))

  return(
    list(
      dataX = dataX,
      dataY = dataY))
} #設定x y 的資料格式

train_size <- as.integer(length(dataset) * 0.67)
test_size <- length(dataset) - train_size

train <- dataset[1:train_size]
test <- dataset[(train_size + 1):length(dataset)]

cat(length(train), length(test))  #切分訓練集與測試集
```

```
## 482 238
```

```
# 482 238

look_back <- 1           #設定t-1
trainXY <- create_dataset(train, look_back)
testXY <-  create_dataset(test, look_back)

dim_train <- dim(trainXY$dataX)
dim_test <- dim(testXY$dataX)

# reshape input to be [samples, time steps, features]
dim(trainXY$dataX) <- c(dim_train[1], 1, dim_train[2])
dim(testXY$dataX) <- c(dim_test[1], 1, dim_test[2])
```

training data

```
model <- keras_model_sequential()

model %>%
  layer_lstm(
    units = 4,
    input_shape = c(1, look_back)) %>%
  layer_dense(
    units = 1) %>%
  compile(
    loss = 'mean_squared_error',
    optimizer = 'adam') %>%
  fit(trainXY$dataX,
      trainXY$dataY,
      epochs = 30,  #調30代就差不多
      batch_size = 1,
      verbose = 2)
```

result

```
trainScore <- model %>%
  evaluate(
    trainXY$dataX,
    trainXY$dataY,
    verbose = 2)

testScore <- model %>%
  evaluate(
    testXY$dataX,
    testXY$dataY,
    verbose = 2)

# trainScore_inv = trainScore * spread + min_value

sprintf(
  'Train Score: %.4f MSE (%.4f RMSE)',
  trainScore * spread^2,
  sqrt(trainScore) * spread)
```

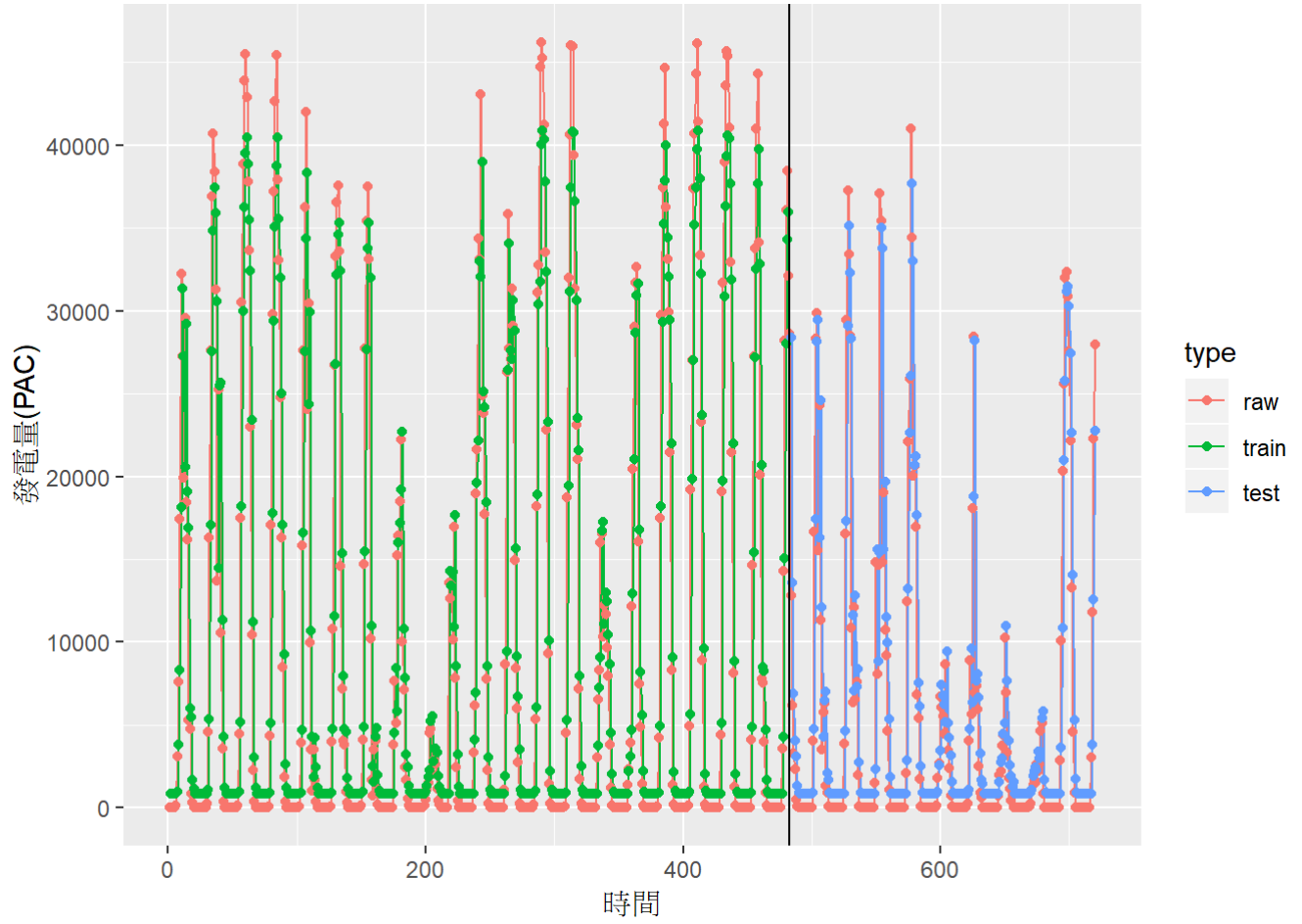
```
## [1] "Train Score: 31101279.1256 MSE (5576.8521 RMSE)"
```

```
sprintf(
  'Test Score: %.4f MSE (%.4f RMSE)',
  testScore * spread^2,
  sqrt(testScore) * spread)
```

```
## [1] "Test Score: 23540653.7070 MSE (4851.8712 RMSE)"
```

visualization of the result

```
trainPredict <- model %>%
  predict(
    trainXY$dataX,
    verbose = 2)
testPredict <- model %>%
  predict(
    testXY$dataX,
    verbose = 2)

trainPredict <- trainPredict * spread + min_value
testPredict <- testPredict * spread + min_value

df <- data.frame(
  index = 1:length(dataset),
  value = dataset * spread + min_value,
  type = 'raw') %>%
  rbind(
    data.frame(
      index = 1:length(trainPredict) + look_back,
      value = trainPredict,
      type = 'train')) %>%
  rbind(
    data.frame(
      index = 1:length(testPredict) + look_back + length(train),
      value = testPredict,
      type = 'test'))

ggplot(data = df) +
  geom_line(
    mapping = aes(
      x = index,
      y = value,
      color = type)) +
  geom_point(
    mapping = aes(
      x = index,
      y = value,
      color = type)) +
  geom_vline(
    xintercept = length(train) + 0.5) +
  labs(x = "時間") +
  labs(y = "發電量(PAC)")
```




