FinalEnhanced

Yu-Chen Xue

2018 年 6 月 30 日

0. 使用 GAM 分析 EnerNOC Dataset 中商用大樓的用電狀況

1. Dataset 介紹：

名稱：EnerNOC GreenButton Data
來源：open enernoc data
https://github.com/PetoLau/petolau.github.io/tree/master/_rmd/ 簡述：原始資料集由 EnerNOC 電力公司提供，它依循時間序列記錄了 2012 年 100 棟不記名的建築物每 5 分鐘的用電情況。經過整理後的資料記載了每半個小時的用電狀況，其解釋變數如下
- value: 特定時間點下的電耗值
- week: 週次 - date: 日期
- type: 大樓類型

2. 使用的方法：

Generalized additive model (GAM)

3. 大綱

1. 資料集分析
2. 模型選擇
3. 分析解釋變量的重要情況
4. 預測電耗

1. 資料集分析

引入必要的模組

```
library(feather)
library(data.table)
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-23. For overview type 'help("mgcv-package")'.
```

```
library(car)
```

```
## Loading required package: carData
```

```
library(ggplot2)
library(dplyr, warn.conflicts = FALSE)
```

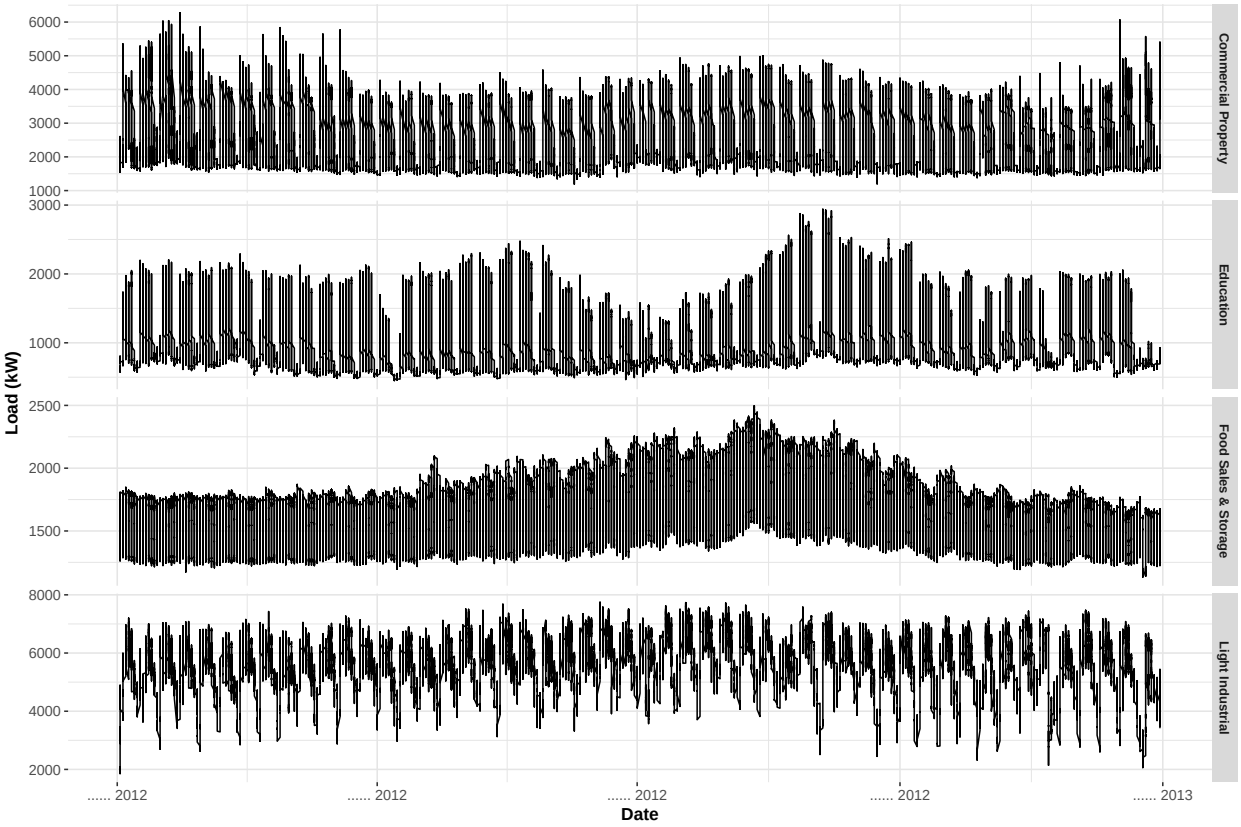
讀取資料

```
DT <- as.data.table(read_feather("D:/WORKSPACE/RProjects/EnorNOC-GAM/DT_4_ind"))
str(DT)
```

```
## Classes 'data.table' and 'data.frame':   70080 obs. of  5 variables:
##  $ date_time: POSIXct, format: "2012-01-02 00:00:00" "2012-01-02 00:30:00" ...
##  $ value    : num  1590 1564 1560 1585 1604 ...
##  $ week     : chr  "Monday" "Monday" "Monday" "Monday" ...
##  $ date     : Date, format: "2012-01-02" "2012-01-02" ...
##  $ type     : chr  "Commercial Property" "Commercial Property" "Commercial Property" "Commercia
##  - attr(*, ".internal.selfref")=<externalptr>
```

畫圖查看資料

```
ggplot(data = DT, aes(x = date, y = value)) +
  geom_line() +
  facet_grid(type ~ ., scales = "free_y") +
  theme(panel.border = element_blank(),
        panel.background = element_blank(),
        panel.grid.minor = element_line(colour = "grey90"),
        panel.grid.major = element_line(colour = "grey90"),
        panel.grid.major.x = element_line(colour = "grey90"),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 12, face = "bold"),
        strip.text = element_text(size = 9, face = "bold")) +
  labs(x = "Date", y = "Load (kW)")
```



可以看出 Food Sales & Storage 這一類的用電情況不隨工作日/雙休日而變化

為了方便描述電耗與週次的關係，這裡使用 package car 中的 function record，新增一個欄位，記載週次所對應的數字

```
DT[, week_num := as.integer(car::recode(week,
    "'Monday'='1';'Tuesday'='2';'Wednesday'='3';'Thursday'='4';
    'Friday'='5';'Saturday'='6';'Sunday'='7'"))]
unique(DT[, week])
```

```
## [1] "Monday"    "Tuesday"   "Wednesday" "Thursday"  "Friday"    "Saturday"
## [7] "Sunday"
```

```
unique(DT[, week_num])
```

```
## [1] 1 2 3 4 5 6 7
```

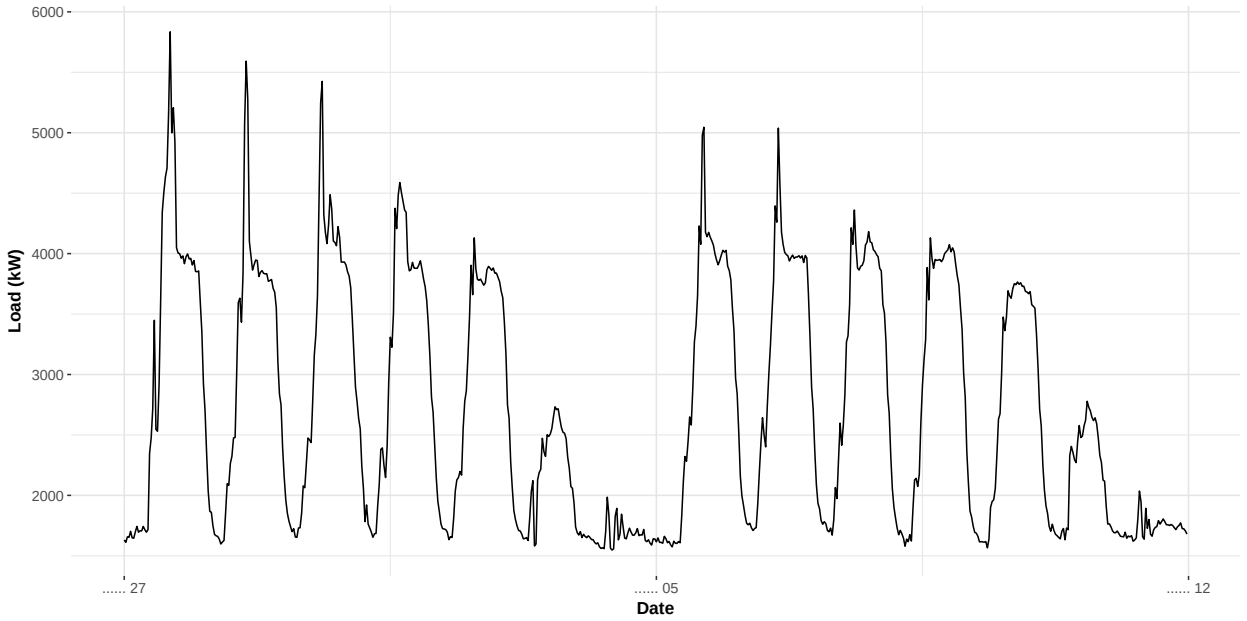
從讀取的資料中獲取 industry, date, weekday and period 等信息，並使用變量來儲存。因為每半個小時觀察一次，所以一天的資料由 48 筆連續的觀察資料組成，因此有 `period <- 48`

```
n_type <- unique(DT[, type])
n_date <- unique(DT[, date])
n_weekdays <- unique(DT[, week])
period <- 48
```

截取兩個禮拜內的商業用樓房的電耗記錄，並儲存在 data_r 變量中。之後畫圖展示之。

`type == n_type[1]` 表示 "Commercial Property", `date %in% n_date[57:70]` 表示兩個禮拜

```
data_r <- DT[(type == n_type[1] & date %in% n_date[57:70])]

ggplot(data_r, aes(date_time, value)) +
  geom_line() +
  theme(panel.border = element_blank(),
        panel.background = element_blank(),
        panel.grid.minor = element_line(colour = "grey90"),
        panel.grid.major = element_line(colour = "grey90"),
        panel.grid.major.x = element_line(colour = "grey90"),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 12, face = "bold")) +
  labs(x = "Date", y = "Load (kW)")
```



根據每天的週期性變化和每週的週期性變化，重新構建資料

```
N <- nrow(data_r) # train set 中的資料筆數
window <- N / period # train set 所囊括的天數
matrix_gam <- data.table(Load = data_r[, value],
                         Daily = rep(1:period, window),
                         Weekly = data_r[, week_num])
head(matrix_gam)
```

```
##        Load Daily Weekly
## 1: 1630.875     1      1
## 2: 1611.201     2      1
## 3: 1657.160     3      1
## 4: 1653.042     4      1
## 5: 1702.316     5      1
## 6: 1648.411     6      1
```

