Datorlaboration 1 och inlämningsuppgift 2 - Del A

Undersökningsmetodik, Statistik och dataanalys 2 (ST1201) HT2023.

Mona Sfaxi, Michael Carlson och Oscar Oelrich

Det här dokumentet innehåller en del material från R-kursen i statistik på masternivå HT20 av Ellinor Fackle Fornius och Jessica Franzén.

Innehåll

Introduktion

Alla datorlaborationer ska genomföras som *Quarto-notebooks*. En stor fördel med notebook-formatet är att det låter er skapa ert egna kursmaterial genom att kombinera kod med text som beskriver vad koden gör. Att skriva sitt egna kursmaterial, alltså att med egna ord tvingas förklara hur saker fungerar, är ett av dom bästa sätten att lära sig.

- Det är helt OK att ni sammarbetar under labben, men skriv din egna labbrapport! Det är viktigt att faktiskt skriva koden själv.
- Om du fastnar, testa att se om du kan hitta en lösning i dokumentationen. För att se dokumentationen för en viss funktion skriver du helt enkelt ett frågetecken följt av funktionens namn i konsolen, exempelvis `?lm`. Funkar inte det, testa google, och funkar inte det, fråga labbansvarig. Vi finns där för att svar på dina frågor, med det är viktigt att du tränar på att lösa problem själv.

I den här laborationen kommer du att lära dig hur man laddar ner och läser in data från externa källor som exempelvis kommer från olika organisationer och hemsidor. Oftast anses nedladdning, inläsning och bearbetning av data vara bland de tråkigaste och krångligaste delarna av programmering. Men oroa dig inte, efter dagens datorlaboration kommer du att älska att läsa in data i R och som bonus vara redo att arbeta med del A på inlämningsuppgiften!

Paket som kommer behövas är:

- `pxweb`
- `readxl`

Och andra paket som starkt rekommenderas är:

- `tidyverse`

Externa datakällor

Det finns väldigt mycket data online som kan laddas ner med olika metoder. API står för `Application Programming Interface` och flera organisationer, företag och myndigheter gör det möjligt för användare att använda sig av API för att ladda ner data. Förenklat kan man säga att API gör det möjligt för datorer att prata med varandra.

`R` erbjuder flera olika paket för att använda sig av API för nedladdning av data. Några fördelar med att använda sig av API är att man kan få tillgång till data genom att programmera och utan att behöva navigera och klicka sig runt på en hemsida.

Ibland kan data laddas ner direkt, men hos vissa aktörer kan man först behöva regsitrera ett konto hos dem innan det blir möjligt. Ibland kan dataseten och deras format också vara annorlunda från vad man hade förväntat sig vid användning av API. En del API kan även vara komplicerade att använda sig av samt så kan det finnas en begränsning av hur mycket man kan ladda ner under en viss tidsperiod.

Ett annat väldigt poulärt verktyg är s.k. Webbscraping som används flitigt inom maskininlärning, väldigt ofta då genom `Python` där man extraherar html-kod för att kopiera data. Men det är inget vi kommer att syssla med i den här kursen.

PXWEB

Är ett API som har utvecklats av statistiska centralbyrån (SCB) och andra nationella statistiska institutioner. R paketet `pxweb` kan användas för att få `R` att interagera med alla PXWEB-API.

- ☐ Börja med att installera paketet `pxweb` med hjälp av install.packages(). Detta bör endast göras en gång. Sedan kan man enkelt kommentera bort den koden med en #. **Att installera om ett paket som redan är installerat varje gång man renderar ett dokument är ineffektivt och bör undvikas.**
- ☐ Aktivera paketet med hjälp av `library()` funktionen.
- ☐ Ladda även paketet `readxl` (alternativt `openxlsx`) samt andra paket som kan komma till användning i samma code-chunk längst upp i ditt dokument. Glöm inte att filtrera ut störande meddelanden med kommandot `#| output: false` längst upp i code-chunken när du aktiverar paketen.

Del 1 - Navigering

Ibland vill man först navigera bland olika API för att sedan kunna ladda ner data ifrån dem.

- ☐ Använd funktionen `interactive_pxweb()` för att undersöka i consolen i `R` vilka API som finns tillgängliga. Du bör få upp något som ser ut så här:

```
==================================================

R PXWEB API CATALOGUE:

==================================================

[ 1 ] : Statistics Sweden

[ 2 ] : Statistics Finland

[ 3 ] : Statistics Finland (old version)

[ 4 ] : The Swedish Agricultural Agency

[ 27 ] : Helsingin seudun aluesarjat -tilastotietokanta

[ 28 ] : Estonia - official statistics

[ 29 ] : Nordic Statistics Database

[ 30 ] : SiStat Database

==================================================

Enter your choice:
```

('esc' = Quit, 'a' = Show all, 'i' = Show id)

Hur många är från svenska sidor? Navigera gärna genom att skriva in ett valfritt nummer i consolen och sedan `Enter` Lägg märke till att man kan backa tillbaka genom att skriva `b` följt av `Enter`.