```
mean((trainPredict - mean(trainPredict))^2)
```

```
## [1] 167062768
```

```
sqrt(mean((trainPredict - mean(trainPredict))^2))
```

```
## [1] 12925.28
```

```
trainActual = trainXY$dataY * spread + min_value
mean(trainActual)
```

```
## [1] 9608.909
```

```
sd(trainActual)
```

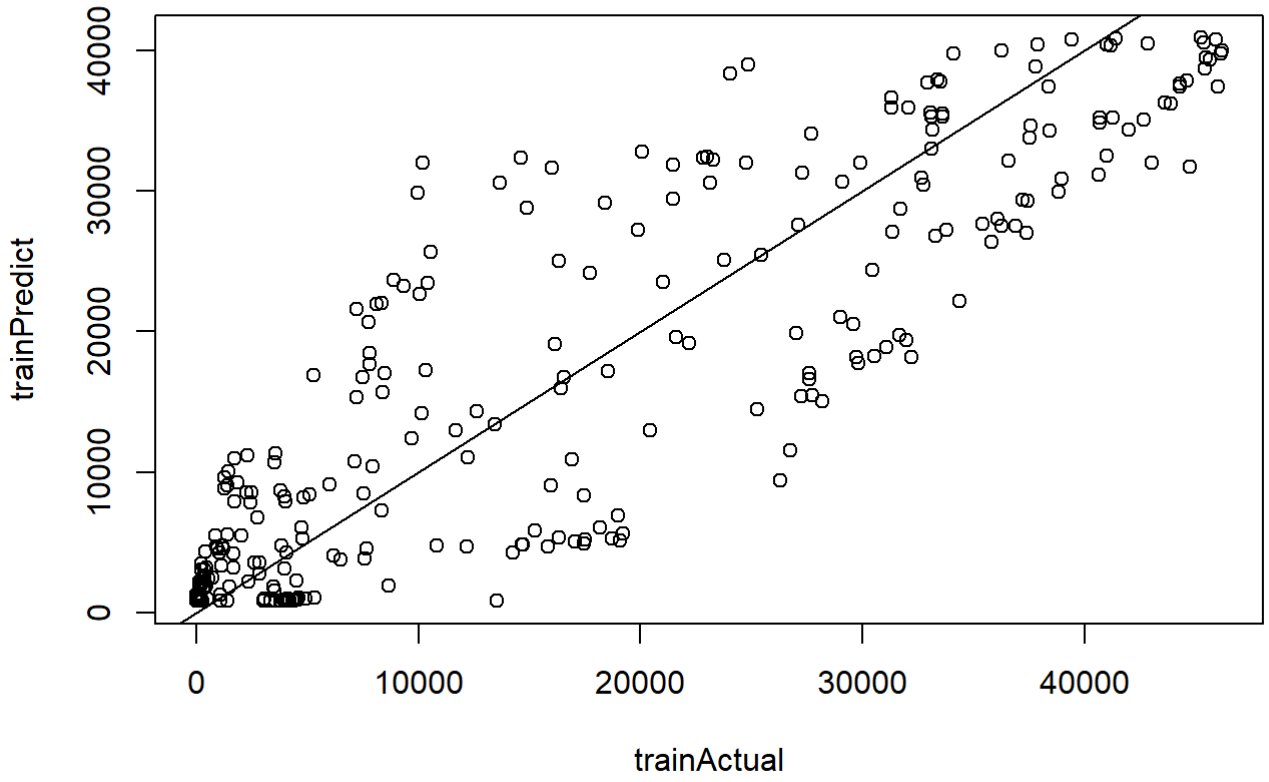
```
## [1] 14108.3
```

```
#summary(trainActual)
#summary(trainPredict)
```

```
cor(trainActual, trainPredict)
```

```
##           [,1]
## [1,] 0.9184858
```

```
plot(trainActual, trainPredict)
abline(a = 0, b = 1)
```




```
testActual = testXY$dataY * spread + min_value
mean(testActual)
```

```
## [1] 5210.124
```

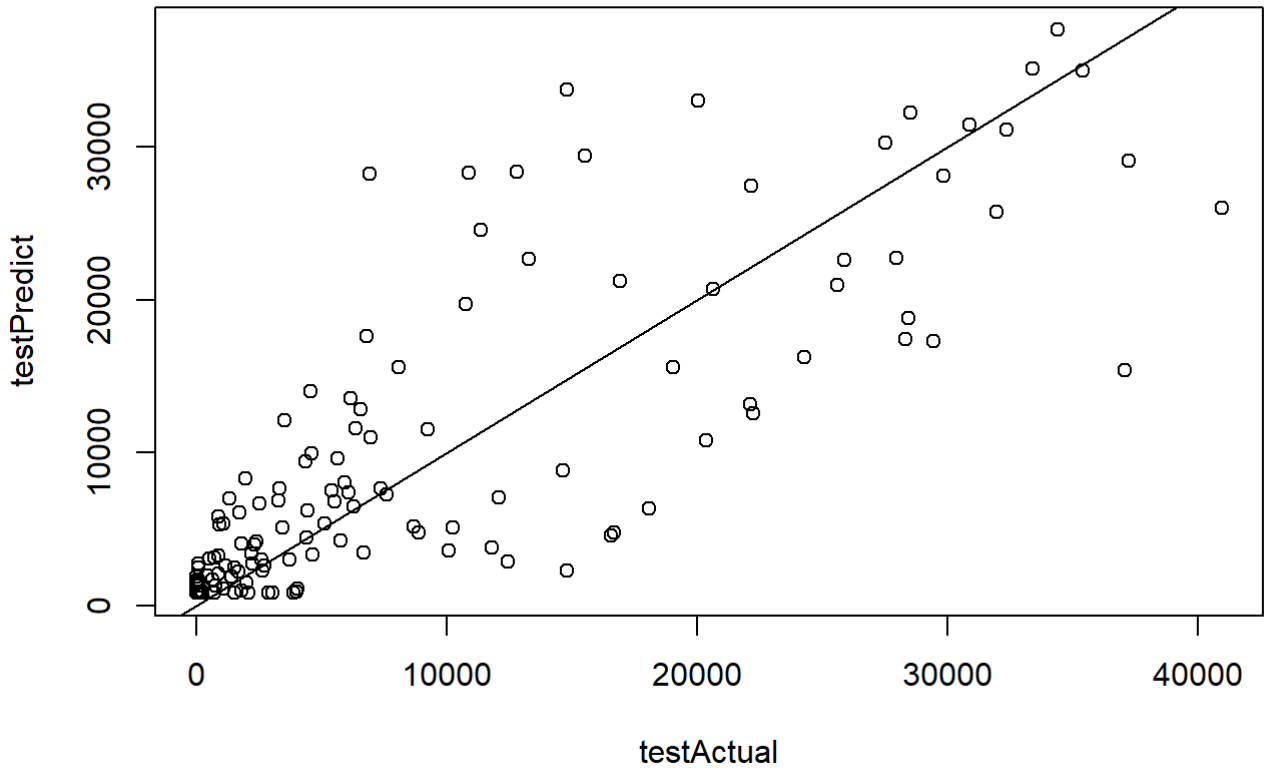
```
sd(testActual)
```

```
## [1] 9215.629
```

```
#summary(testActual)
#summary(testPredict)

cor(testActual, testPredict)
```

```
##           [,1]
## [1,] 0.858131
```

```
plot(testActual, testPredict)
abline(a = 0, b = 1)
```



total Chunghwa data