2. 模型選擇

使用 mgcv 套件包的 gam 函數建立 GAM 模型，其中每天的週期性變化採用 cubic regression spline 模式來描述，每週的週期性變化採用 P-splines 來描述。

```
gam_1 <- gam(Load ~ s(Daily, bs = "cr", k = period) +
               s(Weekly, bs = "ps", k = 7),
             data = matrix_gam,
             family = gaussian)
```

查看模型的 summary

```
summary(gam_1)$r.sq
```

```
## [1] 0.7718406
```

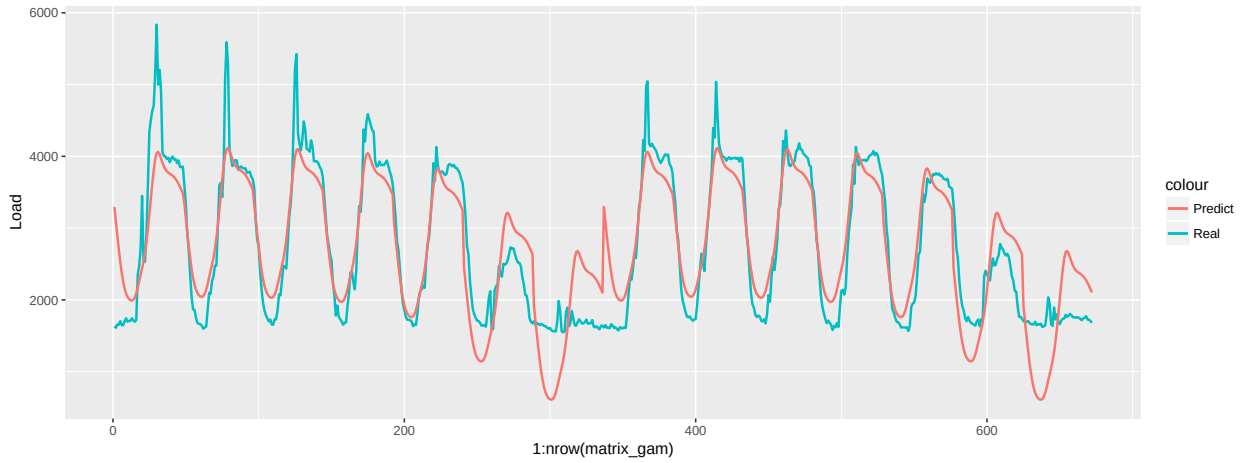
```
summary(gam_1)$sp.criterion
```

```
##   GCV.Cp
## 245544.9
```

GCV 是表示擬合情況的一個指標，越小說明模型的擬合效果越好。另外可以看出，R-sq 的數值不高，這個模型效果不好

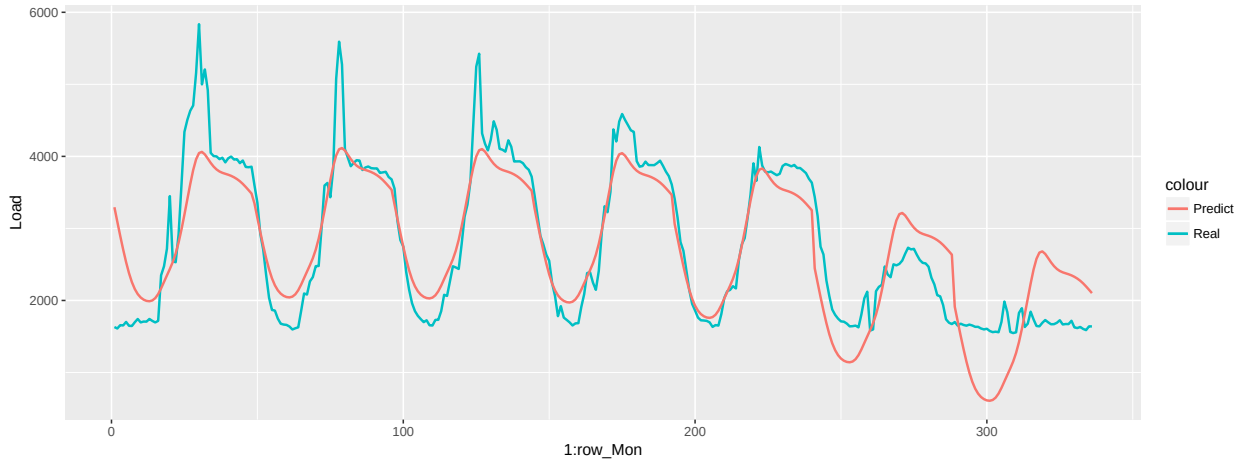
查看這兩個禮拜的用電量的情況實際值與預測值的比較。

```
matrix_gam$Predict=gam_1$fitted.values
ggplot(matrix_gam[1:nrow(matrix_gam),], aes(1:nrow(matrix_gam)))+
           labs('lab')+
           geom_line(aes(y=Load, color="Real"), size = 0.8)+
           geom_line(aes(y = Predict, color = "Predict"), size = 0.8)
```



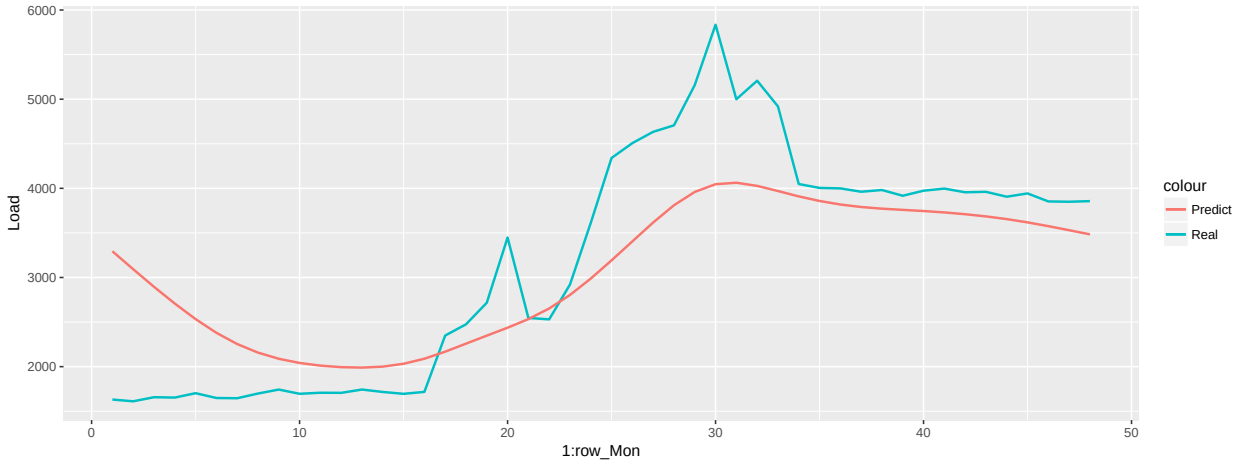
看上去狀況並不好，仔細看第一個禮拜的情況

```
row_Mon <- nrow(matrix_gam)/2
matrix_gam$Predict=gam_1$fitted.values
ggplot(matrix_gam[1:row_Mon,], aes(1:row_Mon))+
           labs('lab')+
           geom_line(aes(y=Load, color="Real"), size = 0.8)+
           geom_line(aes(y = Predict, color = "Predict"), size = 0.8)
```



這個模型只能預測平日用電的趨勢，而具體電耗值卻沒辦法準確預測。在仔細看看禮拜一的用電狀況

```
row_Mon <- nrow(matrix_gam)/14
matrix_gam$Predict=gam_1$fitted.values
ggplot(matrix_gam[1:row_Mon,], aes(1:row_Mon))+
           labs('lab')+
           geom_line(aes(y=Load, color="Real"), size = 0.8)+
           geom_line(aes(y = Predict, color = "Predict"), size = 0.8)
```



問題在於：這天開始的實際用電量與這天結束時的用電量並不吻合，然而模型卻給出了一天週期性的預測結果，這跟實際狀況不一致。因此我們需要換一個思路重建模型。

這回，我們使用 interaction 的方法，把 Daily 和 Weekly 同時進行考量，重建模型，

```
gam_2 <- gam(Load ~ s(Daily, Weekly),
             data = matrix_gam,
             family = gaussian)

summary(gam_2)$r.sq
```

```
## [1] 0.9352108
```

```
summary(gam_2)$sp.criterion
```

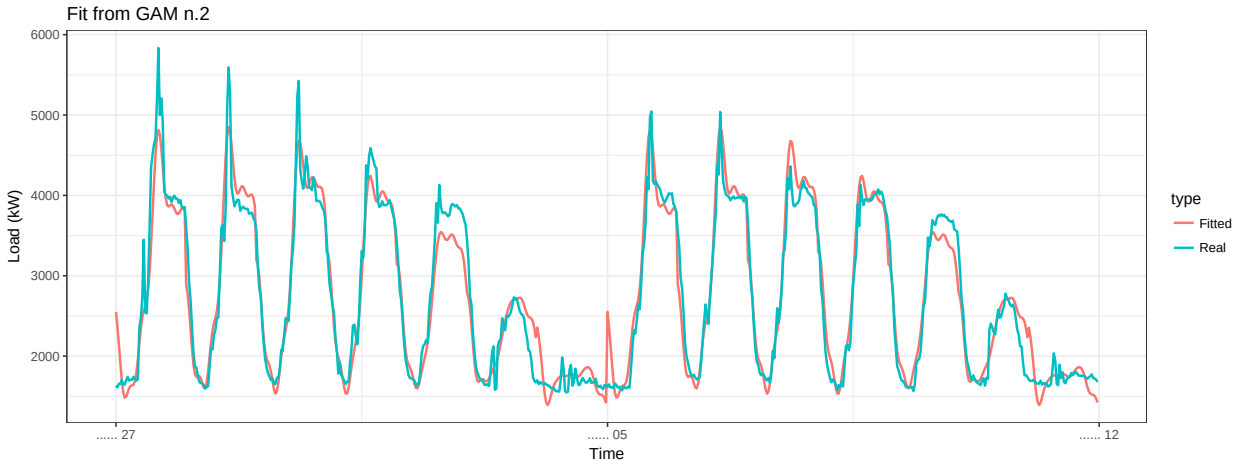
```
##   GCV.Cp
## 71162.37
```

根據 R.sq 和 p-value 可以看出，這個模型比上一個模型來的更好

畫出這個模型的預測結果和實際結果的比較

```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_2$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]
```

```
ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.2")
```



