# ภาคผนวก ข

# ตัวอย่างคำสั่งของโปรแกรม R ในการวิเคราะห์ข้อมูล

## การวิเคราะห์ปริมาณการใช้ไฟฟ้าของมหาวิทยาลัยขอนแก่น โดยวิธีการทำให้เรียบ

1. ติดตั้งแพ็กเกจและเรียกใช้แพ็กเกจ

```
install.packages('ggplot2')
install.packages('forecast')
install.packages('tseries')
library('ggplot2')
library('forecast')
library('tseries')
```

2. นำฐานข้อมูลเข้า

```
kku = read.csv("D:/electricity/kku.csv")
```

3. สร้างอนุกรมเวลา โดยกำหนดค่าเริ่มต้นของข้อมูล และค่าสิ้นสุดของข้อมูล

```
kku_ts = ts(kku$E_Consumption , frequency = 12, start = c(2551,1), end = c(2561,5))
```

4. พล็อตกราฟเส้นและกราฟ ACF เพื่อดูลักษณะการเคลื่อนไหวของข้อมูล

```
plot.ts(kku_ts, ylab="ปริมาณการใช้ไฟฟ้า (หน่วย)",xlab="เวลา (ปี)", col="darkorange2", main= "มหาวิทยาขอนแก่น")
Acf(kku_ts, main='มหาวิทยาลัยขอนแก่น', lag = 125)
```

5. เลือกตัวแบบที่เหมาะสมกับลักษณะของข้อมูล

```
kku_HWaddauto = HoltWinters(kku_ts)
```

6. หาค่าพยากรณ์

```
y_hat4 <- kku_HWaddauto$fitted[,1]
```

7. หาค่าคลาดเคลื่อน

```
error4 =  kku_ts - y_hat4
```

8. หาค่าพยากรณ์ล่วงหน้า 12 เดือน

```
kku_HWaddauto.pred = predict( kku_HWaddauto, n.ahead=12, prediction.interval=TRUE)
```

9. หาค่าคลาดเคลื่อน RMSE, MAE และ MAPE

```
summary(kku)
```

10. พล็อตกราฟเส้นเปรียบเทียบระหว่างค่าจริงกับค่าพยากรณ์

```
plot.ts( kku_ts, ylab="ปริมาณการใช้ไฟฟ้า (หน่วย)" , main = "มหาวิทยาลัยขอนแก่น")
lines(kku_HWaddauto$fitted[,1], lty=2, col="red")
legend("topleft",legend=c("Data", "Fitted"),col=c("black", "red"), lty=1:2, cex=0.8)
```

## การวิเคราะห์ปริมาณการใช้ไฟฟ้าของมหาวิทยาลัยขอนแก่น โดยวิธีของบ็อกซ์และเจนกินส์

1. ติดตั้งแพ็กเกจและเรียกใช้แพ็กเกจ

```
install.packages('ggplot2')          install.packages('forecast')
install.packages('tseries')          install.packages('lmtest')
install.packages('ggpubr')           install.packages('nortest')
install.packages('Metrics')
library('ggplot2')          library('forecast')
library('tseries')          library('lmtest')
library('ggpubr')           library('nortest')
library('Metrics')
```

2. นำฐานข้อมูลเข้า

```
dentistdata = read.csv('D:/electricity/Dentist.csv', header=TRUE)
```

3. สร้างอนุกรมเวลา และพล็อตกราฟเส้น กราฟ ACF และ PACF เพื่อดูลักษณะการเคลื่อนไหวของข้อมูล

```
dentist_ts= ts(dentistdata[,c('E_Consumption')], frequency = 12, start = c(2551, 1), end = c(2561, 5))
plot(dentist_ts, ylab="ปริมาณการใช้ไฟฟ้า (หน่วย)",xlab="เวลา (ปี)",col='mediumorchid3', main=
"คณะทันตแพทยศาสตร์")
Acf(dentist_ts, main='คณะทันตแพทยศาสตร์', lag = 125)
Pacf(dentist_ts, main='คณะทันตแพทยศาสตร์', lag = 125)
```

