05_02_Intra_day_model5_2018_2022

Maggie

2023-05-02

```
source("Functions.R")
library(mgcv)
```

```
##      nlme
```

```
## This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

```
library(plotly)
```

```
## Warning:   'plotly' R 4.2.3
```

```
##      ggplot2
```

```
## Warning:   'ggplot2' R 4.2.3
```

```
##
##    'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

1. Reconstruct Model 5 using original 2014 data: 0.89429

```
## Load the data
alldata <- read.csv("original2014.csv")
alldata <- alldata[,c("Demand", "Temp", "DSTTime", "Year")]
head(alldata)
```

```
##    Demand Temp    DSTTime     Year
## 1 7135.67 22.4 0.04166667 0.00e+00
## 2 7154.87 22.5 0.04513889 9.51e-06
## 3 7086.94 22.4 0.04861111 1.90e-05
## 4 7042.09 22.5 0.05208333 2.85e-05
## 5 6942.04 22.6 0.05555556 3.81e-05
## 6 7017.70 22.6 0.05902778 4.76e-05
```

```
alldata$Temp <- round(alldata$Temp)
alldata$WtdTemp <- wtdtemp(alldata$DSTTime, alldata$Temp)
```

```
## Split into regression data and out of sample test data.
#fitdata <- alldata[((0*288)+1):(250*288),]
fitdata <- alldata

head(fitdata)
```

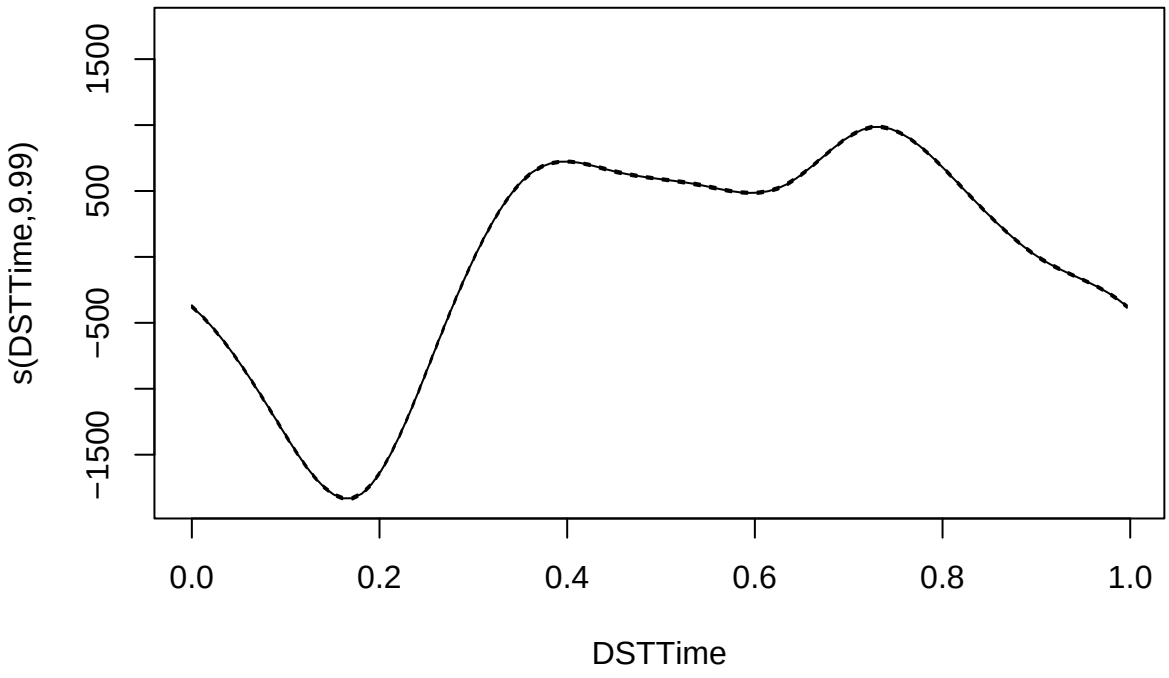
```
##    Demand Temp    DSTTime     Year  WtdTemp
## 1 7135.67   22 0.04166667 0.00e+00 20.58519
## 2 7154.87   22 0.04513889 9.51e-06 20.57619
## 3 7086.94   22 0.04861111 1.90e-05 20.56731
## 4 7042.09   22 0.05208333 2.85e-05 20.55858
## 5 6942.04   23 0.05555556 3.81e-05 20.82500
## 6 7017.70   23 0.05902778 4.76e-05 20.81237
```

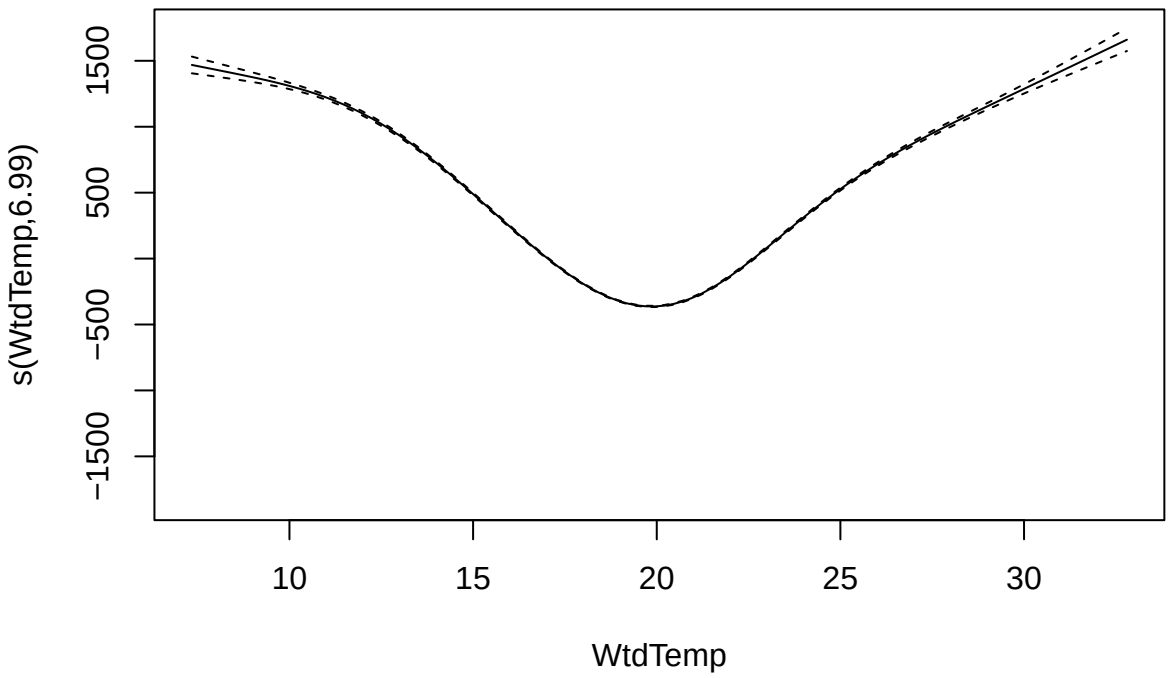
Fit Model 5 for 2014: R-square = 0.89429 VS 0.898 in Original Paper Original Paper:

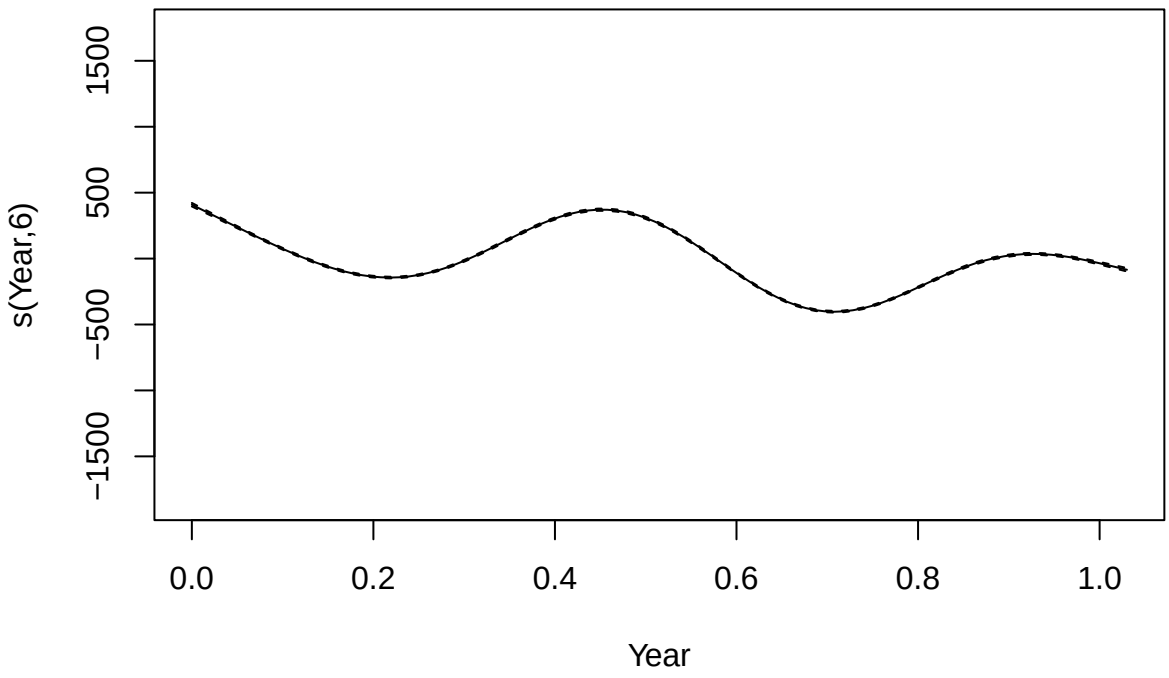
```
gamlwmod <- Demand ~  s(DSTTime, bs = "cc", k = 12) + s(WtdTemp, bs = "tp", k =8) + s(Year, bs = "tp", k
wtdyear <- gamm(gamlwmod,  data = fitdata)
summary(wtdyear$gam)$r.sq
```

```
## [1] 0.8942935
```

```
plot(wtdyear$gam, all.terms=T)
```




s(WtdTemp,6.99)
1500
500
−500
−1500
10
15
20
25
30
WtdTemp




```
perf <- data.frame(year=2018:2022, r.sq=0)
for (year in perf$year) {
  df <- read.csv(paste(year, ".csv", sep=""))
  mdl <- gamm(gamlwmod,  data = df)
  perf[perf$year==year, "r.sq"] <- summary(mdl$gam)$r.sq
}
perf
```

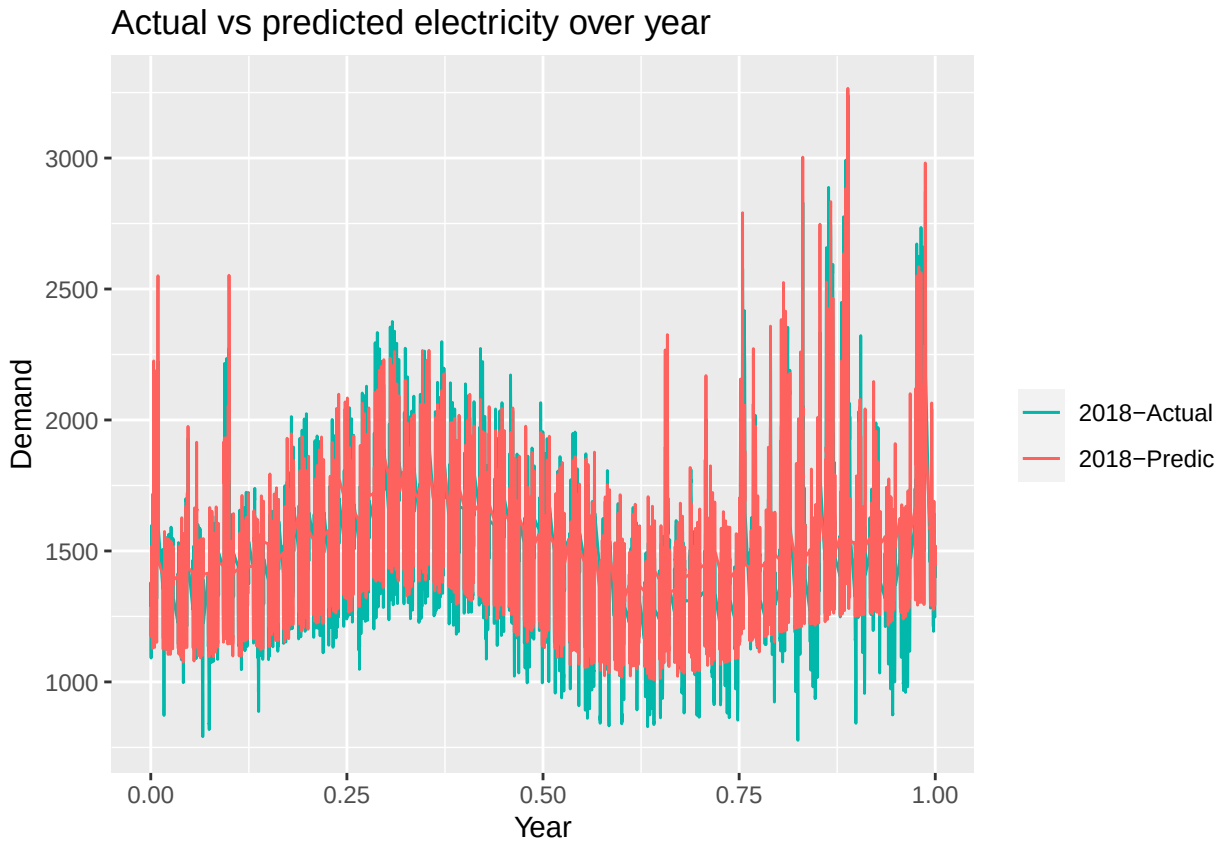
Fit model 5 for 2018-2022: lower R-square

```
##   year      r.sq
## 1 2018 0.7225950
## 2 2019 0.6934220
## 3 2020 0.7017409
## 4 2021 0.7138796
## 5 2022 0.7147552
```

2. Residual Analysis

```
data.2018 <- read.csv("2018.csv")
## Actual
# fig = plot_ly(data.2018, x = ~ Year, y= ~ Demand, type = "scatter", mode = "lines")

gamlwmod <- Demand ~  s(DSTTime, bs = "cc", k = 12) + s(WtdTemp, bs = "tp", k =8) + s(Year, bs = "tp", k
```