這回明顯可以看出：禮拜一到禮拜四的擬合效果提升了不少

接著使用另一種更進階的 Interaction 方法，這回使用另外一種稱為 “tensor product” 的 smooth function 類型

```
gam_3 <- gam(Load ~ te(Daily, Weekly,
                       bs = c("cr", "ps")),
             data = matrix_gam,
             family = gaussian)

summary(gam_3)$r.sq
```

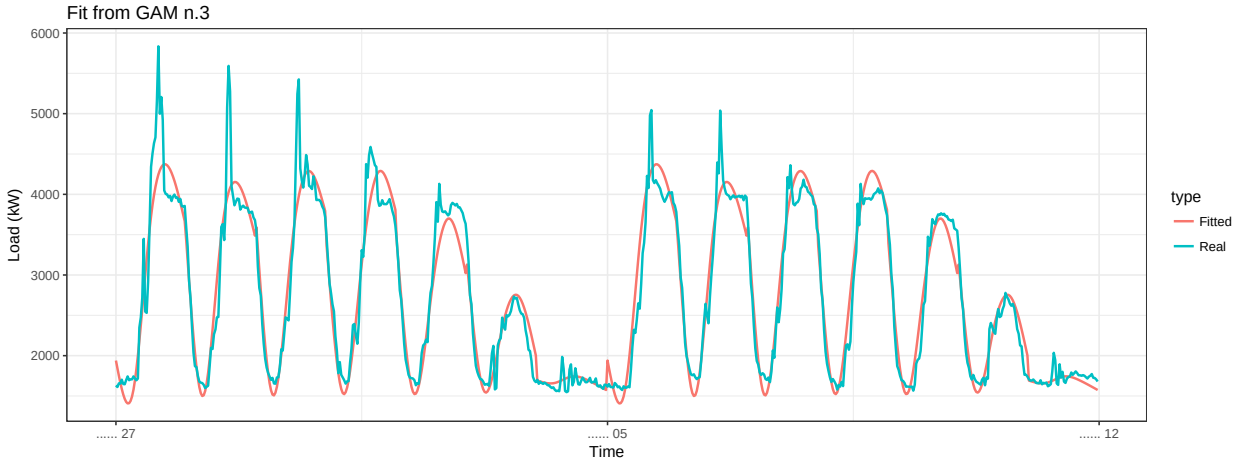
```
## [1] 0.9268452
```

```
summary(gam_3)$sp.criterion
```

```
##  GCV.Cp
## 79724.9
```

作圖分析 gam_3

```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_3$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.3")
```



我們還可以做得更好，比如讓 **smooth function** 的 **knots**（類似維度的概念）更接近每天和每周的週期性情況

```
gam_4 <- gam(Load ~ te(Daily, Weekly,
                       k = c(period, 7),
                       bs = c("cr", "ps")),
             data = matrix_gam,
             family = gaussian)

summary(gam_4)$r.sq
```

```
## [1] 0.9727604
```

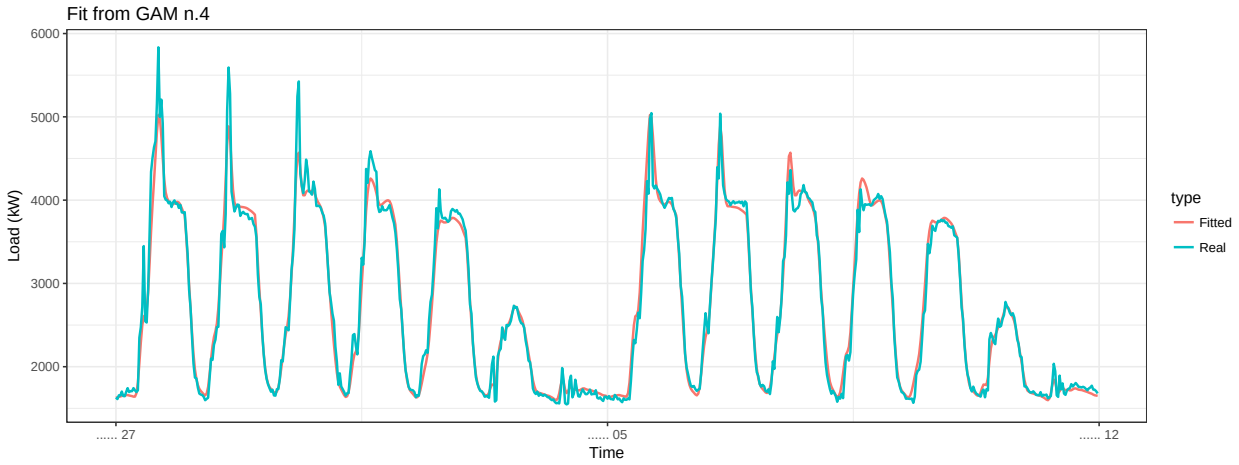
```r
summary(gam_4)$sp.criterion
```

```
##    GCV.Cp
## 34839.46
```

可以看見 R-sq 又上升了一點點，關鍵是 edf value 上升了 5 倍之多。

畫出 gam_4 的圖像

```r
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_4$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.4")
```



好了，加入我們想要貪婪一點，把前面的一些方法都加進來，結果會怎麼樣呢？這裡構建一個 gam_5 來驗證一下

```r
gam_5 <- gam(Load ~ s(Daily, bs = "cr", k = period) +
                    s(Weekly, bs = "ps", k = 7) +
```

```
                    ti(Daily, Weekly,
                       k = c(period, 7),
                       bs = c("cr", "ps")),
              data = matrix_gam,
              family = gaussian)

summary(gam_5)$r.sq
```

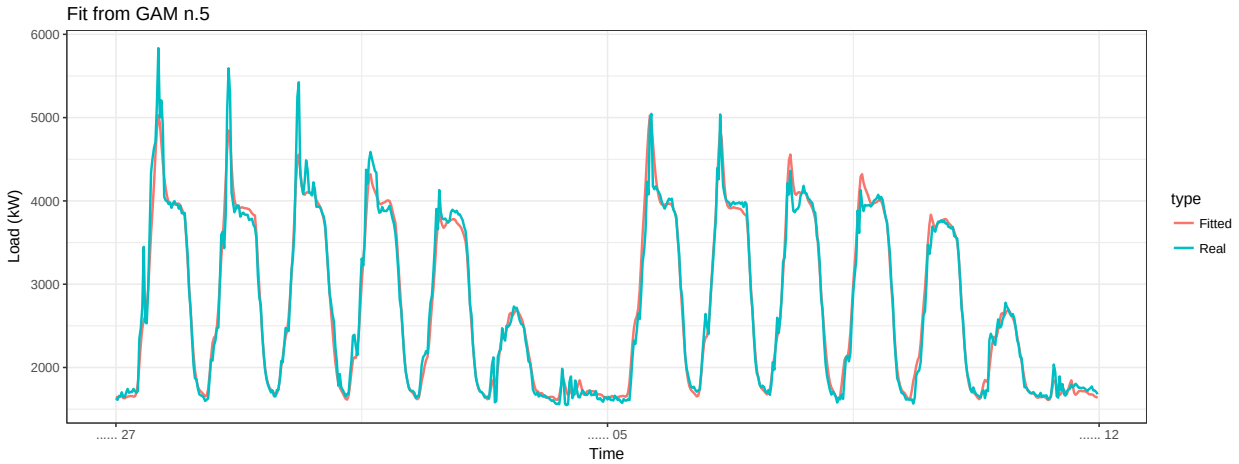
```
## [1] 0.9717469
```

```
summary(gam_5)$sp.criterion
```

```
##    GCV.Cp
## 35772.35
```

雖然 p-value 一樣為 0，R-sq 數值下降了一點，GCV 數值上升了，說明這個模型不如前面的 gam_4 來的好 ## 畫出 gam_5 的圖像

```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_5$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.5")
```



呼～，這回是最後一個招數了，這裡再多嘗試一個 tensor product interactions 方法，並設定 `full = TRUE` 來設定更加嚴格的懲罰條件

```
gam_6 <- gam(Load ~ t2(Daily, Weekly,
                       k = c(period, 7),
                       bs = c("cr", "ps"),
                       full = TRUE),
             data = matrix_gam,
             family = gaussian)

summary(gam_6)$r.sq
```

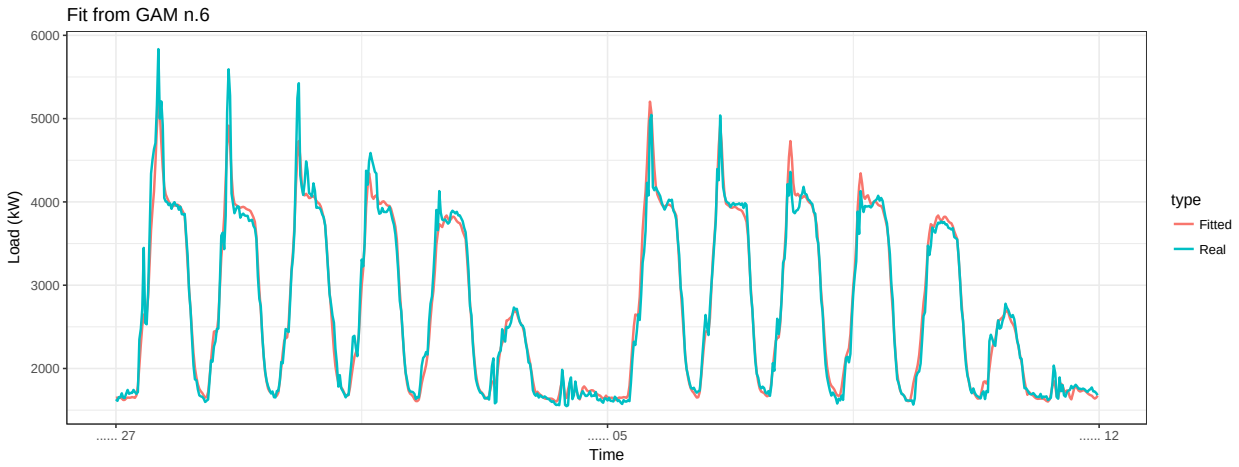
```
## [1] 0.9738273
```

```
summary(gam_6)$sp.criterion
```

```
##   GCV.Cp
## 32230.68
```

畫圖分析 gam_6 的效果

```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_6$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.6")
```