```
max_value <- max(dataframe$PAC)
min_value <- min(dataframe$PAC)
spread <- max_value - min_value

dataset <- (dataframe$PAC - min_value) / spread #正規化資料
range(dataset)
```

```
## [1] 0 1
```

```
create_dataset <- function(dataset,
                           look_back = 1)
{
  l <- length(dataset)
  dataX <- array(dim = c(l - look_back, look_back))

  for (i in 1:ncol(dataX))
  {
    dataX[, i] <- dataset[i:(l - look_back + i - 1)]
  }

  dataY <- array(
    data = dataset[(look_back + 1):l],
    dim = c(l - look_back, 1))

  return(
    list(
      dataX = dataX,
      dataY = dataY))
} #設定x y 的資料格式

train_size <- as.integer(length(dataset) * 0.67)
test_size <- length(dataset) - train_size

train <- dataset[1:train_size]
test <- dataset[(train_size + 1):length(dataset)]

cat(length(train), length(test))  #切分訓練集與測試集
```

```
## 1007 496
```

```
#1007 496
```

```
for (i in c(1:5)){
  look_back <- i           #設定t-1
  trainXY <- create_dataset(train, look_back)
  testXY <-  create_dataset(test, look_back)

  dim_train <- dim(trainXY$dataX)
  dim_test <- dim(testXY$dataX)

  # reshape input to be [samples, time steps, features]
  dim(trainXY$dataX) <- c(dim_train[1], 1, dim_train[2])
  dim(testXY$dataX) <- c(dim_test[1], 1, dim_test[2])


  #建模訓練
  model <- keras_model_sequential()

  model %>%
    layer_lstm(
      units = 4,
      input_shape = c(1, look_back)) %>%
    layer_dense(
      units = 1) %>%
    compile(
      loss = 'mean_squared_error',
      optimizer = 'adam') %>%
    fit(trainXY$dataX,
        trainXY$dataY,
        epochs = 30,  #調30代就差不多
        batch_size = 1,
        verbose = 2)

  #訓練結果
  trainScore <- model %>%
    evaluate(
      trainXY$dataX,
      trainXY$dataY,
      verbose = 2)

  testScore <- model %>%
    evaluate(
      testXY$dataX,
      testXY$dataY,
      verbose = 2)

  # trainScore_inv = trainScore * spread + min_value

  train_RMSE <- sprintf(
    'Train Score: %.4f MSE (%.4f RMSE)',
    trainScore * spread^2,
    sqrt(trainScore) * spread)
  print(train_RMSE)


  test_RMSE <- sprintf(
    'Test Score: %.4f MSE (%.4f RMSE)',
    testScore * spread^2,
    sqrt(testScore) * spread)
```

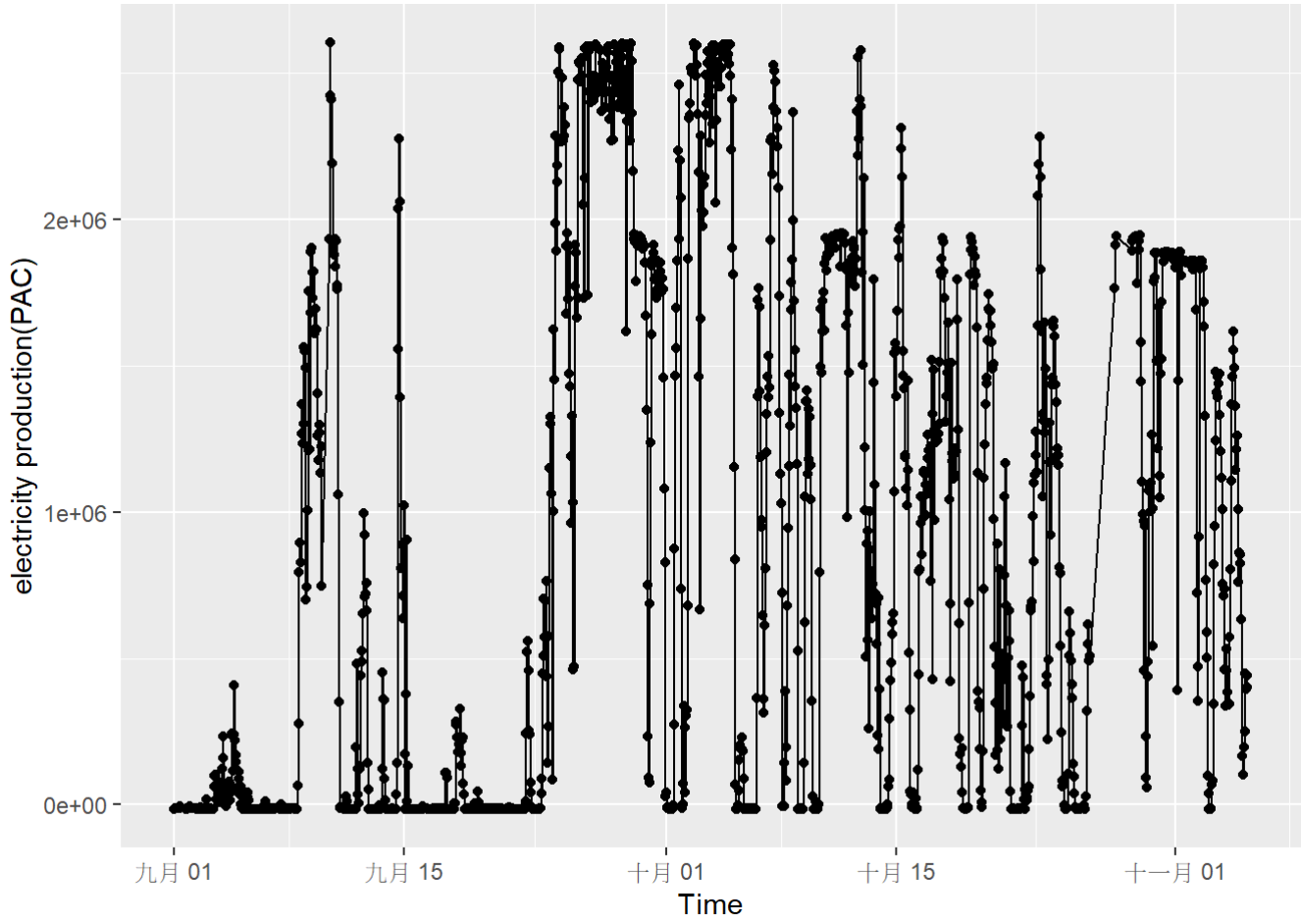
```
  print(test_RMSE)


  #把訓練集的擬合值、測試及的預測值和原始數據畫在一起
  #灰線是真實數據，深藍色是訓練集的訓練結果，淺藍色是預測集的預測值
  trainPredict <- model %>%
    predict(
      trainXY$dataX,
      verbose = 2)
  testPredict <- model %>%
    predict(
      testXY$dataX,
      verbose = 2)

  trainPredict <- trainPredict * spread + min_value
  testPredict <- testPredict * spread + min_value

  df <- data.frame(
    index = 1:length(dataset),
    value = dataset * spread + min_value,
    type = 'raw') %>%
    rbind(
      data.frame(
        index = 1:length(trainPredict) + look_back,
        value = trainPredict,
        type = 'train')) %>%
    rbind(
      data.frame(
        index = 1:length(testPredict) + look_back + length(train),
        value = testPredict,
        type = 'test'))



  mean((trainPredict - mean(trainPredict))^2)
  sqrt(mean((trainPredict - mean(trainPredict))^2))

  trainActual = trainXY$dataY * spread + min_value
  mean(trainActual)
  sd(trainActual)

  train_cor <- cor(trainActual, trainPredict)
  print(train_cor)


  testActual = testXY$dataY * spread + min_value
  mean(testActual)
  sd(testActual)

  test_cor <- cor(testActual, testPredict)
  print(test_cor)

