LABORATION 1

732G52 - HT2024

Intro

Innehåll:

- Hantera tidseriedata i R
- Göra grafer över tidseriedata
- Tidserieregression
- Komponentuppdelning

Denna laboration är till för er övning på kursmaterialet. Det finns ingen obligatorisk inläming för den. Uppgifterna utgår från R och Rstudio. Om du behöver repetition i R:

- Titta på kurshemsidan för 732G33: här.
- Cheat sheets här

Material:

- Introduction to Time Series Analysis and Forecasting (TSAF): se R-kod i boken
- Forecasting: Principles and Practice (FPP), tredje upplagan: länk
- Repo för kursen: länk
- Om man behöver repetition på vanlig linjär regression, se kursen Linjära modeller 1, titta tex här: länk

Se sektionen "The `lm()` function" på sidan 4 för detaljer kring funktionen `lm()`.

Uppgifter

1. `ts`-objekt i R: `ts`-objekt är objekt av klassen "ts" som är speciellt skapad för att hantera tidserier på olika sätt. Det finns många olika funktioner som fungerar för eller är anpassade för ts-objekt. Gå igenom Del 1 i koden som finns här: länk.

2. Nu ska ni kolla på några av de dataset med tidserier som finns i olika R-paket som man direkt kan läsa in med funktionen `data()`. Gå igenom Del 2 i koden som länkas ovan. Här får ni bekanta er med `acf()` (som beräknar sample autocorrelation) och med `lag()` som beräknar diffar eller laggar av en vektor/tidserie.

3. Regression med tidseriedata: När vi arbetar med regression på tidseriedata så är mycket likt fallet när vi jobbar med "vanlig data" (tänk kursen Linjära modeller 1).

   (a) Det är vanligt att man skapar en tidsvariabel/tidsindex (kalla den *time* här) som vi använder för att modellera trender i tidserien. Detta eftersom vi inte kan räkna på "råa datum". Vi kan låta trenden vara linjär, men vi kan använda kvadratisk eller kubisk trend om vi tror att det anpassar data bättre. Då skapar vi lämpliga transformationer på formen: $time$, $time^2$, $time^3$, om vi gör det vill vi ofta centrera eller standardisera *time* först.

   (b) Dock så gäller fortfarande de vanliga antaganden på feltermen för att inferensen ska vara giltig. Vilka är dessa antaganden? Det är vanligt att det finns ett tidsberoende kvar i residualerna efter att man har anpassat en regressionsmodell på tidseriedata. Då ska vi inte göra vanlig inferens (test, konfidensintervall etc), då antagenden för inferensen inte är uppfyllda. Det finns metoder för att göra inferens här, mer om det senare i kursen.

   (c) Gå igenom koden som finns här: länk.

4. Komponentuppdelning: Gå igenom koden som finns här: länk.

5. Ett annat sätt att hantera tidserier med paketet `tsibble`, där objekt av klassen `tbl_ts` används för representera tidserier. "Boken Forecasting: Principles and Practice" (FPP) använder `tsibble`. Se följande länkar för mer information:

   - Introduction to tsibble
   - https://tsibble.tidyverts.org/, https://tidyverts.org/
   - R package: https://cran.r-project.org/web/packages/tsibble/index.html

6. Utgå från boken FPP och gå igenom och återskapa koden i följande kaptiel:

   - 2, 3.1-3.4, 4.1-4.2

7. FPP Kap 2.10 Exercises: Gör uppgift 9.

8. Ni ska nu analysera data över pappersproduktion. Data finns i filen "`pappersproduktion.csv`" och en beskrivning finns i "`sw_prod_paper_90-04.txt`".

   (a) Läs in data som csv-fil i R.

   (b) Skapa en tidsvariabel och gör en tidseriesgraf. Visst finns säsongsvariation. Kommentera.

   (c) Skapa tolv indikatorvariabler, en för varje månad och ge dessa lämpliga namn. Använd 11 av dessa senare i regressionsmodellen.

   (d) Anpassa en regressionsmodell med `lm()`. Ta fram *Four in one* residualplottar, använd residual_diagnostics().

   (e) Tolka valfri dummy-variabel. Utför DW-testet och residualanalys. Ta fram SAC på residualerna. Tolka resultaten.

   (f) Gör en prognos för nästkommande fyra månader. Kommentera.

9. Gör en komponentuppdelning på data över pappersproduktion. Analysera och tolka resultaten.

The `lm()` function

The function `lm()` will be important during the course. Check out the documentation with `?lm()`.

Note that R is an object oriented language, and the `lm()` returns objects with class "`lm`", with has the form of a list, so you can easily fetch different part of the object when needed. These objects has several useful generic functions connected to it:

- `coef()`: Gives the regression coefficients
- `residuals()`: calculates the residuals of the model
- `fitted()`: Gives the fitted values of the model
- `summary()`: give detail summary and inference. It will return a object of class "summary.lm". `coef()` will work on this object.
- `anova()`: Calculates anova table for the model
- `predict()`: make predictions with the model for (new) data
- `plot()`: output diagnostics plots for the model

In general, for documentation for these methods run commands of the type `?summary.lm()` in the terminal. Another useful function is to use `str()` on the lm-object, to get detailed information about it.