這個圖就看上去漂亮很多了

這麼多模型，要怎麼看誰的效果最好呢？交給萬能的 AIC 來解決吧

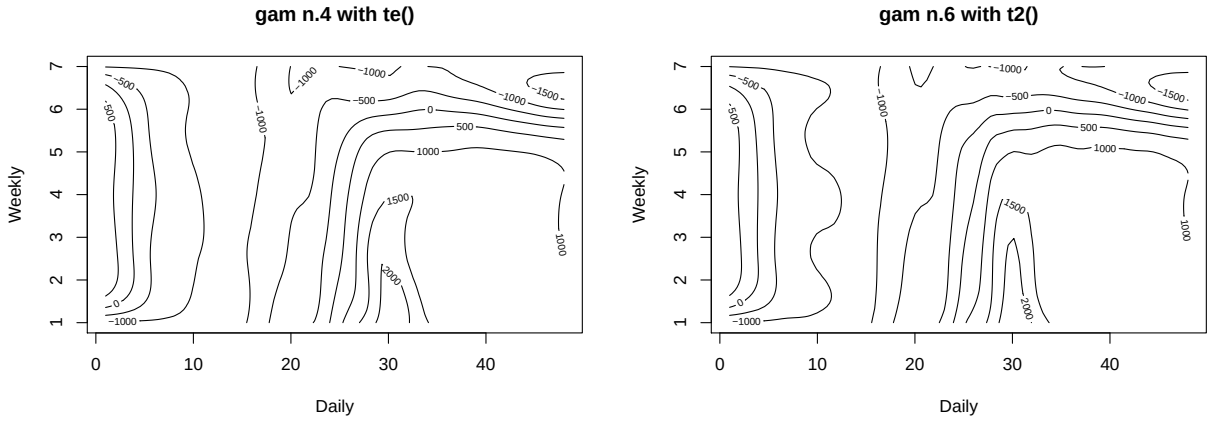
```
AIC(gam_1, gam_2, gam_3, gam_4, gam_5, gam_6)
```

```
##              df       AIC
## gam_1  17.46996 10248.993
## gam_2  30.70080  9415.768
## gam_3  25.65709  9492.545
## gam_4 121.41166  8912.611
## gam_5 115.80849  8932.746
## gam_6 100.12005  8868.628
```

很明顯 gam_4, gam_5, gam_6 是優勝組，其中 gam_6 是最好的那一個，緊隨其後的是 gam_4

接下來就 gam_4, gam_6 這幾個模型單獨畫圖看看結果是什麼樣的吧

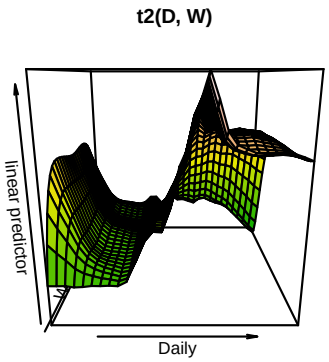
```
layout(matrix(1:2, nrow = 1))
plot(gam_4, rug = FALSE, se = FALSE, n2 = 80, main = "gam n.4 with te()")
plot(gam_6, rug = FALSE, se = FALSE, n2 = 80, main = "gam n.6 with t2()")
```



這些類似等高線的圖像顯示了各個模型對 `Weekly` 和 `Daily` 的反應情況。`gam_4`, `gam_6` 類似，但 `gam_6` 的圖形有更多的波動性，說明它的靈敏程度更好。

最後在這個章節結束前，看看怎麼把 `gam_6` 的圖像畫得更好看吧。首先使用 `mgcv` 套件包裡的 `vis.gam` 功能

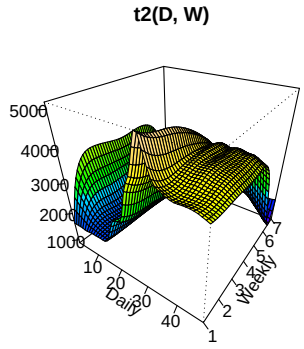
```
#    vis.gam(gam_6, main = "t2(D, W)", plot.type = "contour",
#            color = "terrain", contour.col = "black", lwd = 2)
vis.gam(gam_6, main = "t2(D, W)",
        color = "terrain", contour.col = "black", lwd = 2)
```



可以看出，平日的用電量比週末來得多很多，每天的工作時間用電量比較大，禮拜一到禮拜四的下午 3 點左右是用電最高峰時期。

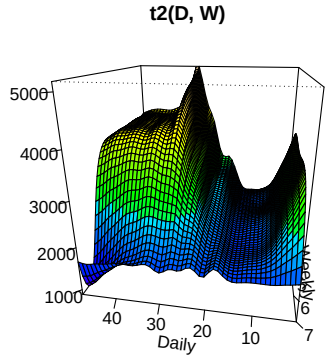
加入不使用 `contour.col` 選項，還可以作一個 3D 版本的圖

```
vis.gam(gam_6, n.grid = 50, theta = 35, phi = 32, zlab = "",
        ticktype = "detailed", color = "topo", main = "t2(D, W)")
```



轉換一下視角

```
vis.gam(gam_6, n.grid = 50, theta = 190, phi = 20, zlab = "",
        ticktype = "detailed", color = "topo", main = "t2(D, W)")
```



3. 分析解釋變量的重要情況

現在來看看減少剔除部分解釋變數對模型的有什麼樣的影響

```
gam_6D <- gam(Load ~ t2(Daily,
                        k = period,
                        bs = "cr",
                        full = TRUE),
              data = matrix_gam,
              family = gaussian)

summary(gam_6D)$r.sq
```

```
## [1] 0.5129567
```

```
summary(gam_6D)$sp.criterion
```

```
##   GCV.Cp
## 518169.2
```

嗯，明顯效果變差了

這回剔除 Daily 這個因素

```
gam_6W <- gam(Load ~ t2(Weekly,
                        k = 7,
                        bs =  "ps",
                        full = TRUE),
              data = matrix_gam,
              family = gaussian)

summary(gam_6W)$r.sq
```

```
## [1] 0.2502016
```

```
summary(gam_6W)$sp.criterion
```

```
##   GCV.Cp
## 791454.6
```

可以看到這下效果更糟糕了

保持著嚴謹的態度，使用 anova 來比較這三個模型的差異

先看看剔除 Weekly 會造成怎麼樣的差異

```
anova(gam_6, gam_6D, test="F")
```

```
## Analysis of Deviance Table
##
## Model 1: Load ~ t2(Daily, Weekly, k = c(period, 7), bs = c("cr", "ps"),
##     full = TRUE)
## Model 2: Load ~ t2(Daily, k = period, bs = "cr", full = TRUE)
##   Resid. Df Resid. Dev      Df   Deviance      F    Pr(>F)
## 1    550.77   15740821
## 2    661.13  339050831 -110.37 -323310010 106.61 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

再看看剔除 Daily 會造成怎麼樣的差異

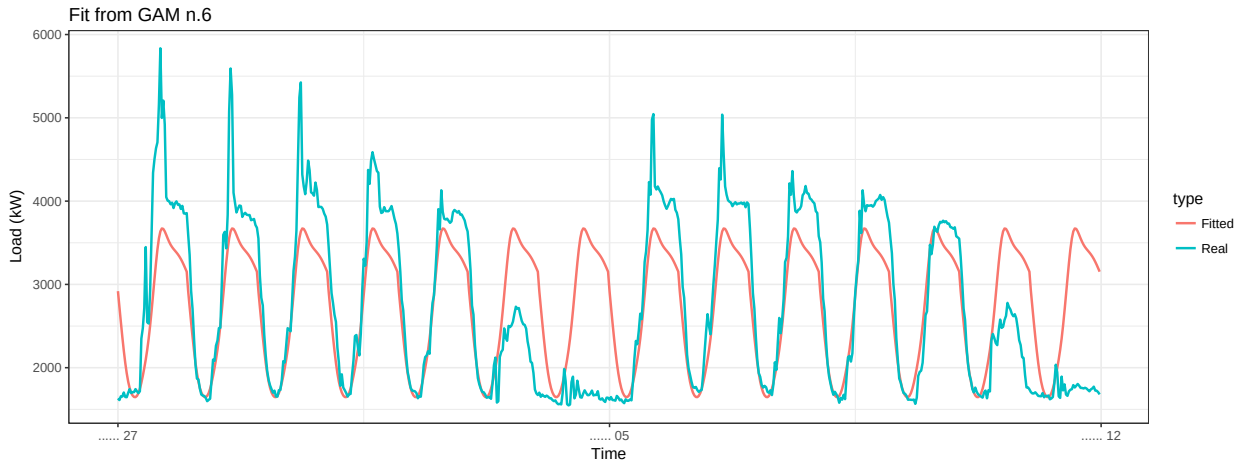
```
anova(gam_6, gam_6W, test="F")
```

```
## Analysis of Deviance Table
##
## Model 1: Load ~ t2(Daily, Weekly, k = c(period, 7), bs = c("cr", "ps"),
##     full = TRUE)
## Model 2: Load ~ t2(Weekly, k = 7, bs = "ps", full = TRUE)
##   Resid. Df Resid. Dev      Df   Deviance     F    Pr(>F)
## 1    550.77   15740821
## 2    667.88  526095056 -117.11 -510354235 158.6 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

結果顯而易見，Weekly 因素和 Daily 因素缺一不可！

作圖展示不考慮 Weekly 因素的模型的擬合效果：

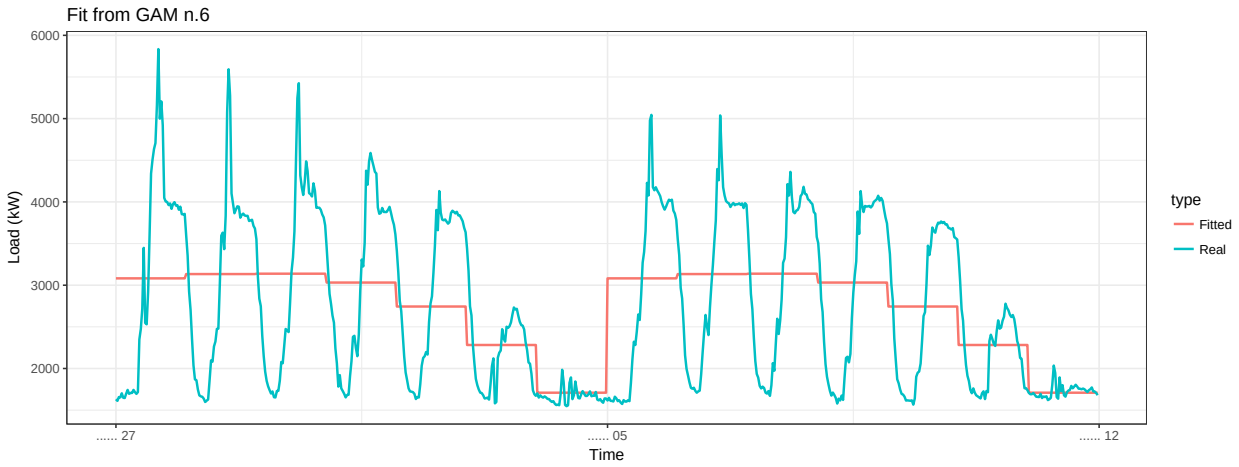
```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_6D$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.6")
```