}
```

```
## [1] "Train Score: 27289719.1606 MSE (5223.9563 RMSE)"
## [1] "Test Score: 24865721.5359 MSE (4986.5541 RMSE)"
##           [,1]
## [1,] 0.9065908
##           [,1]
## [1,] 0.8934987
## [1] "Train Score: 22217644.2743 MSE (4713.5596 RMSE)"
## [1] "Test Score: 20328863.3442 MSE (4508.7541 RMSE)"
##           [,1]
## [1,] 0.9252816
##        [,1]
## [1,] 0.9134
## [1] "Train Score: 19219340.1915 MSE (4383.9868 RMSE)"
## [1] "Test Score: 20576109.4516 MSE (4536.0897 RMSE)"
##           [,1]
## [1,] 0.9349341
##           [,1]
## [1,] 0.9135184
## [1] "Train Score: 18384115.1825 MSE (4287.6701 RMSE)"
## [1] "Test Score: 19617713.3052 MSE (4429.1888 RMSE)"
##           [,1]
## [1,] 0.9378912
##           [,1]
## [1,] 0.9166485
## [1] "Train Score: 17047898.3351 MSE (4128.9101 RMSE)"
## [1] "Test Score: 18681834.4706 MSE (4322.2488 RMSE)"
##           [,1]
## [1,] 0.9428206
##           [,1]
## [1,] 0.9200075
```

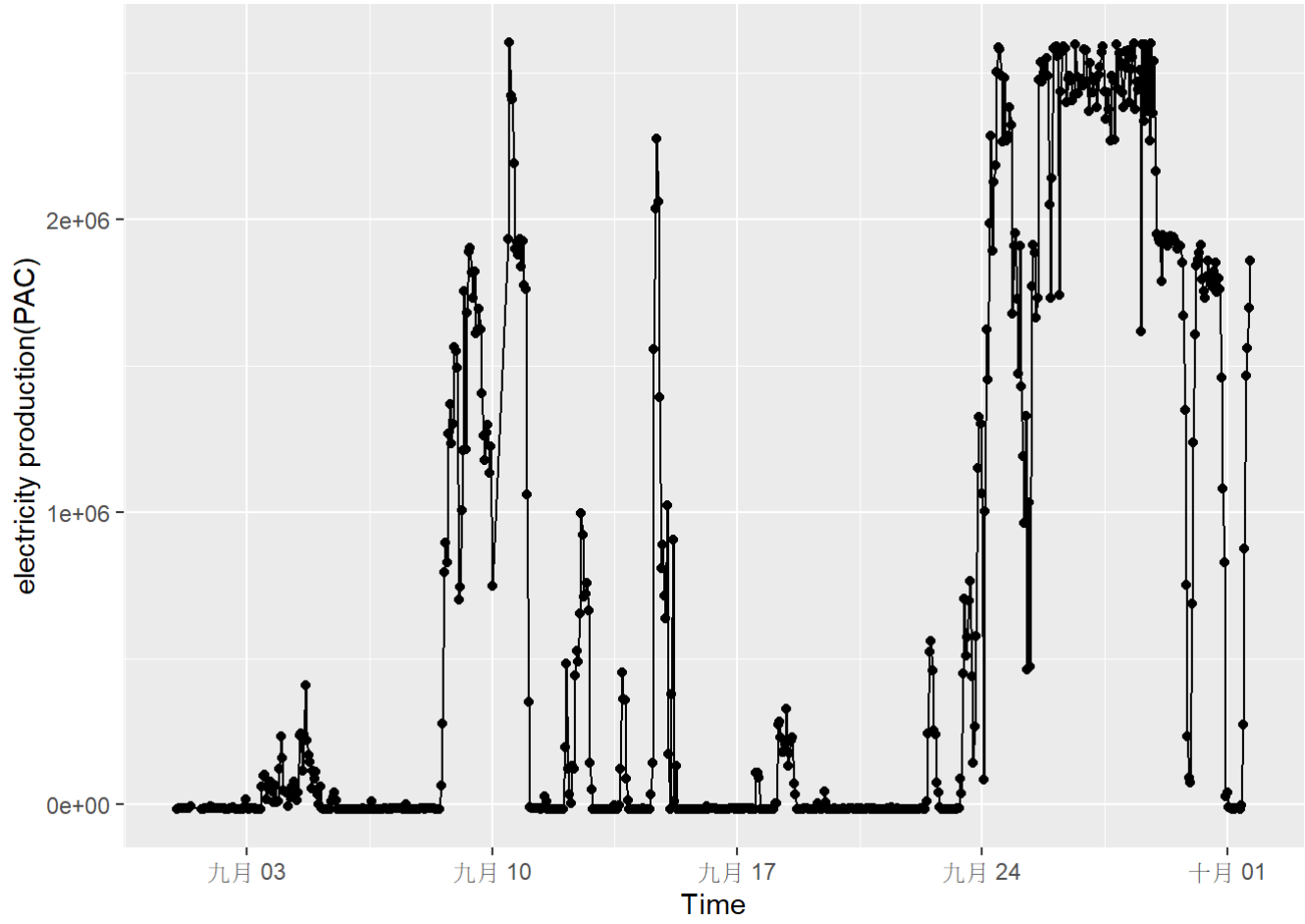
6.2 wind data

We only use wind driven generator observed from Formosa.

```
dataframe <- read.csv(
  './Data_1105/17.csv')

date <- substr(dataframe$MYTIME, 1,10)
time <- paste(substr(dataframe$MYTIME,12, 13),":00:00",sep = "")
dataframe$MYTIME <- as.POSIXct(paste(date,time, sep = " "))


dataframe <- dataframe[order(dataframe$MYTIME),]

ggplot(
  data = dataframe,
  mapping = aes(
    x = MYTIME,
    y = PAC)) +
  geom_line() +
  geom_point() +
  labs(x = "Time") +
  labs(y = "electricity production(PAC)")
```



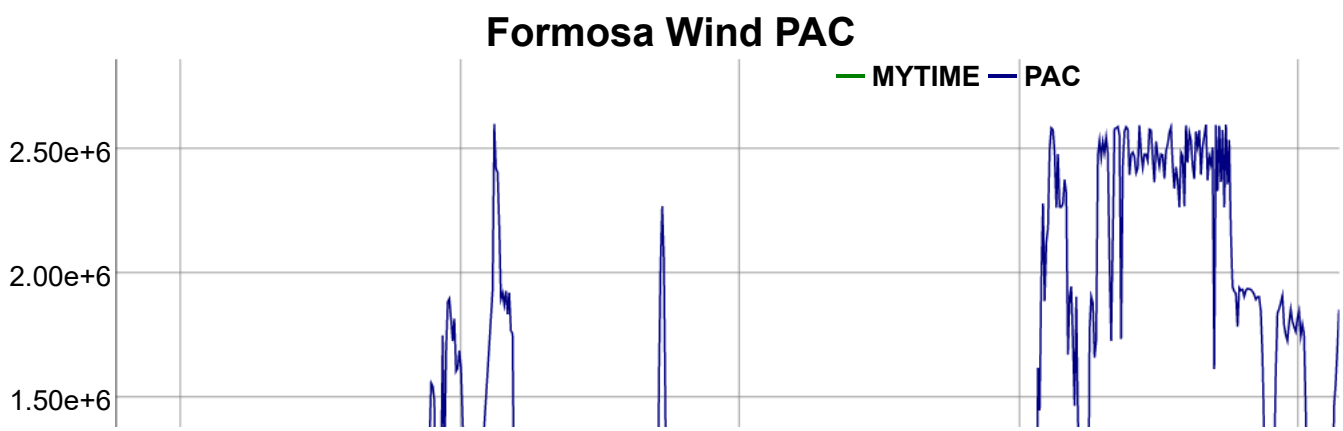