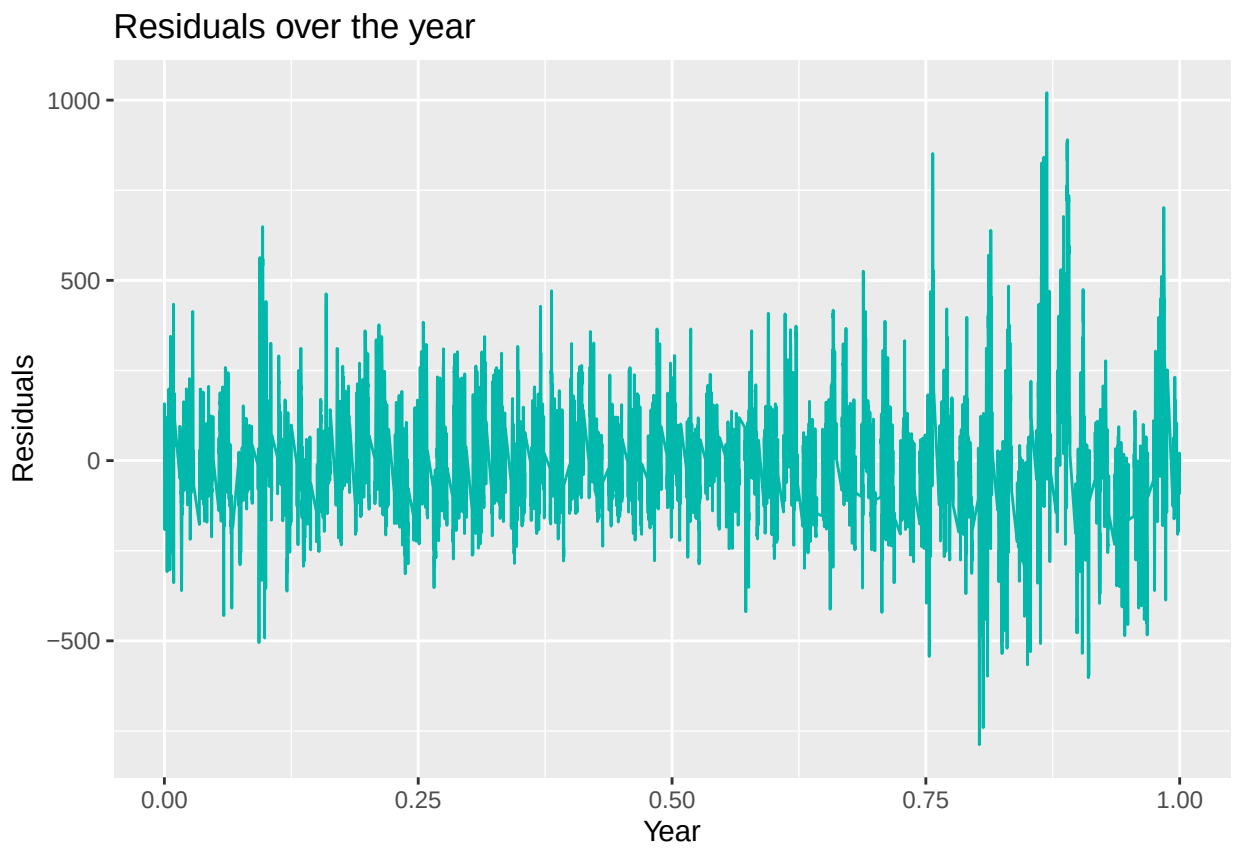
```r
wtdyear.2018 <- gamm(gamlwmod,  data = data.2018)
predicted.2018 = predict(wtdyear.2018$gam, newdata = data.2018)

## Compare
# plot_ly(x = data.2018$Year, y= data.2018$Demand,
#         type = "scatter",
#         mode = "lines",
#         name = "2018-Actual") %>% add_lines(data.2018$Year, predicted.2018, name = "2018-Predic") %>%

library(ggplot2)

df <- data.frame(
  Year = data.2018$Year,
  Demand = data.2018$Demand,
  predicted.2018 = predicted.2018
)

ggplot(df, aes(x = Year)) +
  geom_line(aes(y = Demand, color = "2018-Actual")) +
  geom_line(aes(y = predicted.2018, color = "2018-Predic")) +
  scale_color_manual(name = "", values = c("2018-Actual" = "#01b8aa", "2018-Predic" = "#FD625E")) +
  labs(x = "Year", y = "Demand")+
  ggtitle("Actual vs predicted electricity over year")
```



1) 2018

```r
## Residuals
residuals.2018 <- resid(wtdyear.2018$gam)
```

```r
# plot_ly(x = data.2018$Year, y = residuals.2018,type = "scatter", mode = "lines")%>% layout(xaxis = li

ggplot(data.frame(x = data.2018$Year, y = residuals.2018), aes(x = x, y = y)) +
  geom_line(col = "#01b8aa") +
  xlab("Year") +
  ylab("Residuals")+
  ggtitle("Residuals over the year")
```

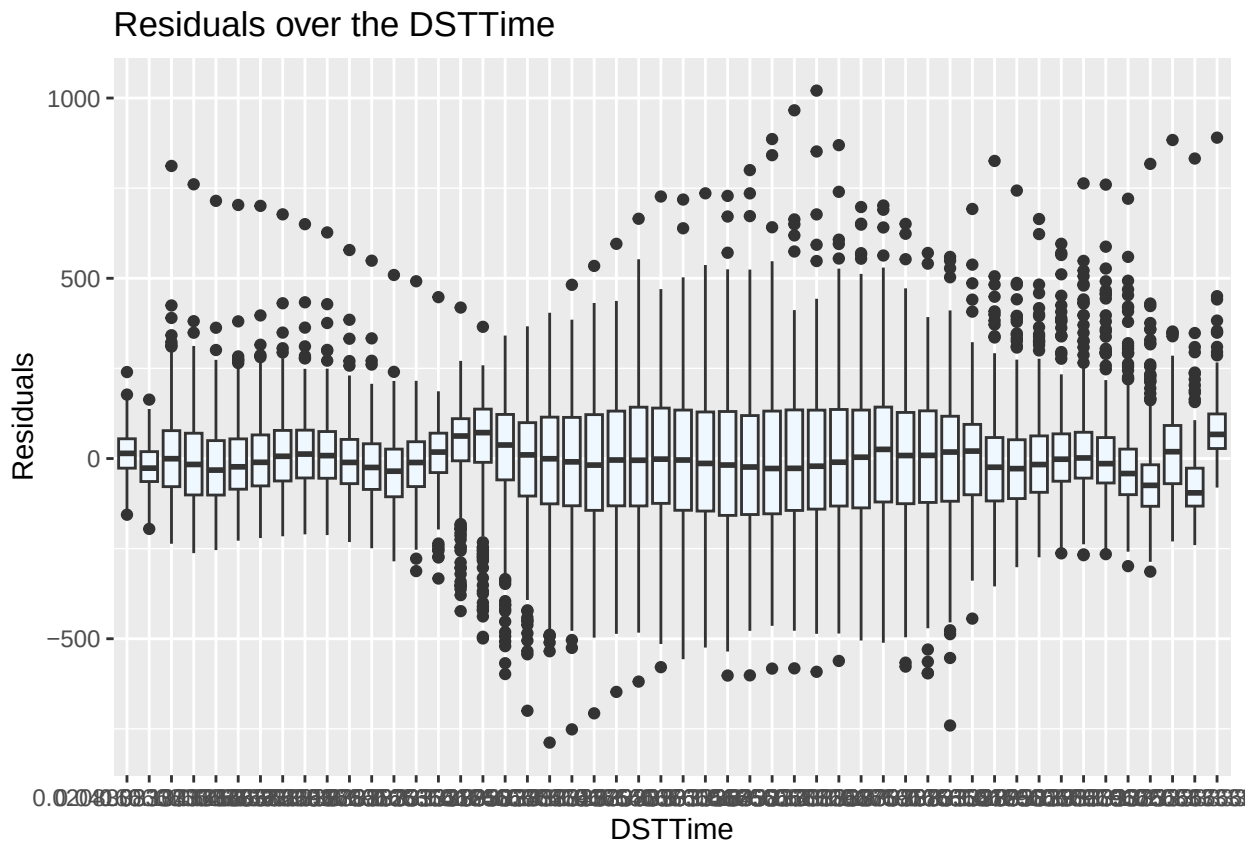



```r
#
```

Large residual along the year, with residual around +-500, but total demand around 1500, nearly 1/3 variation