可以看出沒有考慮 Weekly 因素的話，預測結果不會體現週次不同所造成的用電起伏變化

作圖展示不考慮 Daily 因素的模型的擬合效果：

```
datas <- rbindlist(list(data_r[, .(value, date_time)],
                        data.table(value = gam_6W$fitted.values,
                                   data_time = data_r[, date_time])))
datas[, type := c(rep("Real", nrow(data_r)), rep("Fitted", nrow(data_r)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Fit from GAM n.6")
```



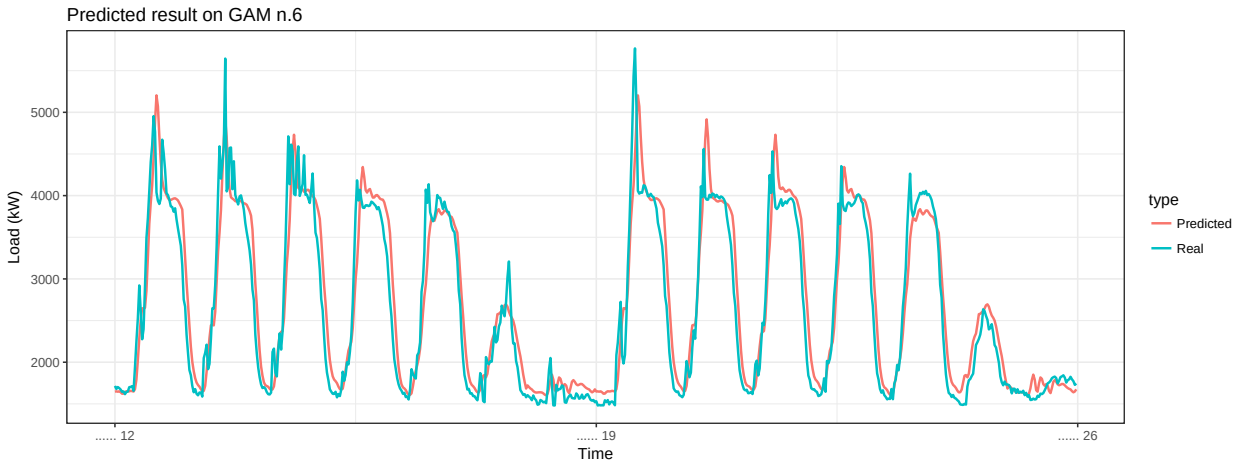
可以看出沒有考慮 Daily 因素的話，預測結果不會體現一天 24 小時內的用電起伏變化

```
predWeek <- function(data, set_of_date){

  # Subsetting the dataset by dates
  data_train <- data[date %in% set_of_date]

  N <- nrow(data_train)
  window <- N / period # number of days in the train set
  # 1, ..., period, 1, ..., period - and so on for the daily season
  # Using feature "week_num" for the weekly season
  matrix_train <- data.table(Load = data_train[, value],
                             Daily = as.factor(rep(1:period, window)),
                             Weekly = as.factor(data_train[, week_num]))

  # Creation of the model
  lm_m <- lm(Load ~ 0 + Daily + Weekly + Daily:Weekly, data = matrix_train)

  # Creation of the forecast for one week ahead
  pred_week <- predict(lm_m, matrix_train[1:(7*period), -1, with = FALSE])

  return(as.vector(pred_week))
}
```

4. 預測電耗

最後，最振奮人心的——預測接下來兩周的用電量

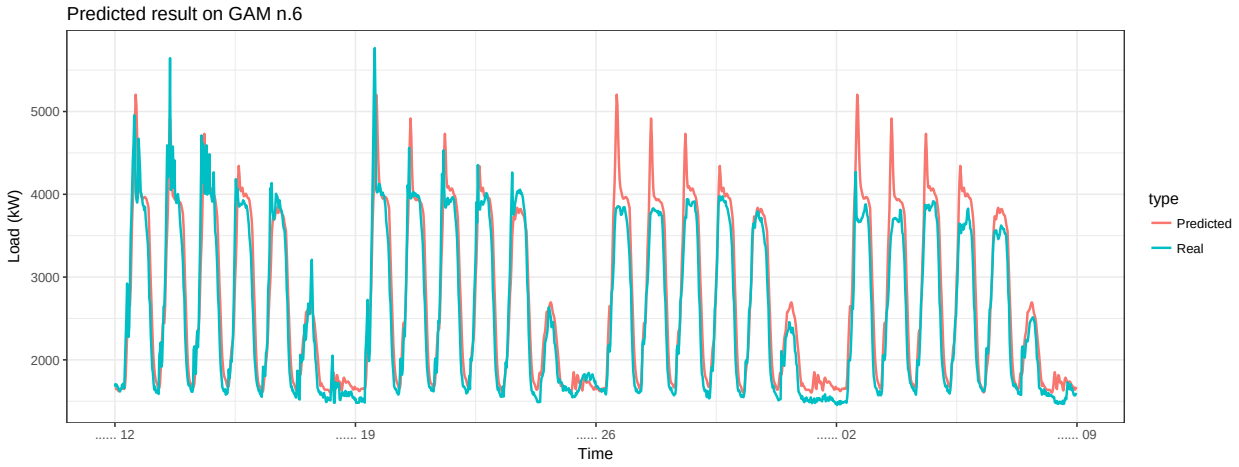
```
data_test <- DT[(type == n_type[1] & date %in% n_date[71:84])]
matrix_test <- data.table(Load = data_test[, value],
                          Daily = rep(1:period, window),
                          Weekly = data_test[, week_num])
pred_week <- predict(gam_6, matrix_test[1:(7*period)],interval="confidence", level = 0.95)

datat <- rbindlist(list(data_test[, .(value, date_time)],
                        data.table(value = pred_week,
                                   data_time = data_test[, date_time])))
datat[, type := c(rep("Real", nrow(data_test)), rep("Predicted", nrow(data_test)))]
ggplot(data = datat, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Predicted result on GAM n.6")
```



預測接下來一個月的用電量

```
data_test <- DT[(type == n_type[1] & date %in% n_date[71:98])]
matrix_test <- data.table(Load = data_test[, value],
                          Daily = rep(1:period, window),
                          Weekly = data_test[, week_num])
pred_week <- predict(gam_6, matrix_test[1:(7*period)],interval="confidence", level = 0.95)
```

```
datat <- rbindlist(list(data_test[, .(value, date_time)],
                        data.table(value = pred_week,
                                   data_time = data_test[, date_time])))
datat[, type := c(rep("Real", nrow(data_test)), rep("Predicted", nrow(data_test)))]
ggplot(data = datat, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Predicted result on GAM n.6")
```



期末結束新增的部分：

定義 MAPE 函數

```
mape <- function(real, pred){
  return(100 * mean(abs((real - pred)/real)))
}
```

定義各個模型的評估標準（R-sq, GCV, MAPE):

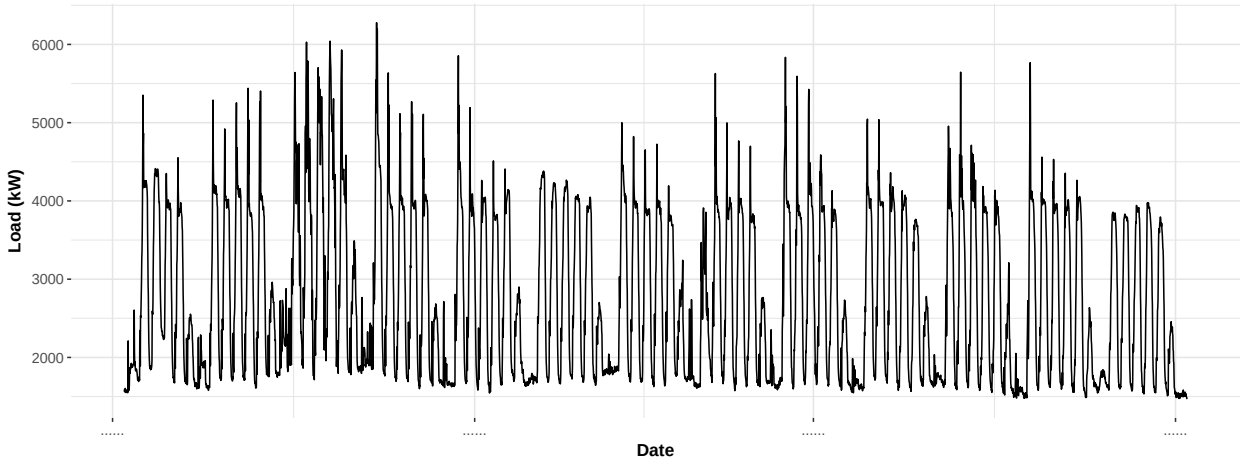
```
gam_eval <- function(model){
    return(data.table(RSQ=summary(model)$r.sq,
                      GCV=summary(model)$sp.criterion,
                      MAPE=mape(data_new[, value], model$fitted.values)))
}
```

定義各個模型的圖表分析函數

```
gam_plot <- function(model, title){
    datas <- rbindlist(list(data_new[, .(value, date_time)],
                        data.table(value = model$fitted.values,
                                   data_time = data_new[, date_time])))
datas[, type := c(rep("Real", nrow(data_new)), rep("Fitted", nrow(data_new)))]

ggplot(data = datas, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = title)
}
```

利用第一個季度的資料構建訓練集

```
data_new <- DT[(type == n_type[1] & date %in% n_date[1:91])]

ggplot(data_new, aes(date_time, value)) +
  geom_line() +
  theme(panel.border = element_blank(),
        panel.background = element_blank(),
        panel.grid.minor = element_line(colour = "grey90"),
        panel.grid.major = element_line(colour = "grey90"),
        panel.grid.major.x = element_line(colour = "grey90"),
        axis.text = element_text(size = 10),
        axis.title = element_text(size = 12, face = "bold")) +
  labs(x = "Date", y = "Load (kW)")
```