- ☐ Använd `esc` knappen för att avbryta interaktionen via `pxweb`. Lägg dock märke till att Quarto dokumentet inte kommer kunna renderas då det endast står `interactive_pxweb()` i en code-chunk. Detta är för lite information och koden bör endast användast för navigering och för att hitta data som man senare tänker ladda ner.

Del 2 - Hämta tidsserien KPIF via API

I den här deluppgiften kommer du att ladda ner och presentera data över inflationen i Sverige utifrån tidsserien konsumentprisindex med fast ränta (KPIF) som bland annat används av riksbanken och andra institutioner i analyser över den ekonomiska utvecklingen.

Del 2a - ladda ner data

Börja med att använda dig av funktionen `interactive_pxweb()` i consolen.

- ☐ Återigen kommer en lista att presenteras med olika institutioner, varav de flesta inte är svenska. Klicka på 1 följt av `Enter` i consolen när katalogen dyker upp för att kunna bläddra i SCB:s databas.
- ☐ Du bör nu få fram en text där det står `R PXWEB: Content of 'api.scb.se'` följt av `[1] : v1` på nästkommande rad. Skriv igen in 1 följt av `Enter`.
- ☐ Du kommer sedan att få välja språk. välj svenska genom att skriva 2 och sen `Enter`.
- ☐ Därefter kommer det endast dyka upp ett alternativ där det står `Statistikdatabasen`. Välj alternativet genom att skriva in en 1:a följt av `Enter`.
- ☐ Nu kommer det presenteras en massa alternativ. Mata in ett `a` följt av `Enter` i consolen för att få en överblick över alla ämnesområden. Välj alternativ 18 som står för Priser och konsumtion och tryck sedan på `Enter`.
- ☐ Flera olika indexserier listas fram, välj den första (`KPI`) genom att skriva in 1 följt av `Enter`.
- ☐ En ny lista kommer att presenteras, välj nummer 3 i listan (`KPIF`), då detta vanligtvis används som inflationsmått, och tryck sedan på `Enter`.

- ☐ En ny text kommer att presenteras där det står `[ 1 ] : Konsumentprisindex med fast ränta (KPIF), 1987=100. Månad 1987M01 - 2023M10`, verifiera att du vill ladda ner serien genom att trycka på 1 följt av `Enter`.
- ☐ Du kommer att presenteras med tre olika alternativ, välj det tredje genom att skriva in en 3:a följt av `Enter`.
- ☐ En lista med alla tidsperioder kommer dyka upp. Välj alla genom att skriva in `*` och sedan `Enter` i consolen.
- ☐ En text där det står `Do you want to print code to query and download data?` välj ja genom att skriva `y` och sedan `Enter`. Denna kod kommer att behövas senare ifall du vill ladda ner datasetet igen i R, utan att behöva gå igenom alla steg ovan.
- ☐ Sedan kommer en till fråga att dyka upp där det står `Do you want to print query in json format (otherwise query is printed as an R list)?`. Här går det bra att svara ja ifall man vill men det är inte nödvändigt så svara nej genom att skriva `n` följt av `Enter`.
- ☐ En ny fråga kommer då att ställas där det står `Do you want to download the data?`. Egentligen bör du svara ja här ifall du direkt vill ladda ner datasetet. Men svara istället nej genom att skriva `n` följt av `Enter`. Detta kommer att avsluta interaktionen och i consolen kommer det att dyka upp en massa kod som du bör kopiera och spara i ditt Quarto dokument.

Det allra viktigaste är att spara ner det som står i listan `pxweb_query_list` samt även det som står i funktionen `pxweb_get()`. I `pxweb_query_list` står `ContentsCode` för seriens kodnamn, varje dataset har ett eget kodnamn så att det enkelt ska gå att identifiera det. Därefter kan det komma fler specifikationer i listan, där det står vilka variabler som har valts ut och för vilken tidsperiod.

Det första argumentet i `pxweb_get()` pekar på vart datasetet ska laddas ner ifrån, dvs själva adressen. Det andra argumentet anger vilka variabler och värden som ska laddas ner. Om allt har gått som det ska har du säkert fått något som ser ut på liknande vis:

```
pxweb_query_list <-
  list("ContentsCode"=c("Kodnummer"),
       "Tid"=c("1987M01","1987M02","... ","2023M02","2023M10"))

px_data <-
  pxweb_get(url = "https://någon adress",
            query = pxweb_query_list)
```

Som vi ser ovan så kan listan bli väldigt lång pga att det finns över 440 observationer. För att undvika detta kan man skriva in att `"Tid"= c("*")`. På så sätt kommer alla observationer att väljas utan att de tar upp en massa onödig plats i ens quarto dokument.