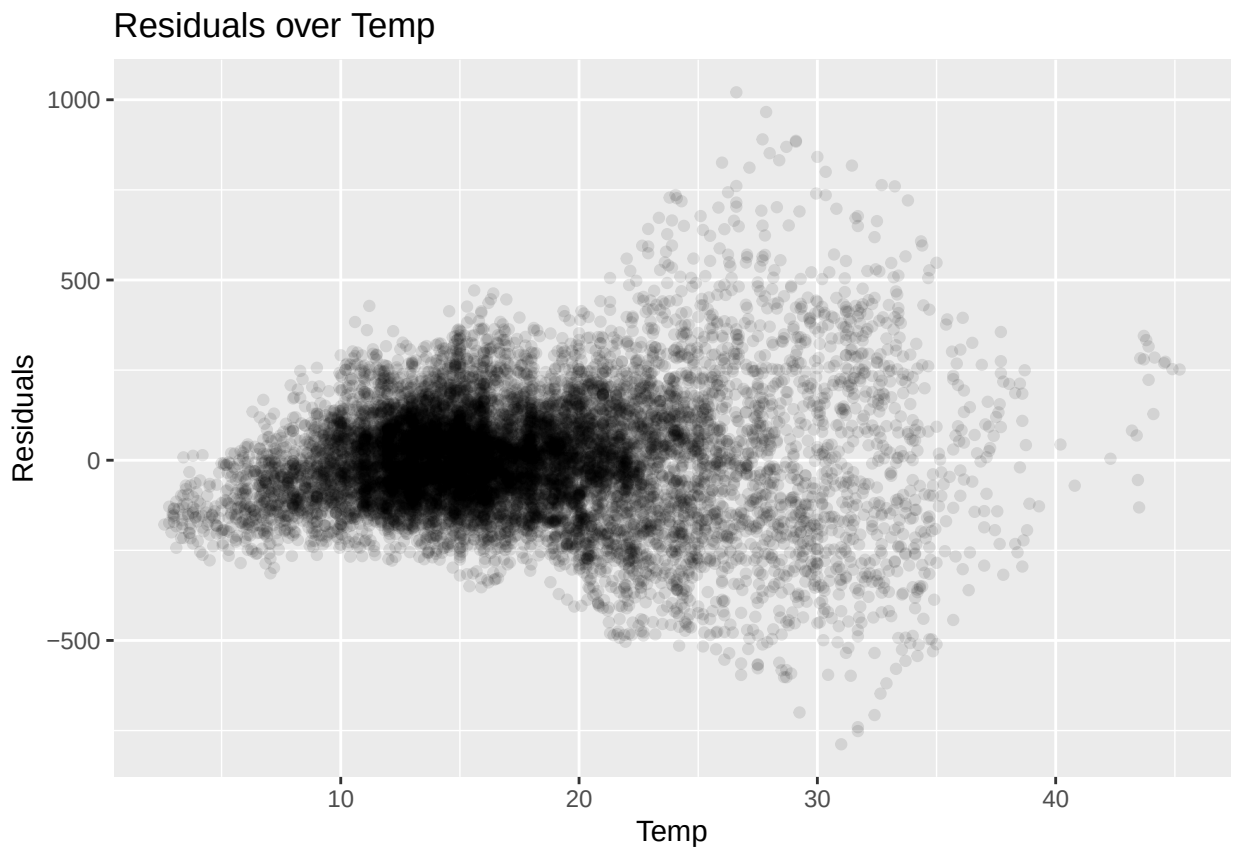
```r
# ## Residuals
# residuals.2018 <- resid(wtdyear.2018$gam)
#
# plot_ly(x = data.2018$DSTTime, y = residuals.2018,type = "box", mode = "lines")%>% layout(xaxis = lis

library(ggplot2)

# create a dataframe for plotting
df <- data.frame(DSTTime = data.2018$DSTTime, Residuals = residuals.2018)

# create a boxplot for each DSTTime
ggplot(df, aes(x = factor(DSTTime), y = Residuals)) +
  geom_boxplot(fill = "aliceblue") +
```

```
  xlab("DSTTime") +
  ylab("Residuals") +
  ggtitle("Residuals over the DSTTime")
```




```
#
```

```
## Year & Temperature
# plot_ly(x = data.2018$Year, y= data.2018$Temp,
#         type = "scatter",
#         mode = "lines"
#         ) %>% layout(xaxis = list(title = "Year"), yaxis = list(title = "Temperature"))

ggplot(data.2018, aes(x = Temp, y = residuals.2018)) +
  geom_point(color = grey(0, 0.1)) +
  xlab("Temp") +
  ylab("Residuals")+
  ggtitle("Residuals over Temp")
```



3. Comparing author's data (2014 NSW) VS our new data (2018-2022 SA)

```
## preprocessing
df = read.csv("merged_interpolated.csv")
df <- df[,c('datetime', 'tempc', 'net_load')]
names(df) <- c('dt', 'temp', 'demand')
```

1) data processing

1. merged_interpolated.csv [datetime]: ACST - Australia Central Standard Time
2. as.numeric: transform Australia h/min/s => UTC time 00:00:00 => number
3. Simon: first postpone dt 23:30 + 0930 saved as df$ACDT

```
Sys.setenv(TZ = "Australia/South")

# df$ACDT <- as.POSIXct(df$dt, format="%F %T %z", tz="Australia/South")
df$ACDT <- as.POSIXct(paste(df$dt, "+0930"), format="%F %T %z", tz="Australia/South")
# df$ACDT2 <- as.POSIXct(df$dt, format="%Y-%m-%d %H:%M:%S")
## Australian Central Daylight Time
df$isDST <- as.POSIXlt(df$ACDT)$isdst > 0    ## whether implement dst

df$ES <- fasttime::fastPOSIXct(df$dt,tz="Australia/South")
df$ES <- fasttime::fastPOSIXct(df$dt)
df$new_ES = as.POSIXct(df$dt, format="%Y-%m-%d %H:%M:%S", tz="Australia/South")
```

```
## the time before impelementing dst, ACDT+9:30
# df$ED <- df$ES - df$isDST * 3600 ## adjust for DST
df$new_ED = df$new_ES - df$isDST * 3600
df$new_ED <- as.POSIXct(df$new_ED, format="%Y-%m-%d %H:%M:%S")
# df$hr = format(df$ED, "%H")
df[1:5,]
```

```
##                   dt  temp demand                ACDT isDST
## 1 2018-03-06 09:30:00 20.75   1288 2018-03-06 10:30:00  TRUE
## 2 2018-03-06 10:00:00 21.50   1237 2018-03-06 11:00:00  TRUE
## 3 2018-03-06 10:30:00 22.25   1189 2018-03-06 11:30:00  TRUE
## 4 2018-03-06 11:00:00 23.00   1150 2018-03-06 12:00:00  TRUE
## 5 2018-03-06 11:30:00 23.55   1122 2018-03-06 12:30:00  TRUE
##                   ES              new_ES              new_ED
## 1 2018-03-06 20:00:00 2018-03-06 09:30:00 2018-03-06 08:30:00
## 2 2018-03-06 20:30:00 2018-03-06 10:00:00 2018-03-06 09:00:00
## 3 2018-03-06 21:00:00 2018-03-06 10:30:00 2018-03-06 09:30:00
## 4 2018-03-06 21:30:00 2018-03-06 11:00:00 2018-03-06 10:00:00
## 5 2018-03-06 22:00:00 2018-03-06 11:30:00 2018-03-06 10:30:00
```

```
df$dst  = (as.numeric(df$new_ES) %% 86400) / 86400
# chase = as.POSIXct(df$dt, format="%F %T", tz="UTC")
# new_chase <- as.POSIXct(paste(chase, "+0930"), format="%F %T %z", tz="Australia/South")
# new2_chase = fasttime::fastPOSIXct(new_chase)
```

```
## some extra columns ...
d = df
d$yfrac = as.POSIXlt(d$ES)$yday/365
d$year  = as.POSIXlt(d$ES)$year+1900
d$dow   = (as.POSIXlt(d$ES)$wday + 1) %% 7
```

Holiday issues:

```
# d$time = (as.numeric(d$ES) %% 86400) / 86400
# d$dst  = (as.numeric(d$ED) %% 86400) / 86400
# d$day  = as.numeric(d$ES) %/% 86400
# d$dayd = as.numeric(d$ED) %/% 86400
```

```
#chase
df = read.csv("merged_interpolated.csv")
df2 <- df[,c('datetime', 'tempc', 'net_load')]

df2$acst <- as.POSIXct(df2$datetime, format="%F %T", tz="UTC")


df2$acdt <- as.POSIXct(paste(df2$datetime, "+1030"), format="%F %T %z", tz="Australia/South")
head(df2)
```

```
##              datetime tempc net_load                acst                acdt
## 1 2018-03-06 09:30:00 20.75     1288 2018-03-06 09:30:00 2018-03-06 09:30:00
## 2 2018-03-06 10:00:00 21.50     1237 2018-03-06 10:00:00 2018-03-06 10:00:00
## 3 2018-03-06 10:30:00 22.25     1189 2018-03-06 10:30:00 2018-03-06 10:30:00
## 4 2018-03-06 11:00:00 23.00     1150 2018-03-06 11:00:00 2018-03-06 11:00:00
## 5 2018-03-06 11:30:00 23.55     1122 2018-03-06 11:30:00 2018-03-06 11:30:00
## 6 2018-03-06 12:00:00 24.10     1102 2018-03-06 12:00:00 2018-03-06 12:00:00
```

```r
Sys.time()
```

```
## [1] "2023-05-05 10:25:29 ACST"
```

```r
unclass(Sys.time())
```

```
## [1] 1683248130
```

```r
# as.character(.POSIXct(runif(1e4) * unclass(Sys.time()), "GMT"))
```

```r
## preprocessing
library(lubridate)
```