重新整理資料，新增前一天同時間的電耗和前一禮拜同時間的電耗

```
matrix_new <- data.table(Load = data_new[, value],
                         PrevDayLoad = c(data_new[1:48, value], data_new[1:4320, value]),
                         PrevWeekLoad = c(data_new[1:336, value], data_new[1:4032, value]),
                         Daily = rep(1:period, window),
                         Weekly = data_r[, week_num])
```

```
## Warning in data.table(Load = data_new[, value], PrevDayLoad =
## c(data_new[1:48, : Item 4 is of size 672 but maximum size is 4368 (recycled
## leaving remainder of 336 items)
```

```
## Warning in data.table(Load = data_new[, value], PrevDayLoad =
## c(data_new[1:48, : Item 5 is of size 672 but maximum size is 4368 (recycled
## leaving remainder of 336 items)
```

建立第一個模型

```
gam_new_1 <- gam(Load ~ s(Daily, bs = "cr", k = period) +
                   s(Weekly, bs = "ps", k = 7),
                 data = matrix_new,
                 family = gaussian)
```

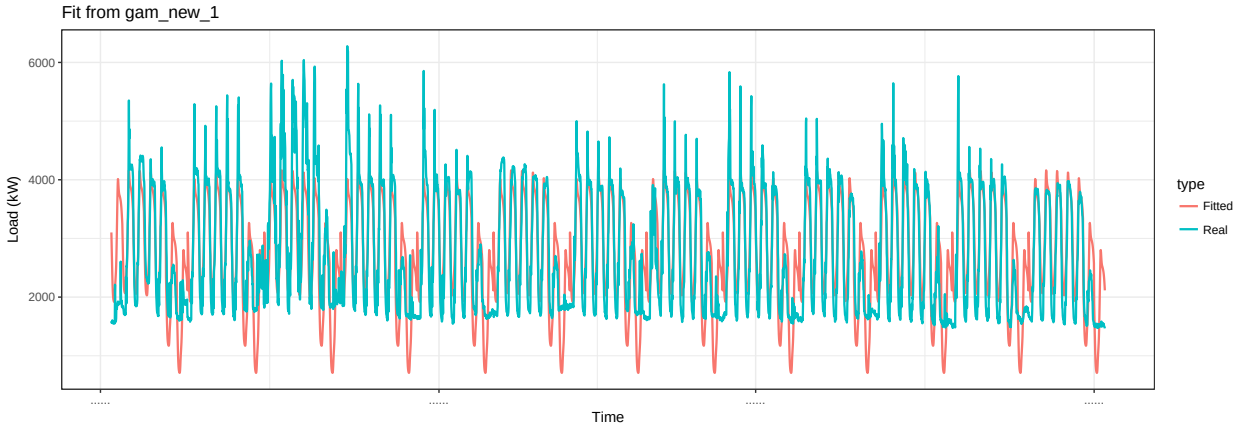
第一個模型的數據分析

```
eval_1 <- gam_eval(gam_new_1)
eval_1
```

```
##          RSQ      GCV     MAPE
## 1: 0.7194149 311287.5 18.89976
```

第一個模型的擬合狀況圖像

```
gam_plot(gam_new_1, "Fit from gam_new_1")
```



建立第二個模型

```
gam_new_2 <- gam(Load ~ s(Daily, Weekly),
                 data = matrix_new,
                 family = gaussian)
```

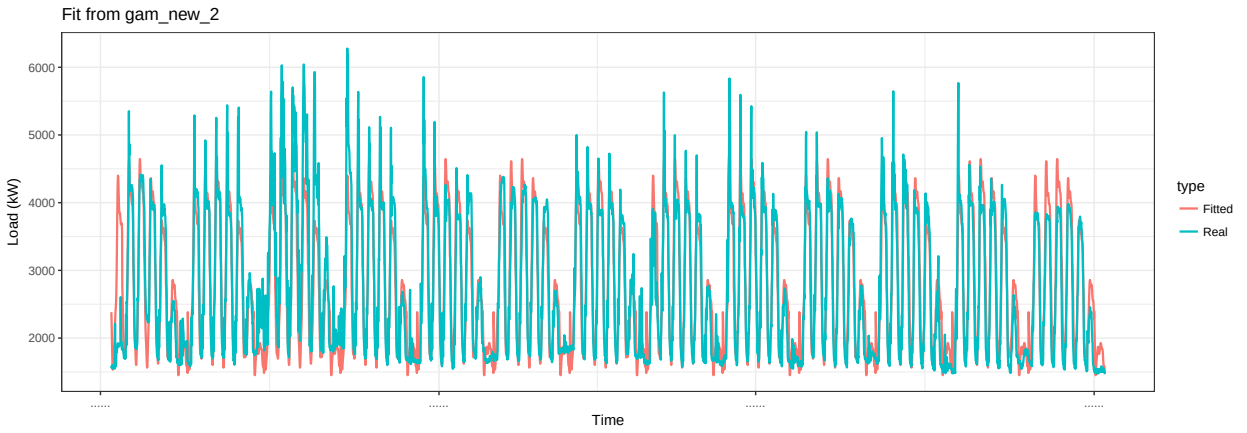
第二個模型的數據分析

```
eval_2 <- gam_eval(gam_new_2)
eval_2
```

```
##          RSQ      GCV     MAPE
## 1: 0.8487935 168039.3 10.62365
```

第二個模型的擬合狀況圖像

```
gam_plot(gam_new_2, "Fit from gam_new_2")
```



建立第三個模型

```
gam_new_3 <- gam(Load ~ te(Daily, Weekly,
                           bs = c("cr", "ps")),
                 data = matrix_new,
                 family = gaussian)
```

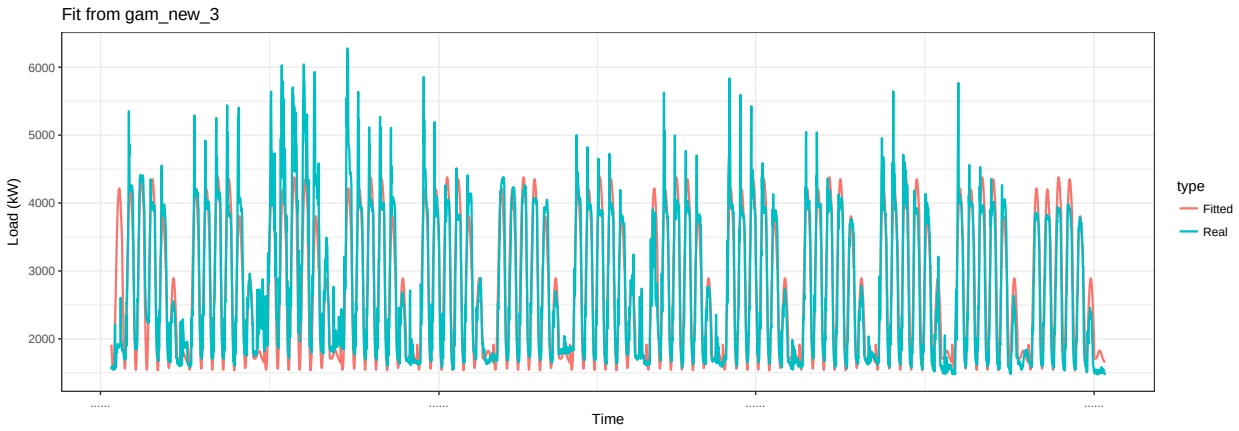
第三個模型的數據分析

```
eval_3 <- gam_eval(gam_new_3)
eval_3
```

```
##          RSQ      GCV     MAPE
## 1: 0.8423209 175028.3 10.18985
```

第三個模型的擬合狀況圖像

```
gam_plot(gam_new_3, "Fit from gam_new_3")
```



建立第四個模型

```
gam_new_4 <- gam(Load ~ te(Daily, Weekly,
                           k = c(period, 7),
                           bs = c("cr", "ps")),
                 data = matrix_new,
                 family = gaussian)
```

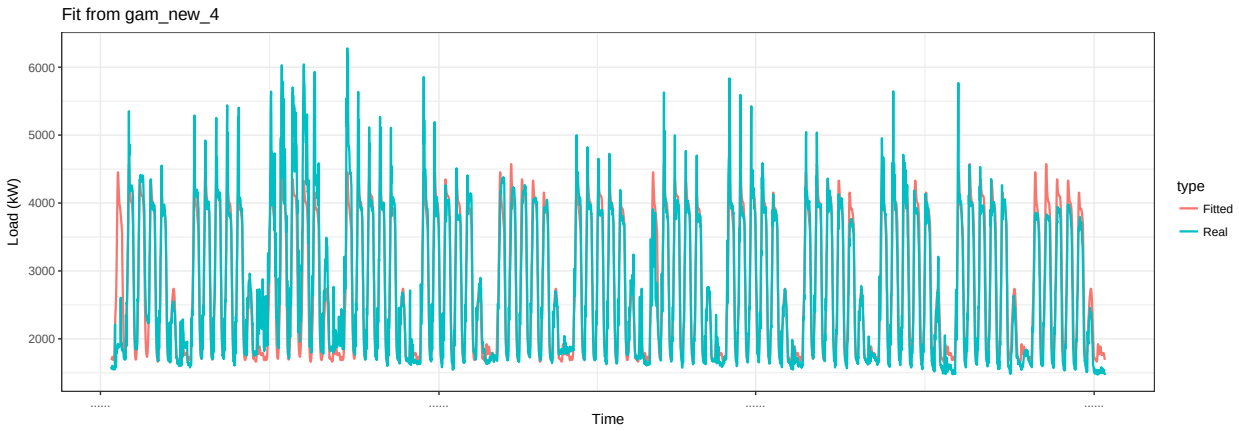
第四個模型的數據分析

```
eval_4 <- gam_eval(gam_new_4)
eval_4
```

```
##          RSQ      GCV     MAPE
## 1: 0.8796766 135799.2 8.319575
```

第四個模型的擬合狀況圖像

```
gam_plot(gam_new_4, "Fit from gam_new_4")
```



建立第五個模型

```
gam_new_5 <- gam(Load ~ s(Daily, bs = "cr", k = period) +
                    s(Weekly, bs = "ps", k = 7) +
                    ti(Daily, Weekly,
                       k = c(period, 7),
                       bs = c("cr", "ps")),
             data = matrix_new,
             family = gaussian)
```