```
sept <- dataframe[1:720,]

ggplot(
  data = sept,
  mapping = aes(
    x = MYTIME,
    y = PAC)) +
  geom_line() +
  geom_point() +
  labs(x = "Time") +
  labs(y = "electricity production(PAC)")
```

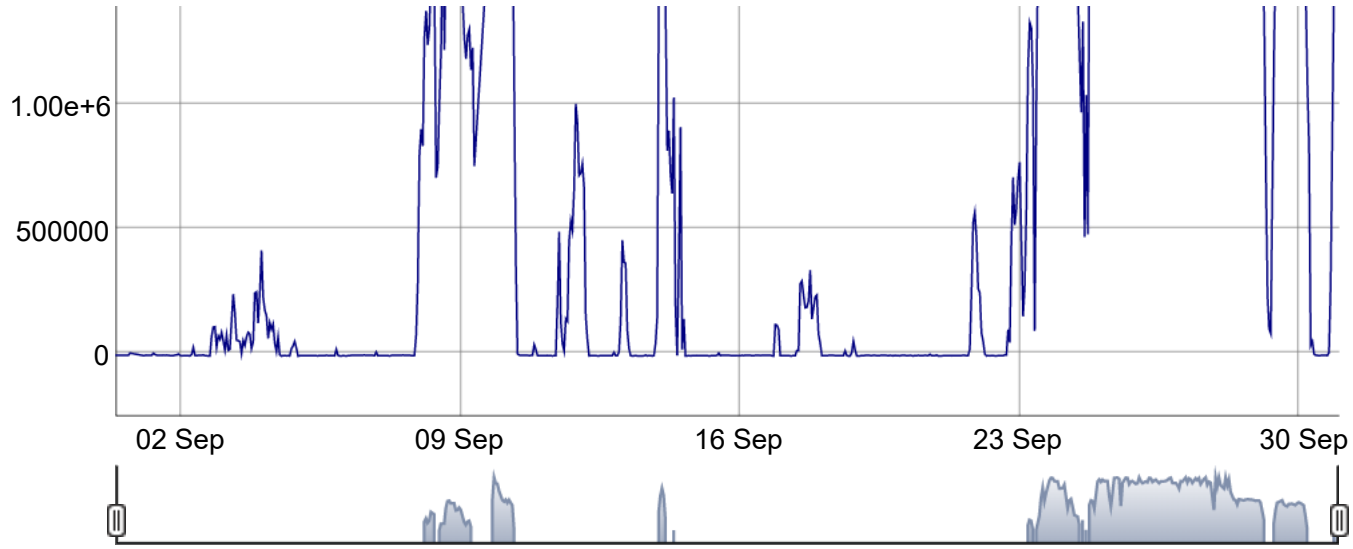



```
names(sept)
```

```
##  [1] "SESSIONS"      "RANK"          "MYTIME"        "MYHOUR"
##  [5] "INITIAL_DATE"  "FCST"          "STNNO"         "FCST_DATETIME"
##  [9] "WS10M"         "WS65M"         "SWDOWN"        "USER_ID"
## [13] "INITIAL_TIME"  "EQUIPMENT_ID"  "T2"            "SHUM2"
## [17] "RAIN"          "PAC"           "VAC"           "IAC"
## [21] "EAC"
```

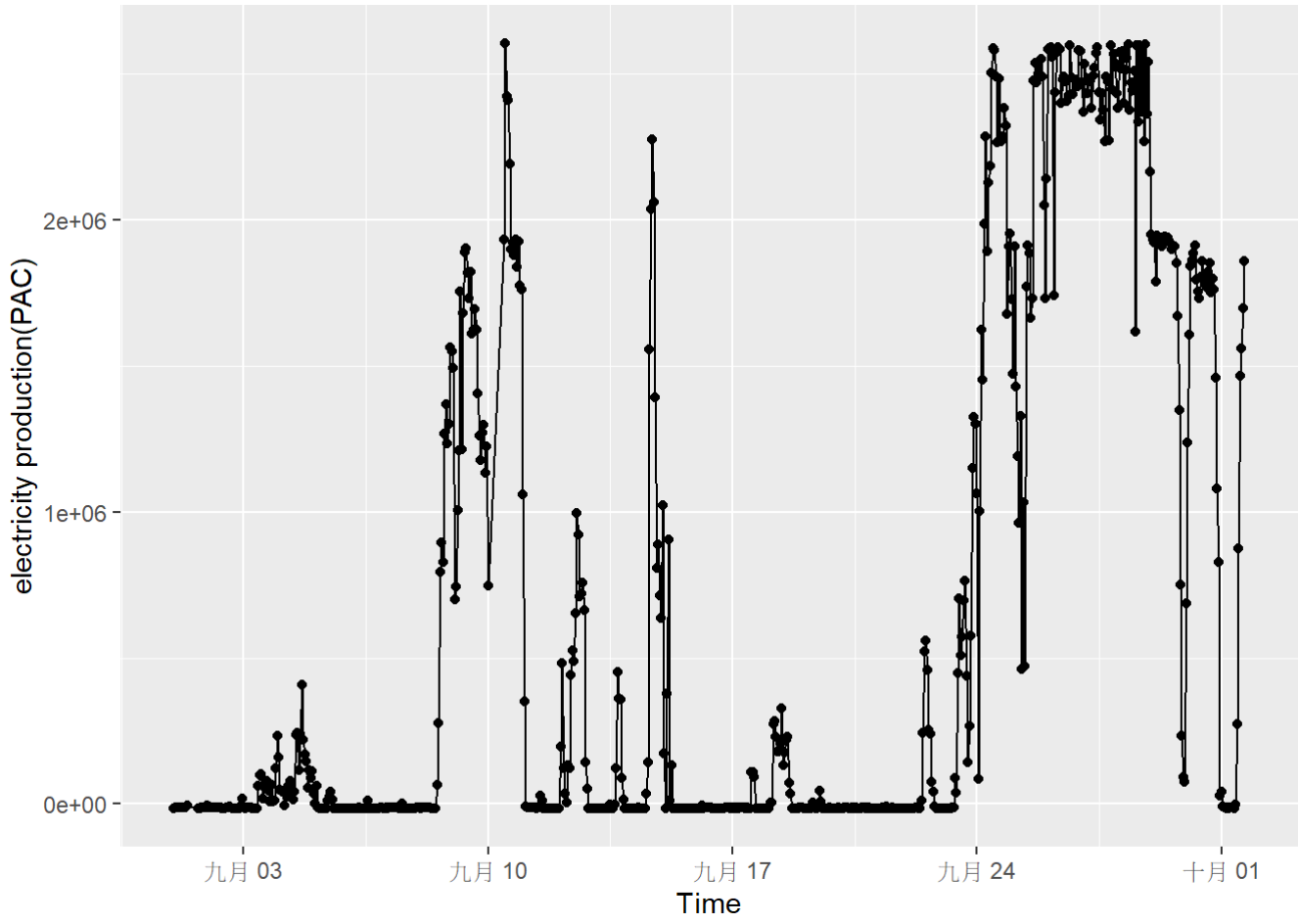
```
sept <- sept[,c(3,18)]

library(dygraphs)
sept_xts <-  xts(sept, order.by = sept$MYTIME)

dygraph(sept_xts, main = " Formosa Wind PAC") %>%
  dyRangeSelector()
```





only September