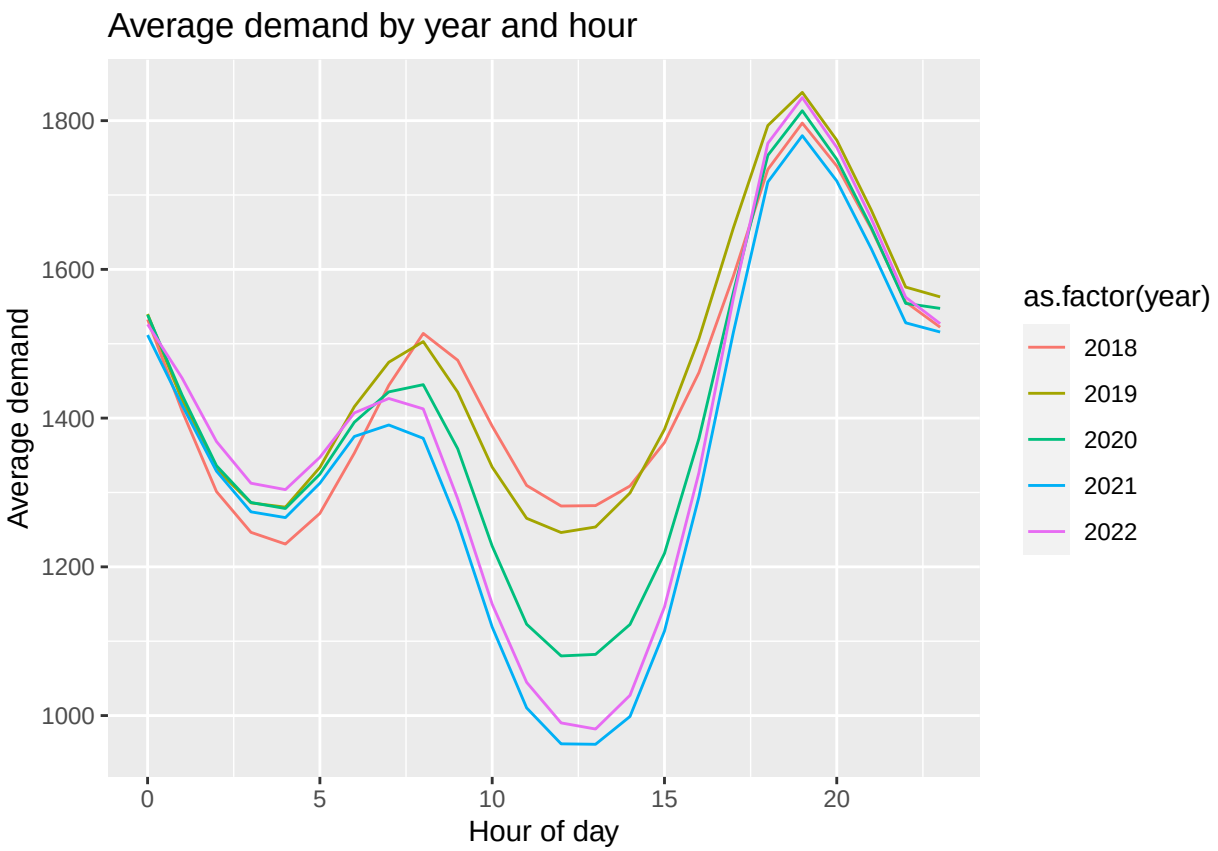
```
## Warning:  'lubridate' R 4.2.3
```

```
##
##     'lubridate'
```

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

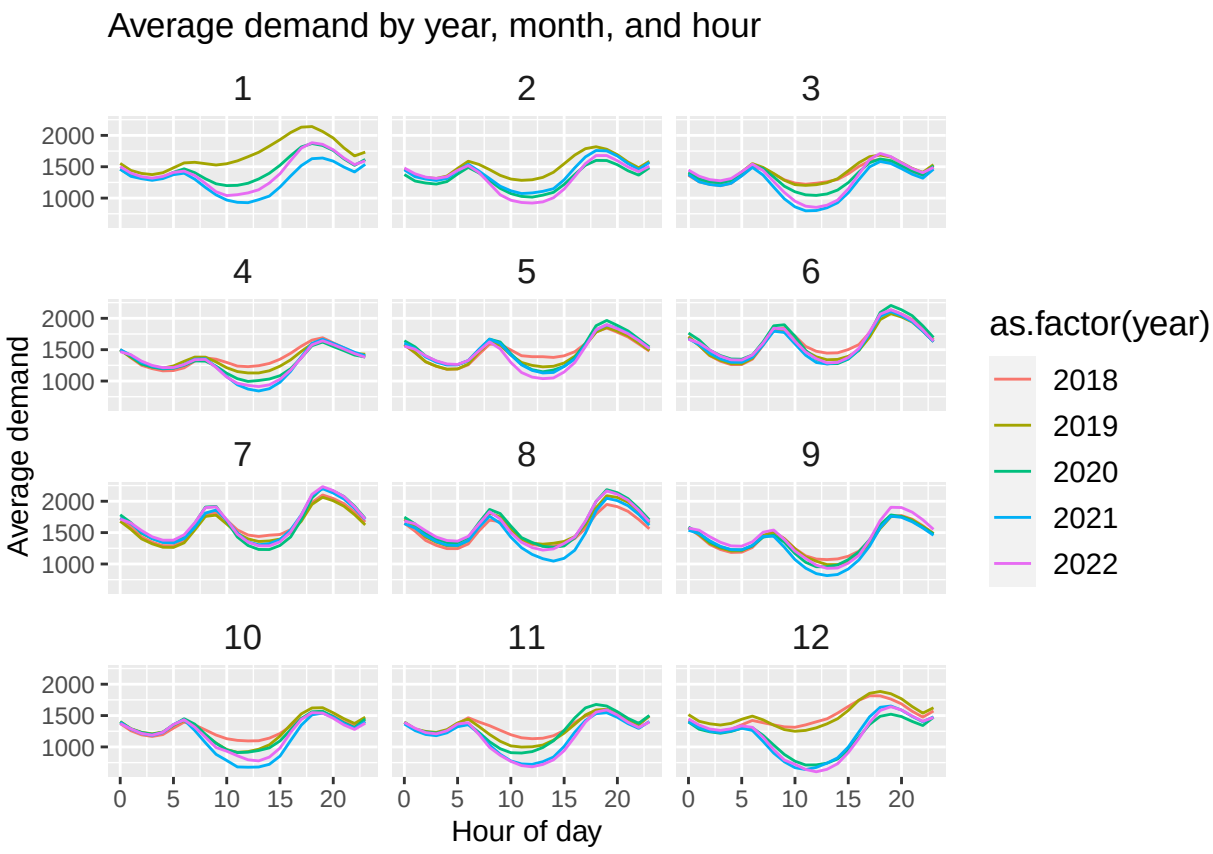
```r
df = read.csv("merged_interpolated.csv")
df <- df[,c('datetime', 'tempc', 'net_load')]
names(df) <- c('dt', 'temp', 'demand')
df$ACDT <- as.POSIXct(df$dt, format="%Y-%m-%d %H:%M:%S")
df$isDST <- as.POSIXlt(df$ACDT)$isdst > 0
df$ED = df$ACDT - df$isDST * 3600
df$year = as.numeric(format(df$ED, "%Y"))
df$month = as.numeric(format(df$ED," %m"))
df$hour = as.numeric(format(df$ED, "%H"))
df$minutes = as.numeric(format(df$ED, "%M"))
df$DST = (df$hour*60 + df$minutes) / 1440
df$yday = yday(df$ED)
df$yfrac = df$yday/365
df$dw <- wday(df$ED, week_start = 1)
```

```r
df_weekday = df[df$dw>=1 & df$dw<=5,]
# Load the ggplot2 package
library(ggplot2)
df_weekday2022 = df_weekday[df_weekday$year %in% c(2018,2019,2020,2021,2022),]

# Calculate the mean demand for each year and hour combination
mean_demand <- aggregate(demand ~ year + hour, data = df_weekday2022, FUN = mean)

# Plot a line graph showing the average demand by year and hour
ggplot(mean_demand, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Hour of day") +
  ylab("Average demand") +
  ggtitle("Average demand by year and hour")
```