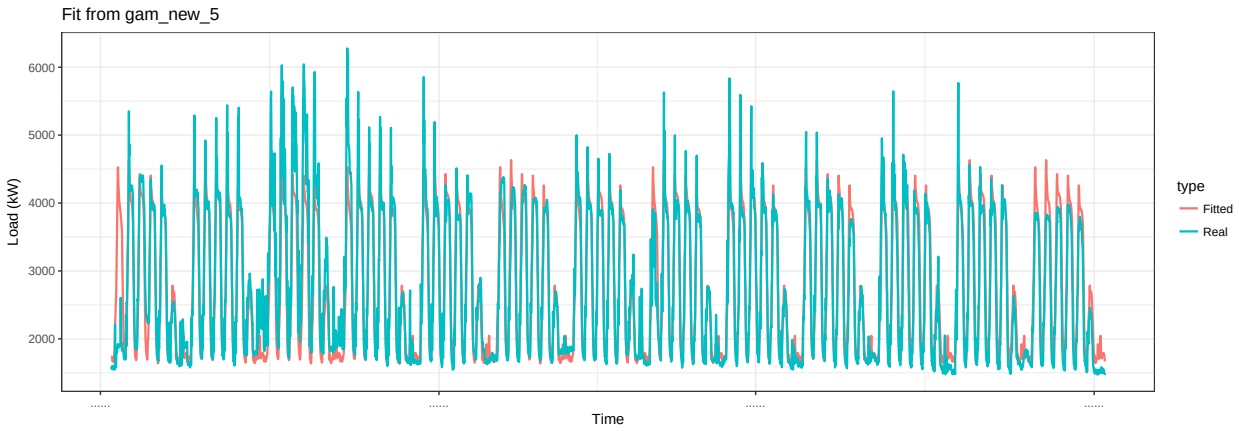
第五個模型的數據分析

```
eval_5 <- gam_eval(gam_new_5)
eval_5
```

```
##          RSQ    GCV     MAPE
## 1: 0.8801601 135306 8.337419
```

第五個模型的擬合狀況圖像

```
gam_plot(gam_new_5, "Fit from gam_new_5")
```



建立第六個模型

```
gam_new_6 <- gam(Load ~ t2(Daily, Weekly,
                           k = c(period, 7),
                           bs = c("cr", "ps"),
                           full = TRUE),
                 data = matrix_new,
                 family = gaussian)
```

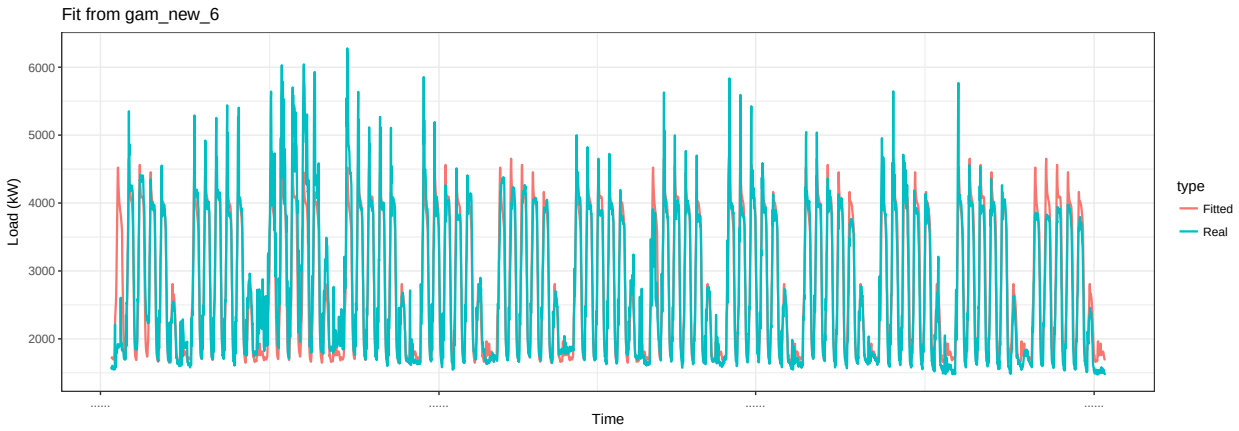
第六個模型的數據分析

```
eval_6 <- gam_eval(gam_new_6)
eval_6
```

```
##          RSQ      GCV    MAPE
## 1: 0.8802681 134768.2 8.32655
```

第六個模型的擬合狀況圖像

```
gam_plot(gam_new_6, "Fit from gam_new_6")
```



構建第七個模型——不使用 **smooth function** 的 **gam** 模型：

```
gam_new_simple <- gam(Load ~ Daily+Weekly,
                      data = matrix_new,
                      family = gaussian)
```

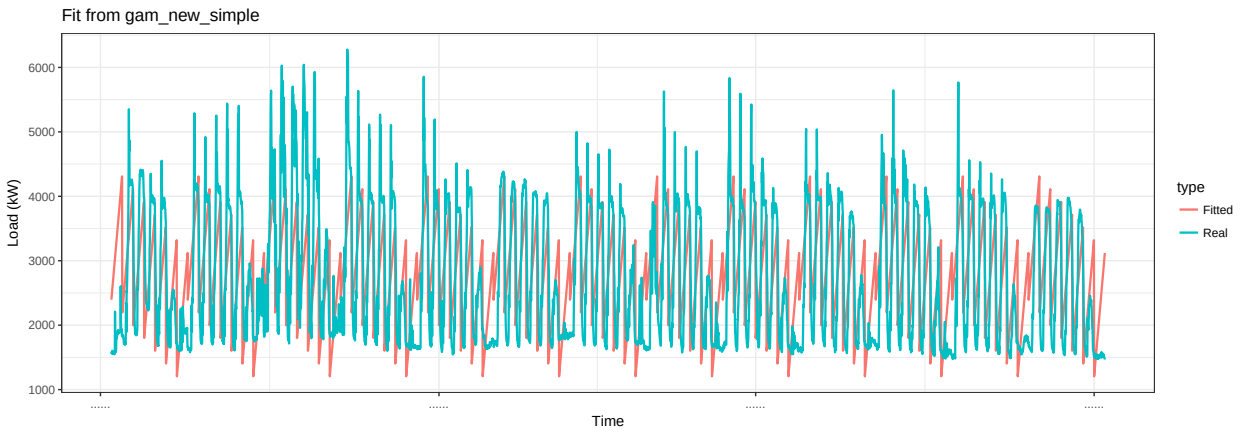
第七個模型的數據分析

```
eval_7 <- gam_eval(gam_new_simple)
eval_7
```

```
##          RSQ      GCV     MAPE
## 1: 0.4302044 629325.7 25.61359
```

第七個模型的擬合狀況圖像

```
gam_plot(gam_new_simple, "Fit from gam_new_simple")
```



建立第八個模型——只對 Daily 作 te smooth function

```
gam_new_te_daily <- gam(Load ~ te(Daily, bs = "cr", k = period) +Weekly,
             data = matrix_new,
             family = gaussian)
```

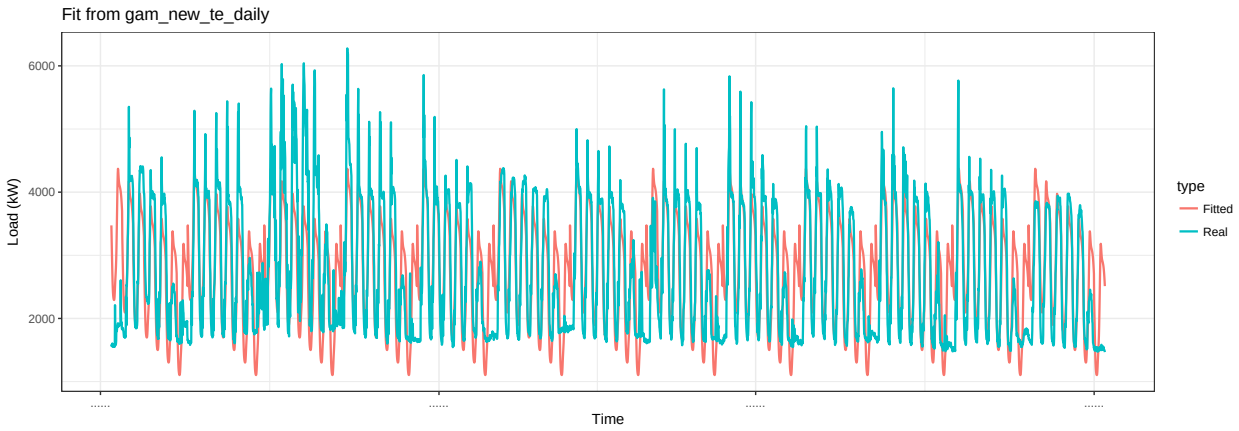
第八個模型的數據分析

```
eval_8 <- gam_eval(gam_new_te_daily)
eval_8
```

```
##          RSQ      GCV     MAPE
## 1: 0.6366094 402558.4 20.44199
```

第八個模型的擬合狀況圖像

```
gam_plot(gam_new_te_daily, "Fit from gam_new_te_daily")
```



建立第九個模型——只對 Weekly 作 te smooth function

```
gam_new_te_weekly <- gam(Load ~ te(Weekly, bs = "ps", k = 7) +Daily,
             data = matrix_new,
             family = gaussian)
```

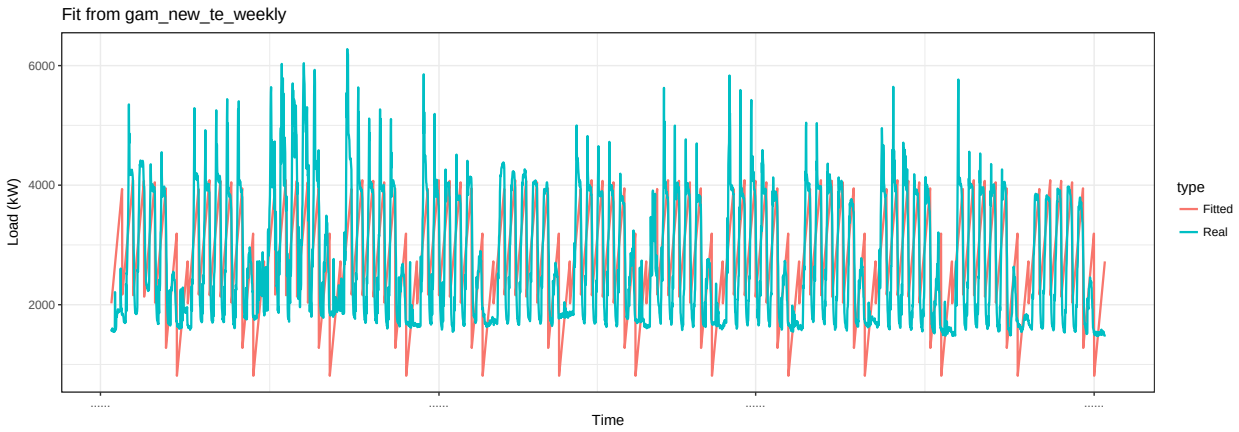
第九個模型的數據分析

```
eval_9 <- gam_eval(gam_new_te_weekly)
eval_9
```

```
##          RSQ    GCV   MAPE
## 1: 0.5124041 539142 24.279
```