4. หากข้อมูลไม่สเตชันนารีให้ทำการหาผลต่าง หรือ หาผลต่างฤดูกาล จนกว่าข้อมูลจะสเตชันนารี

nsdiffs() คือ คำสั่งที่นับว่าควรหาผลต่างฤดูกาลกี่ครั้งจึงจะสเตชันนารี

ndiffs() คือ คำสั่งที่นับว่าควรหาผลต่างกี่ครั้งจึงจะสเตชันนารี

dentist_d1 = diff(DenLog, differences = 1) คือ การหาผลต่าง 1 ครั้ง

Acf(dentist_d1, main='ACF for Differenced Series', lag=125)

Pacf(dentist_d1, main='PACF for Differenced Series', lag=125)

dentist_d12 = diff(dentist_d1, differences = 1, lag=12) คือ การหาผลต่างฤดูกาล 1 ครั้ง

Acf(dentist_d12, main='ACF for Differenced Season Series', lag=125)

Pacf(dentist_d12, main='PACF for Differenced Season Series', lag=125)

4.1 กรณีที่ข้อมูลความแปรปรวนไม่คงที่ สามารถแปลงข้อมูลได้ดังนี้

DenLog=log(dentist_ts)

SQRdentist=sqrt(dentist_ts)

fracden=1/dentist_ts

5. ทำการเลือกตัวแบบ

dentistmodel4=auto.arima(DenLog,d=1,D=1) คือ การให้โปรแกรมเลือกตัวแบบให้อัตโนมัติ โดยโปรแกรมจะเลือกตัวแบบที่ให้ค่า AIC ต่ำสุด และสามารถกำหนดคุณสมบัติของตัวแบบที่ต้องการได้

summary(dentistmodel4) คือ คำสั่งให้แสดงผลการเลือกตัวแบบ

coeftest(dentistmodel4) คือ การทดสอบค่าพารามิเตอร์ในตัวแบบว่าเท่ากับศูนย์หรือไม่

dentistmodel3=arima(DenLog, order=c(1,1,1), seasonal = list(order=c(0,1,1))) #aic = -64.99 คือ คำสั่งที่สามารถกำหนดอันดับ p, d, q, P, D และ Q ในตัวแบบได้ด้วยตัวเอง

6. ตรวจสอบความเหมาะสมของตัวแบบ โดยตรวจสอบคุณสมบัติของค่าความคลาดเคลื่อน

6.1 ตรวจสอบสหสัมพันธ์ในตัวเองของความคลาดเคลื่อน

tsdisplay(residuals(dentistmodel4), lag.max=125, main='(3,1,2)(0,1,1) Model Residuals')

# Ljung-Box Q Test

Box.test(residuals(dentistmodel4), lag = 124, type = "Ljung")

6.2 ตรวจสอบความคลาดเคลื่อนว่าค่าเฉลี่ยของความคลาดเคลื่อนเป็นศูนย์

```
t.test(dentistmodel4$residuals, mu=0, alternative = 'two.sided')
```

6.3 ตรวจสอบความแปรปรวนของความคลาดเคลื่อน

```
plot(residuals(dentistmodel4), type='p')
```
คือ พล็อตกราฟการกระจายตัวของความคลาดเคลื่อน

6.4 ตรวจสอบความคลาดเคลื่อนว่าความคลาดเคลื่อนมีการแจกแจงปกติ

```
plotForecastErrors(residuals(dentistmodel4))
```
คือ พล็อตกราฟฮิสโตแกรม

7. พยากรณ์ล่วงหน้า 12 เดือน

```
fcast1<- forecast(dentistmodel4, h=12)

plot(fcast1)

summary(forecast(dentistmodel4, h=12))
```

8. สร้างกราฟพยากรณ์ที่แสดงค่าจริงและค่าพยากรณ์

```
plot(dentist_ts, ylab="ปริมาณการใช้ไฟฟ้า (หน่วย)",xlab="เวลา (ปี)", col="red", main= "คณะทันตแพทยศาสตร์")

lines(Refitted,lty = 2, col="blue1")

legend("topleft",legend=c("Data", "Fitted"),col=c("red", "blue1"), lty=1:2, cex=0.8)
```