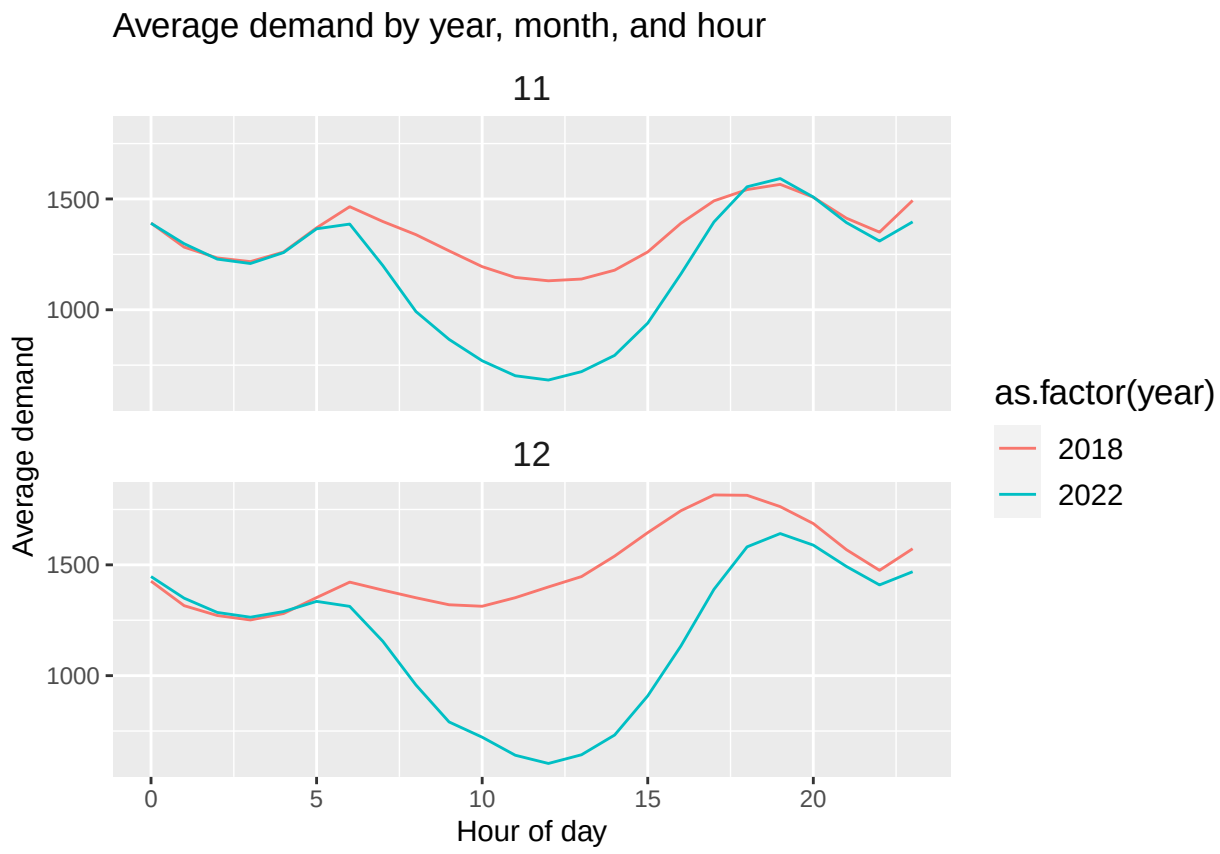
2018-2022 SA demand (by hour of day): (2023 not end yet) * overall pattern: generally consistent shape, but peak & valleys differ a lot. * morning peak shift to left; * night peak stay the same

```r
# Load the ggplot2 package
library(ggplot2)

# Calculate the mean demand for each year, month, and hour combination
mean_demand <- aggregate(demand ~ year + month + hour, data = df_weekday2022, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
ggplot(mean_demand, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Hour of day") +
  ylab("Average demand") +
  ggtitle("Average demand by year, month, and hour") +
  facet_wrap(~ month, ncol = 3) +
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"),
        legend.text = element_text(size = rel(1)),
        legend.title = element_text(size = rel(1.2)))
```



Drill into 12 months: the pattern still very similar among different years, (Dec.: much lower in recent years)

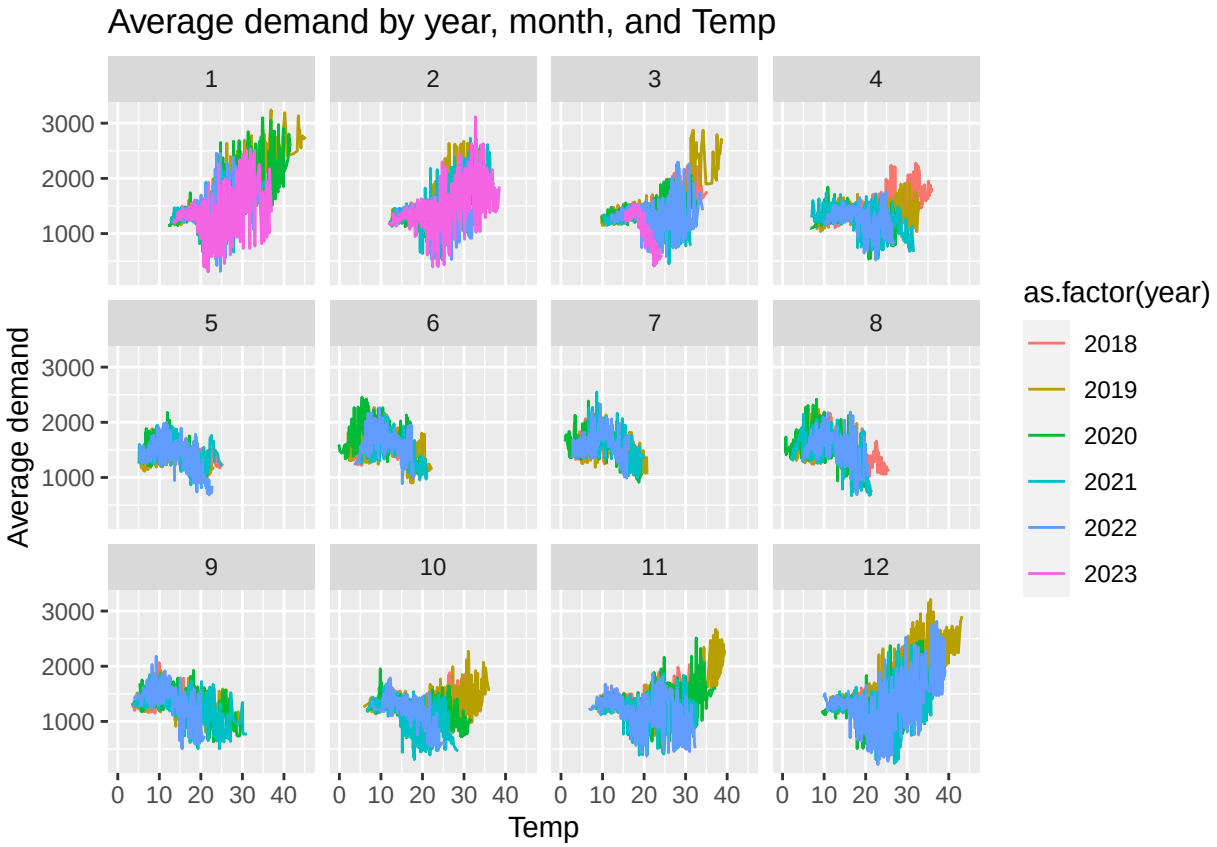
```
# Load the ggplot2 package
library(ggplot2)
df_neat = df_weekday[df_weekday$year %in% c(2018,2022) & df_weekday$month %in% c(11,12), ]

# Calculate the mean demand for each year, month, and hour combination
mean_demand <- aggregate(demand ~ year + month + hour, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
ggplot(mean_demand, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Hour of day") +
  ylab("Average demand") +
  ggtitle("Average demand by year, month, and hour") +
  facet_wrap(~ month, ncol = 1) +
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"),
        legend.text = element_text(size = rel(1)),
        legend.title = element_text(size = rel(1.2)))
```