第九模型的擬合狀況圖像

```
gam_plot(gam_new_te_weekly, "Fit from gam_new_te_weekly")
```



用表格查看 9 個模型的數據分析

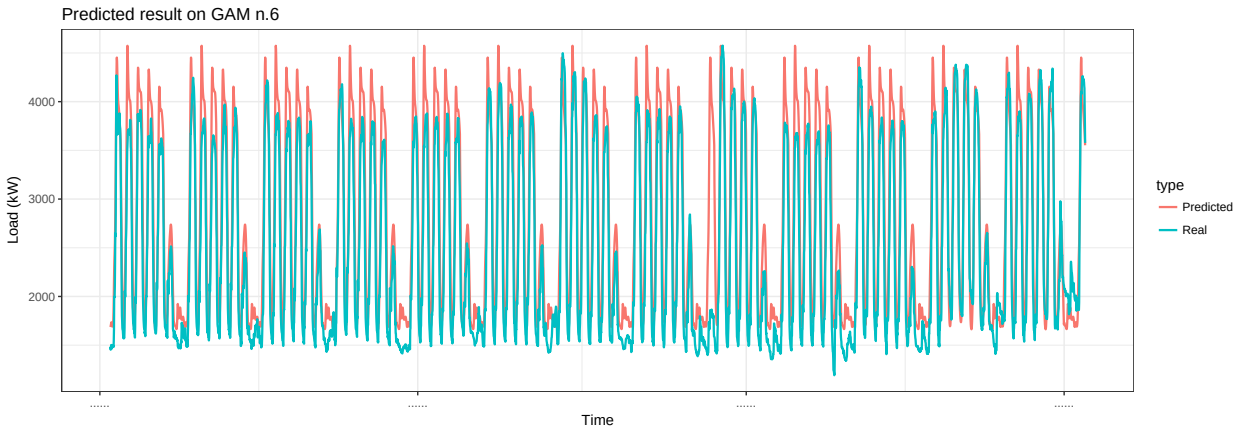
```
eval_table <- bind_rows(eval_1, eval_2, eval_3, eval_4, eval_5, eval_6, eval_7, eval_8, eval_9)
all_aic <- AIC(gam_new_1, gam_new_2, gam_new_3, gam_new_4, gam_new_5, gam_new_6, gam_new_simple, g

eval_table[, AIC :=all_aic]
eval_table <- data.table(MODELS=c("gam_new_1 ", "gam_new_2", "gam_new_3", "gam_new_4", "gam_new_5"
eval_table
```

```
##               MODELS       RSQ      GCV      MAPE      AIC
## 1:        gam_new_1  0.7194149 311287.5 18.899761 67646.26
## 2:         gam_new_2 0.8487935 168039.3 10.623654 64953.21
## 3:         gam_new_3 0.8423209 175028.3 10.189855 65131.27
## 4:         gam_new_4 0.8796766 135799.2  8.319575 64020.79
## 5:         gam_new_5 0.8801601 135306.0  8.337419 64004.82
## 6:         gam_new_6 0.8802681 134768.2  8.326550 63987.99
## 7:    gam_new_simple 0.4302044 629325.7 25.613586 70721.15
## 8:  gam_new_te_daily 0.6366094 402558.4 20.441993 68769.43
## 9: gam_new_te_weekly 0.5124041 539142.0 24.278996 70045.54
```

使用第四個 model 預測當年第二個季度

```
data_test_2qt <- DT[(type == n_type[1] & date %in% n_date[92:183])]
matrix_test_2qt <- data.table(Load = data_test_2qt[, value],
```

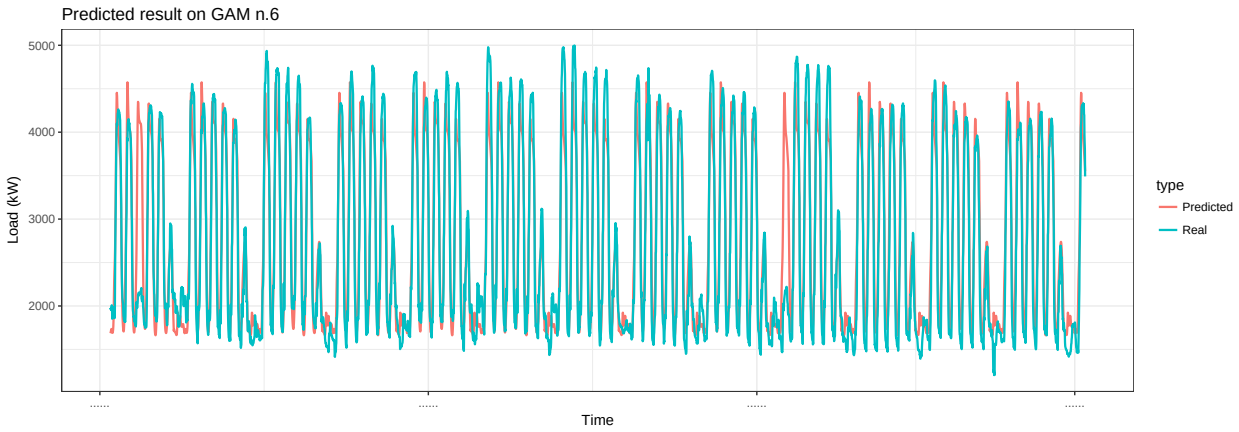
```
                            Daily = rep(1:period, 91),
                            Weekly = data_test_2qt[, week_num])
pred_2qt <- predict(gam_new_4, matrix_test_2qt[1:(7*period)],interval="confidence", level = 0.95)

datat <- rbindlist(list(data_test_2qt[, .(value, date_time)],
                        data.table(value = pred_2qt,
                                   data_time = data_test_2qt[, date_time])))
datat[, type := c(rep("Real", nrow(data_test_2qt)), rep("Predicted", nrow(data_test_2qt)))]
ggplot(data = datat, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Predicted result on GAM n.6")
```



使用第四個 model 預測當年第三個季度

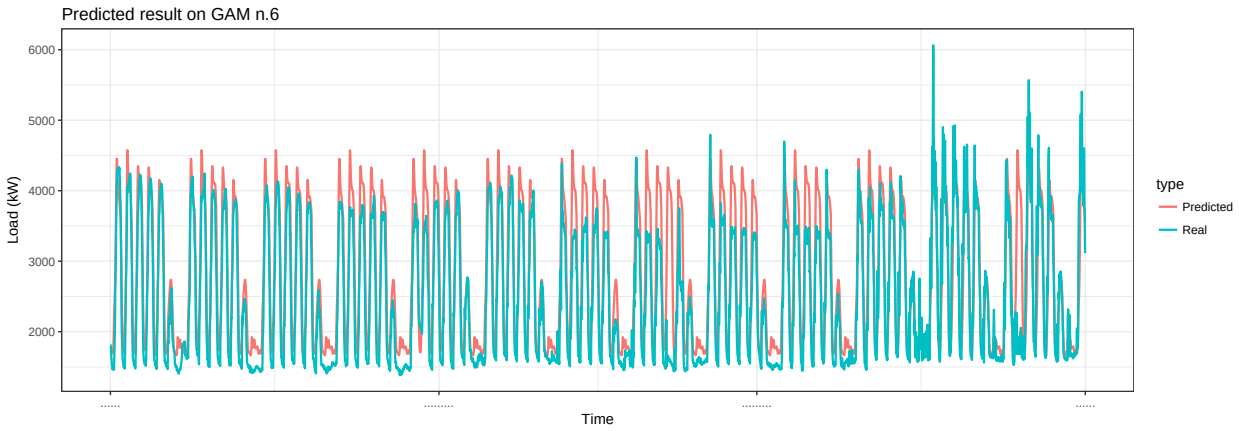
```
data_test_3qt <- DT[(type == n_type[1] & date %in% n_date[183:274])]
matrix_test_3qt <- data.table(Load = data_test_3qt[, value],
                            Daily = rep(1:period, 91),
                            Weekly = data_test_3qt[, week_num])
pred_3qt <- predict(gam_new_4, matrix_test_3qt[1:(7*period)],interval="confidence", level = 0.95)

datat <- rbindlist(list(data_test_3qt[, .(value, date_time)],
                        data.table(value = pred_3qt,
                                   data_time = data_test_3qt[, date_time])))
datat[, type := c(rep("Real", nrow(data_test_3qt)), rep("Predicted", nrow(data_test_3qt)))]
ggplot(data = datat, aes(date_time, value, group = type, colour = type)) +
```

```
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Predicted result on GAM n.6")
```



使用第四個 model 預測當年第四個季度

```
data_test_4qt <- DT[(type == n_type[1] & date %in% n_date[274:365])]
matrix_test_4qt <- data.table(Load = data_test_4qt[, value],
                              Daily = rep(1:period, 91),
                              Weekly = data_test_4qt[, week_num])
pred_4qt <- predict(gam_new_4, matrix_test_4qt[1:(7*period)],interval="confidence", level = 0.95)

datat <- rbindlist(list(data_test_4qt[, .(value, date_time)],
                        data.table(value = pred_4qt,
                                   data_time = data_test_4qt[, date_time])))
datat[, type := c(rep("Real", nrow(data_test_4qt)), rep("Predicted", nrow(data_test_4qt)))]
ggplot(data = datat, aes(date_time, value, group = type, colour = type)) +
  geom_line(size = 0.8) +
  theme_bw() +
  labs(x = "Time", y = "Load (kW)",
       title = "Predicted result on GAM n.6")
```



計算 prediciton 的 MAPE

```
mape_2qt <- mape(matrix_test_2qt[1:(7*period)]$Load, pred_2qt)
mape_3qt <- mape(matrix_test_3qt[1:(7*period)]$Load, pred_3qt)
mape_4qt <- mape(matrix_test_4qt[1:(7*period)]$Load, pred_4qt)
mapes <- cbind(mape_2qt, mape_3qt, mape_4qt)
mapes
```

```
##      mape_2qt mape_3qt mape_4qt
## [1,] 12.01464 14.89136 12.93795
```