```
sept <- dataframe[1:720,]

ggplot(
  data = sept,
  mapping = aes(
    x = MYTIME,
    y = PAC)) +
  geom_line() +
  geom_point() +
  labs(x = "Time") +
  labs(y = "electricity production(PAC)")
```



normalization

```
max_value <- max(sept$PAC)
min_value <- min(sept$PAC)
spread <- max_value - min_value

dataset <- (sept$PAC - min_value) / spread #正規化資料
range(dataset)
```

```
## [1] 0 1
```

build the model

```
create_dataset <- function(dataset,
                           look_back = 1)
{
  l <- length(dataset)
  dataX <- array(dim = c(l - look_back, look_back))

  for (i in 1:ncol(dataX))
  {
    dataX[, i] <- dataset[i:(l - look_back + i - 1)]
  }

  dataY <- array(
    data = dataset[(look_back + 1):l],
    dim = c(l - look_back, 1))

  return(
    list(
      dataX = dataX,
      dataY = dataY))
} #設定x y 的資料格式

train_size <- as.integer(length(dataset) * 0.67)
test_size <- length(dataset) - train_size

train <- dataset[1:train_size]
test <- dataset[(train_size + 1):length(dataset)]

cat(length(train), length(test))  #切分訓練集與測試集
```

```
## 482 238
```

```
# 482 238

look_back <- 1           #設定t-1
trainXY <- create_dataset(train, look_back)
testXY <-  create_dataset(test, look_back)

dim_train <- dim(trainXY$dataX)
dim_test <- dim(testXY$dataX)

# reshape input to be [samples, time steps, features]
dim(trainXY$dataX) <- c(dim_train[1], 1, dim_train[2])
dim(testXY$dataX) <- c(dim_test[1], 1, dim_test[2])
```

result

```
model <- keras_model_sequential()

model %>%
  layer_lstm(
    units = 4,
    input_shape = c(1, look_back)) %>%
  layer_dense(
    units = 1) %>%
  compile(
    loss = 'mean_squared_error',
    optimizer = 'adam') %>%
  fit(trainXY$dataX,
      trainXY$dataY,
      epochs = 30,  #調30代就差不多
      batch_size = 1,
      verbose = 2)

#訓練結果
trainScore <- model %>%
  evaluate(
    trainXY$dataX,
    trainXY$dataY,
    verbose = 2)

testScore <- model %>%
  evaluate(
    testXY$dataX,
    testXY$dataY,
    verbose = 2)

# trainScore_inv = trainScore * spread + min_value

sprintf(
  'Train Score: %.4f MSE (%.4f RMSE)',
  trainScore * spread^2,
  sqrt(trainScore) * spread)
```

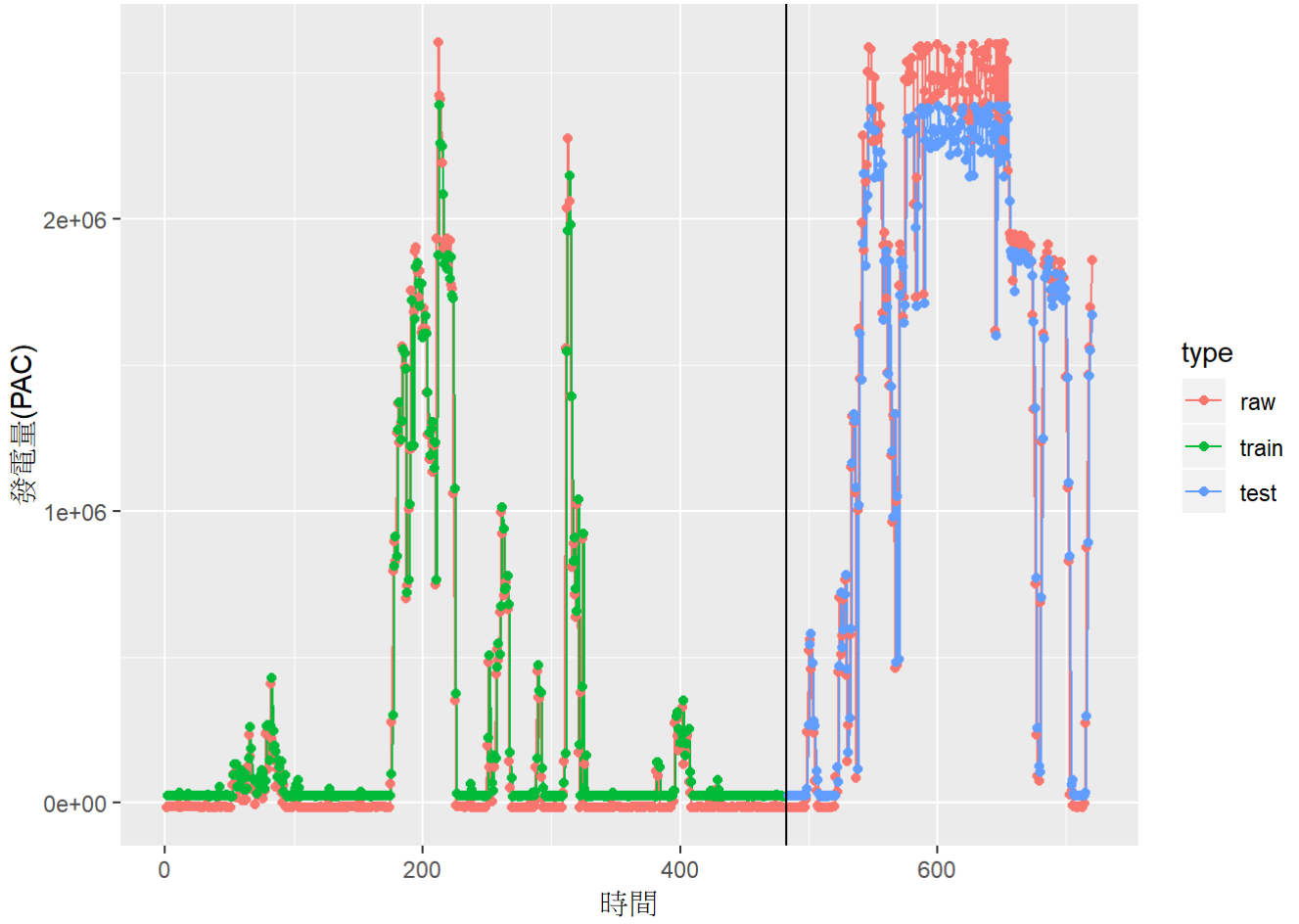
```
## [1] "Train Score: 26764925866.6506 MSE (163599.8957 RMSE)"
```

```
sprintf(
  'Test Score: %.4f MSE (%.4f RMSE)',
  testScore * spread^2,
  sqrt(testScore) * spread)
```

```
## [1] "Test Score: 80477556570.6075 MSE (283685.6651 RMSE)"
```

visualization of the result