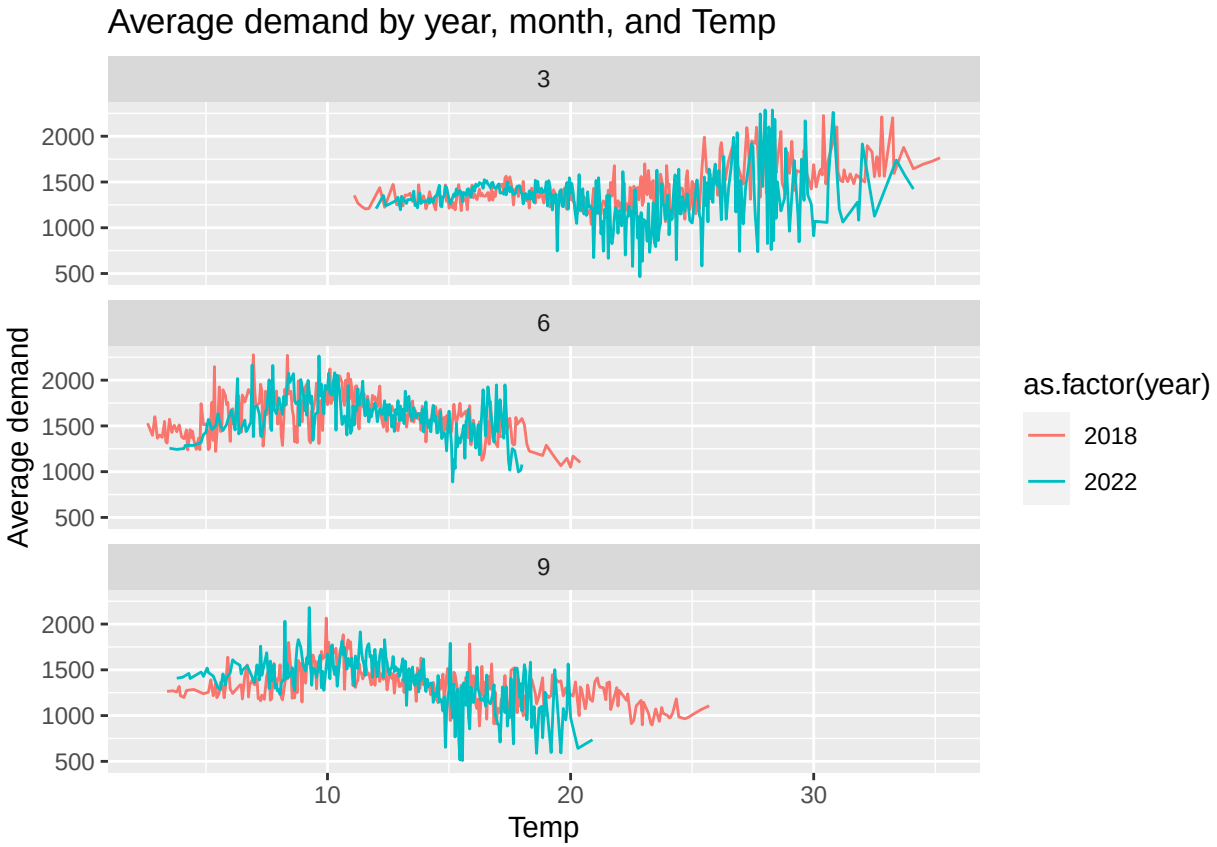
guess?? https://www.roymorgan.com/findings/9091-solar-energy-systems-on-households-more-than-double-since-2018-now-at-nearly-a-third-of-all-households

New South Wales Solar panel rebate program was launched in July 2018 to lower solar panel costs for homes. This legislation promotes solar panels for homes in all states. The solar rebate program was developed by Environment Australia and the NSW government to increase renewable energy systems.

```
library(ggplot2)

# Calculate the mean demand for each year, month, and hour combination
mean_demand <- aggregate(demand ~ year + month + temp, data = df_weekday, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
ggplot(mean_demand, aes(x = temp, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Temp") +
  ylab("Average demand") +
  ggtitle("Average demand by year, month, and Temp") +
  facet_wrap(~ month, ncol = 4)
```




```r
df_neat = df_weekday[df_weekday$year %in% c(2018,2022) & df_weekday$month %in% c(3,6,9), ]
mean_demand <- aggregate(demand ~ year + month + temp, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
ggplot(mean_demand, aes(x = temp, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Temp") +
  ylab("Average demand") +
  ggtitle("Average demand by year, month, and Temp") +
  facet_wrap(~ month, ncol = 1)
```

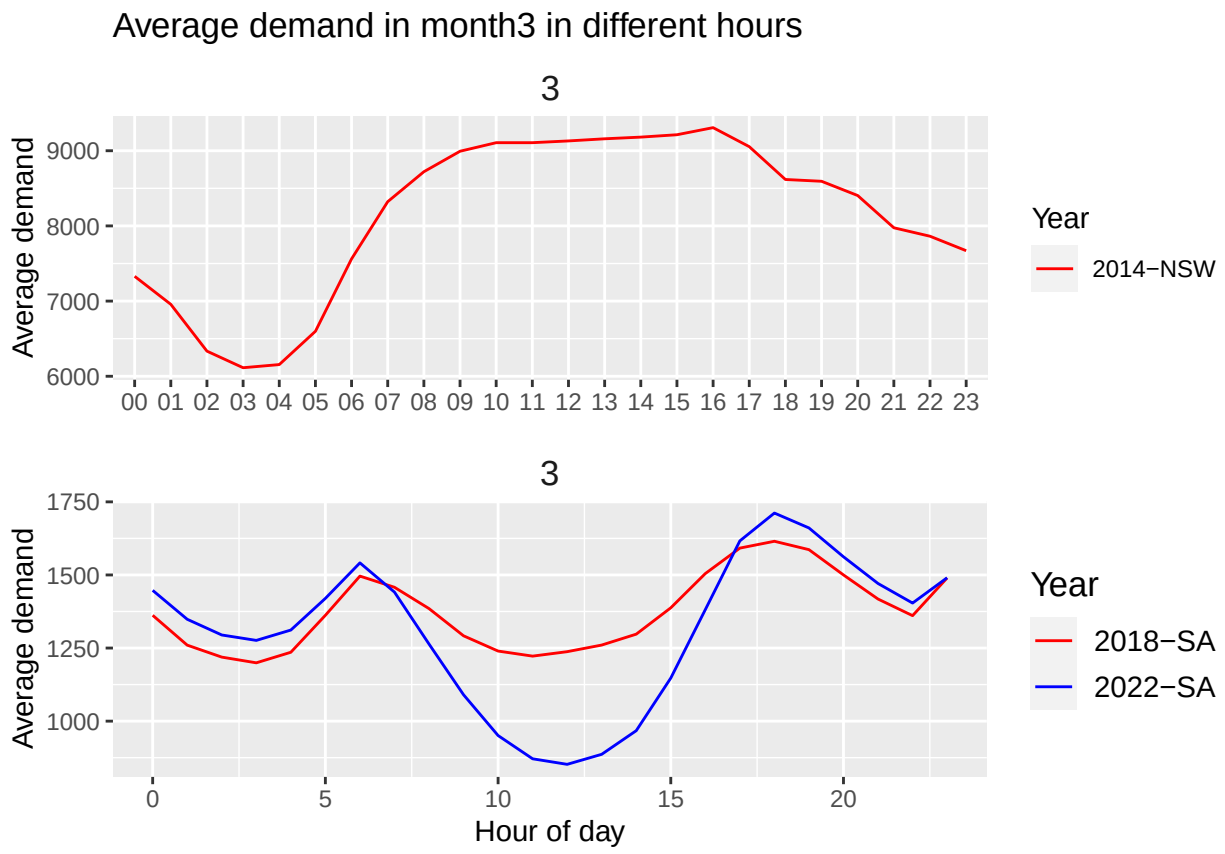



```
library(gridExtra)
```

```
## Warning:   'gridExtra' R 4.2.3
```

```
data2014 = read.csv("original2014.csv")
tuned2014 = data2014[c("DescDate","DSTAscTime","Temp","Month","Demand")]

tuned2014$year = substr(tuned2014$DescDate,1,2)
tuned2014 =tuned2014[tuned2014$year==14,]
# tuned2014$hour = as.numeric(ifelse(grepl(":", tuned2014$DSTAscTime), substr(tuned2014$DSTAscTime, 1,

tuned2014$hour <- strftime(strptime(tuned2014$DSTAscTime, format="%H:%M"), format="%H")

colnames(tuned2014)[c(3, 4,5)] <- c("temp", "month","demand")
# merge_years <- merge(tuned2014, df_weekday, by = c("year","temp","month","hour","demand"), all = TRUE
#
# # ------------------------------
# df_neat = merge_years[merge_years$year %in% c(14,2018,2022) & merge_years$month %in% c(3,7,11), ]
#
# # Calculate the mean demand for each year, month, and hour combination
# mean_demand <- aggregate(demand ~ year + month + hour, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
final_tuned2014 = tuned2014[tuned2014$month %in% c(3), ]
mean_demand2014 <- aggregate(demand ~  year + month + hour, data = final_tuned2014, FUN = mean)

a= ggplot(mean_demand2014, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
```