```
trainPredict <- model %>%
  predict(
    trainXY$dataX,
    verbose = 2)
testPredict <- model %>%
  predict(
    testXY$dataX,
    verbose = 2)

trainPredict <- trainPredict * spread + min_value
testPredict <- testPredict * spread + min_value

df <- data.frame(
  index = 1:length(dataset),
  value = dataset * spread + min_value,
  type = 'raw') %>%
  rbind(
    data.frame(
      index = 1:length(trainPredict) + look_back,
      value = trainPredict,
      type = 'train')) %>%
  rbind(
    data.frame(
      index = 1:length(testPredict) + look_back + length(train),
      value = testPredict,
      type = 'test'))

ggplot(data = df) +
  geom_line(
    mapping = aes(
      x = index,
      y = value,
      color = type)) +
  geom_point(
    mapping = aes(
      x = index,
      y = value,
      color = type)) +
  geom_vline(
    xintercept = length(train) + 0.5) +
  labs(x = "時間") +
  labs(y = "發電量(PAC)")
```




```
mean((trainPredict - mean(trainPredict))^2)
```

```
## [1] 246843203242
```

```
sqrt(mean((trainPredict - mean(trainPredict))^2))
```

```
## [1] 496833.2
```

```
trainActual = trainXY$dataY * spread + min_value
mean(trainActual)
```

```
## [1] 210391.1
```

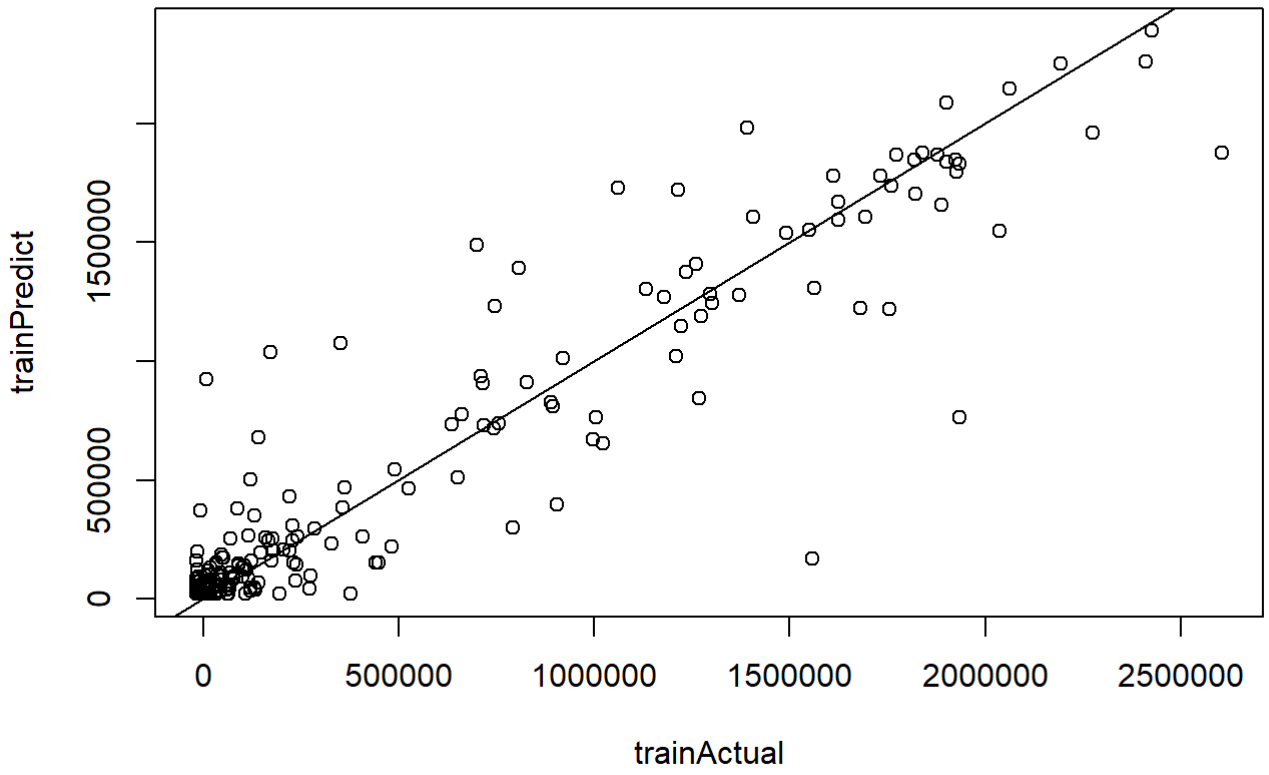
```
sd(trainActual)
```

```
## [1] 521377.3
```

```
#summary(trainActual)
#summary(trainPredict)

cor(trainActual, trainPredict)
```

```
##           [,1]
## [1,] 0.9510238
```

```
plot(trainActual, trainPredict)
abline(a = 0, b = 1)
```




```
testActual = testXY$dataY * spread + min_value
mean(testActual)
```

```
## [1] 1526237
```

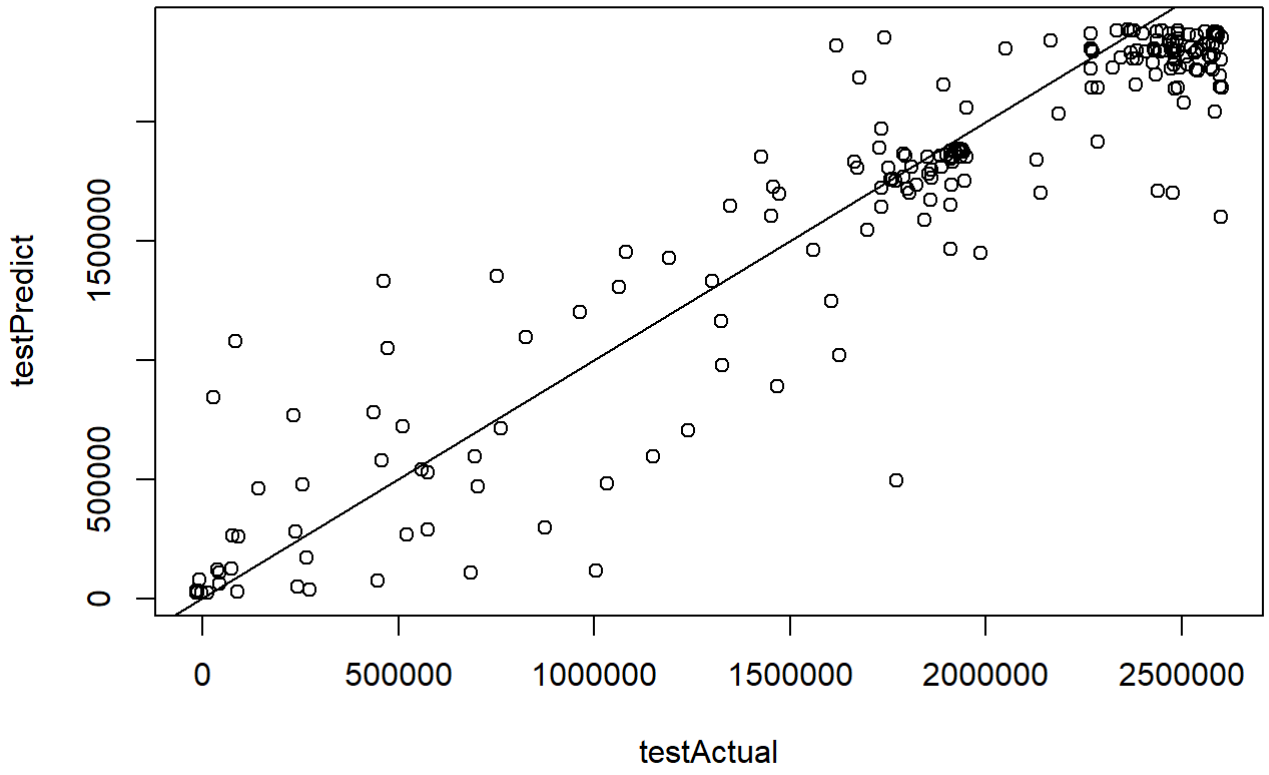
```
sd(testActual)
```