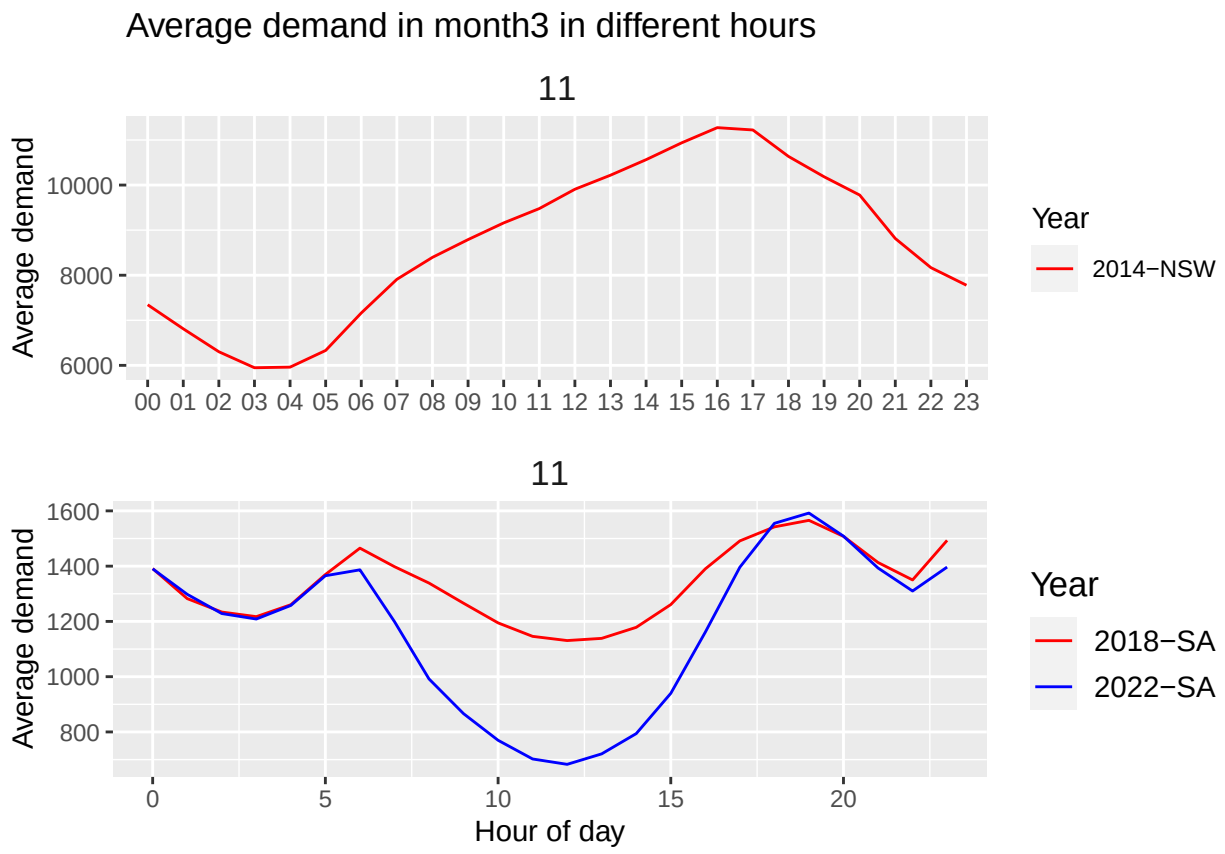
```
  geom_line() +
  xlab(NULL) +
  ylab("Average demand") +
  ggtitle("Average demand in month3 in different hours") +
  facet_wrap(~ month, ncol = 1) +
  scale_color_manual(values = c("red", "blue", "green"),
                     name = "Year",
                     labels = c("2014-NSW", "2018-SA", "2022-SA"))+
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"))

df_neat = df_weekday[df_weekday$year %in% c(2018,2022) & df_weekday$month %in% c(3), ]

# Calculate the mean demand for each year, month, and hour combination
mean_demand <- aggregate(demand ~ year + month + hour, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
b = ggplot(mean_demand, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Hour of day") +
  ylab("Average demand") +
  facet_wrap(~ month, ncol = 1) +
  scale_color_manual(values = c("red", "blue"),
                     name = "Year",
                     labels = c("2018-SA", "2022-SA"))+
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"),
        legend.text = element_text(size = rel(1)),
        legend.title = element_text(size = rel(1.2)))
grid.arrange(a, b, ncol = 1)
```



Explains our lower R-sqaure for SA 2018-2022

```
library(gridExtra)

data2014 = read.csv("original2014.csv")
tuned2014 = data2014[c("DescDate","DSTAscTime","Temp","Month","Demand")]

tuned2014$year = substr(tuned2014$DescDate,1,2)
tuned2014 =tuned2014[tuned2014$year==14,]
# tuned2014$hour = as.numeric(ifelse(grepl(":", tuned2014$DSTAscTime), substr(tuned2014$DSTAscTime, 1,

tuned2014$hour <- strftime(strptime(tuned2014$DSTAscTime, format="%H:%M"), format="%H")

colnames(tuned2014)[c(3, 4,5)] <- c("temp", "month","demand")
# merge_years <- merge(tuned2014, df_weekday, by = c("year","temp","month","hour","demand"), all = TRUE
#
# # ------------------------------
# df_neat = merge_years[merge_years$year %in% c(14,2018,2022) & merge_years$month %in% c(3,7,11), ]
#
# # Calculate the mean demand for each year, month, and hour combination
# mean_demand <- aggregate(demand ~ year + month + hour, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
final_tuned2014 = tuned2014[tuned2014$month %in% c(11), ]
mean_demand2014 <- aggregate(demand ~  year + month + hour, data = final_tuned2014, FUN = mean)

a= ggplot(mean_demand2014, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
```

```
  geom_line() +
  xlab(NULL) +
  ylab("Average demand") +
  ggtitle("Average demand in month3 in different hours") +
  facet_wrap(~ month, ncol = 1) +
  scale_color_manual(values = c("red", "blue", "green"),
                     name = "Year",
                     labels = c("2014-NSW", "2018-SA", "2022-SA"))+
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"))

df_neat = df_weekday[df_weekday$year %in% c(2018,2022) & df_weekday$month %in% c(11), ]

# Calculate the mean demand for each year, month, and hour combination
mean_demand <- aggregate(demand ~ year + month + hour, data = df_neat, FUN = mean)

# Plot a line graph showing the average demand by year, month, and hour, split by month
b = ggplot(mean_demand, aes(x = hour, y = demand, group = year, color = as.factor(year))) +
  geom_line() +
  xlab("Hour of day") +
  ylab("Average demand") +
  facet_wrap(~ month, ncol = 1) +
  scale_color_manual(values = c("red", "blue"),
                     name = "Year",
                     labels = c("2018-SA", "2022-SA"))+
  theme(plot.title = element_text(size = rel(1.2)),
        axis.title = element_text(size = rel(1)),
        strip.text = element_text(size = rel(1.2)),
        strip.background = element_rect(fill = "white"),
        legend.text = element_text(size = rel(1)),
        legend.title = element_text(size = rel(1.2)))
grid.arrange(a, b, ncol = 1)
```


Average demand in month3 in different hours
11
Average demand
10000
8000
6000
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23
Year
2014−NSW
11
Average demand
1600
1400
1200
1000
800
0
5
10
15
20
Hour of day
Year
2018−SA
2022−SA