```
## [1] 979714.4
```

```
#summary(testActual)
#summary(testPredict)

cor(testActual, testPredict)
```

```
##           [,1]
## [1,] 0.9609638
```

```
plot(testActual, testPredict)
abline(a = 0, b = 1)
```



total Formosa data

```
max_value <- max(dataframe$PAC)
min_value <- min(dataframe$PAC)
spread <- max_value - min_value

dataset <- (dataframe$PAC - min_value) / spread #正規化資料
range(dataset)
```

```
## [1] 0 1
```

```
create_dataset <- function(dataset,
                           look_back = 1)
{
  l <- length(dataset)
  dataX <- array(dim = c(l - look_back, look_back))

  for (i in 1:ncol(dataX))
  {
    dataX[, i] <- dataset[i:(l - look_back + i - 1)]
  }

  dataY <- array(
    data = dataset[(look_back + 1):l],
    dim = c(l - look_back, 1))

  return(
    list(
      dataX = dataX,
      dataY = dataY))
} #設定x y 的資料格式

train_size <- as.integer(length(dataset) * 0.67)
test_size <- length(dataset) - train_size

train <- dataset[1:train_size]
test <- dataset[(train_size + 1):length(dataset)]

cat(length(train), length(test))
```

```
## 1002 494
```

```
for (i in c(1:5)){
  look_back <- i           #設定t-1
  trainXY <- create_dataset(train, look_back)
  testXY <-  create_dataset(test, look_back)

  dim_train <- dim(trainXY$dataX)
  dim_test <- dim(testXY$dataX)

  # reshape input to be [samples, time steps, features]
  dim(trainXY$dataX) <- c(dim_train[1], 1, dim_train[2])
  dim(testXY$dataX) <- c(dim_test[1], 1, dim_test[2])


  #建模訓練
  model <- keras_model_sequential()

  model %>%
    layer_lstm(
      units = 4,
      input_shape = c(1, look_back)) %>%
    layer_dense(
      units = 1) %>%
    compile(
      loss = 'mean_squared_error',
      optimizer = 'adam') %>%
    fit(trainXY$dataX,
        trainXY$dataY,
        epochs = 30,  #調30代就差不多
        batch_size = 1,
        verbose = 2)

  #訓練結果
  trainScore <- model %>%
    evaluate(
      trainXY$dataX,
      trainXY$dataY,
      verbose = 2)

  testScore <- model %>%
    evaluate(
      testXY$dataX,
      testXY$dataY,
      verbose = 2)

  # trainScore_inv = trainScore * spread + min_value

  train_RMSE <- sprintf(
    'Train Score: %.4f MSE (%.4f RMSE)',
    trainScore * spread^2,
    sqrt(trainScore) * spread)
  print(train_RMSE)


  test_RMSE <- sprintf(
    'Test Score: %.4f MSE (%.4f RMSE)',
    testScore * spread^2,
    sqrt(testScore) * spread)
```

```
  print(test_RMSE)


  #把訓練集的擬合值、測試及的預測值和原始數據畫在一起
  #灰線是真實數據，深藍色是訓練集的訓練結果，淺藍色是預測集的預測值
  trainPredict <- model %>%
    predict(
      trainXY$dataX,
      verbose = 2)
  testPredict <- model %>%
    predict(
      testXY$dataX,
      verbose = 2)

  trainPredict <- trainPredict * spread + min_value
  testPredict <- testPredict * spread + min_value

  df <- data.frame(
    index = 1:length(dataset),
    value = dataset * spread + min_value,
    type = 'raw') %>%
    rbind(
      data.frame(
        index = 1:length(trainPredict) + look_back,
        value = trainPredict,
        type = 'train')) %>%
    rbind(
      data.frame(
        index = 1:length(testPredict) + look_back + length(train),
        value = testPredict,
        type = 'test'))



  mean((trainPredict - mean(trainPredict))^2)
  sqrt(mean((trainPredict - mean(trainPredict))^2))

  trainActual = trainXY$dataY * spread + min_value
  mean(trainActual)
  sd(trainActual)

  train_cor <- cor(trainActual, trainPredict)
  print(train_cor)


  testActual = testXY$dataY * spread + min_value
  mean(testActual)
  sd(testActual)

  test_cor <- cor(testActual, testPredict)
  print(test_cor)

}
```

```
## [1] "Train Score: 56458689078.3670 MSE (237610.3724 RMSE)"
## [1] "Test Score: 76728026542.0260 MSE (276998.2429 RMSE)"
##           [,1]
## [1,] 0.9724805
##           [,1]
## [1,] 0.9274039
## [1] "Train Score: 52679275760.2919 MSE (229519.6631 RMSE)"
## [1] "Test Score: 69364116254.6188 MSE (263370.6822 RMSE)"
##           [,1]
## [1,] 0.9732155
##           [,1]
## [1,] 0.9302003
## [1] "Train Score: 56581454758.2514 MSE (237868.5661 RMSE)"
## [1] "Test Score: 73271232148.7627 MSE (270686.5940 RMSE)"
##          [,1]
## [1,] 0.973222
##           [,1]
## [1,] 0.9290294
## [1] "Train Score: 54502665290.3705 MSE (233458.0590 RMSE)"
## [1] "Test Score: 74472922999.9307 MSE (272897.2755 RMSE)"
##           [,1]
## [1,] 0.9732349
##          [,1]
## [1,] 0.929162
## [1] "Train Score: 53451869194.8123 MSE (231196.6029 RMSE)"
## [1] "Test Score: 72214472094.2528 MSE (268727.5053 RMSE)"
##           [,1]
## [1,] 0.9732751
##           [,1]
## [1,] 0.9299514
```

Ref

[1] Francois, Chollet and Allaire, J.J. (2017), Deep Learning With R, Manning, p.167-p.222.

[2] Hochreiter, S. and Schmidhuber, J. (1997), Long short-term memory, Neural Computation, 9(8), p.1735-1780.

[3] I code so I am (mikechenx) (2017), 以100張圖理解 Neural Network – 觀念與實踐系列。https://ithelp.ithome.com.tw/articles/10193469 (https://ithelp.ithome.com.tw/articles/10193469)

[4] Lahouar, A. and Slama, Ben Hadj (2017), Hour-ahead wind power forecast based on random forests, Renewable Energy, 109, p.529-541.

[5] 黃小偉 (2018)，基於 Keras 用 LSTM 網路做時間序列預測。https://zhuanlan.zhihu.com/p/41933062 (https://zhuanlan.zhihu.com/p/41933062)