Lägg märke till att `px_data` som nu blivit definierad också är en lista med en massa olika komponenter varav själva observationerna är en del av denna lista. För att få fram en smidig tabell används funktionen `as.data.frame()` som extraherar objektet `data` från listan `px_data` och omvandlar det till en dataframe som blir mycket mer användarvänlig. Detta görs med koden nedan.

```
px_data_frame <- as.data.frame(px_data, column.name.type = "text",
                               variable.value.type = "text")
```

Genom att spara koden i Quarto dokumentet så kan du nu läsa in indexserien utan att behöva ladda ner den som en separat fil nästa gång du öppnar R. Lägg dock märke till att du kommer att få ett varningsmeddelande när du laddar ner datasetet. Detta beror på att de första 12 observationerna i serien saknas (vilket är helt naturligt i det här fallet).

Att referera till källor på ett korrekt sätt är en viktig del av att skriva rapporter. Koden `pxweb_cite(px_data)` kan användas för att referera till själva datasetet. Du bör få flera stycken med text där de senare styckena kan användas ifall man använder sig av LaTeX. Men det som är relevant för dig (om du inte använder dig av LaTeX) är det första stycket där det står:

Statistics Sweden (2023). *"Konsumentprisindex med fast ränta (KPIF), 1987=100 efter tabellinnehåll och månad."* [Data accessed 2023-12-07 14:47:08 using pxweb R package 0.16.2], <URL: https://api.scb.se/OV0104/v1/doris/sv/ssd/PR/PR0101/PR0101G/KPIF>.

Efter att du har kopierat och klistrat in denna text i ditt dokument bör du sedan kommentera bort koden `pxweb_cite(px_data)` likt:

```
# pxweb_cite(px_data)
```

För att undvika en massa rader med störande text som ändå inte kommer användas i ditt dokument.

Del 2b - Illustrera serien grafiskt

Efter att ha laddat ner tidsserien så bör du ha fått en tabell med två kolumner där den ena representerar tid och den andra inflationen (som då går under namnet *KPIF, 12-månadsförändring, 1987=100*). Innan du går vidare med att presentera variabeln grafiskt så bör du först rensa bort de 12 saknade observationerna i tabellen. Passa även gärna på att döpa om tabellen till något mer passande, kort och informativt. Detta kan exempelvis enkelt göras med koden

```
min_nya_df <- min_df[-(1:12), ]
```

De hårda paranteserna används vid indexering. Till vänster om kommatecknet specificeras rader och till höger om kommateckent specificeras kolumner. Det koden egentligen gör ovan är att säga att vi vill ha alla rader förutom de 12 första och eftersom det inte står något till höger om kommatecknet så indikerar det att vi vill behålla alla kolumner i tabellen.

Efter detta bör variablerna eventuellt döpas om till mer användarvänliga namn så att de blir enklare att koda. Med användarvänliga menas namn som helst inte är för långa eller består av specialtecken och mellanslag (kom också ihåg att vissa ord är reserverade i R och bör inte användas som variabelnamn). Detta kan göras med hjälp av funktionen `colnames()` likt

```
colnames(min_df) <- c("namn1", "namn2")
```

Använder man sig av `ggplot()` kan en del variabelnamn rentav hindra en från att plotta sina variabler.

Alternativ 1: Definiera tidsvariabeln som Datum

Eftersom detta också är en tidsserie i månader så bör helst tids-kolumnen definieras som datum. Det kan göras med hjälp av funktionen `as.Date()`. Fördelen med att göra så är att det sedan blir enklare att illustrera variablerna grafiskt med lämpliga etiketter. SCB presenterar sina tidsserier i formatet av typ 1988M01 vilket i R tolkas som observationer av character-typ och kan vara svårt att känna igen som en tidsserie.
Det finns många olika sätt att komma runt det och definiera om kolumnen som datum. Ett sätt är att substituera bort `M` i textsträngen och sedan ersätta det med ett `-`. Detta görs nedan med funktionen `gsub()` som finns inbyggd i R och som kan hjälpa en att göra olika transformationer av textsträngar. Det kommer alltså göra så att varje observation får formatet ÅÅÅÅ-MM. Koden nedan visar hur det i princip kan se ut.

```
min_df$month <- gsub("[M]", "-", min_df$month)
```

R vill också ofta ha ett dagsdatum när datum ska definieras så vi lägger till den första i varje månad (för att göra det enkelt för oss) genom att använda oss av funktionen `paste0()`. Funktionen nedan klistrar ihop en befintlig textsträng med `-01` så att det nya formatet blir ÅÅÅÅ-MM-DD. Därefter är kolumnen färdigbearbetad och funktionen `as.Date()` kan användas där det första argumentet är kolumen med datum och det andra argumentet instruerar att året kommer först i textsträngen, följt av månaden och därefter av dagen.

```
min_df$month <- paste0(min_df$month, "-01")
min_df$month <- as.Date(min_df$month, "%Y-%m-%d")
```

Alternativ 2: skapa en ny variabel med observationsnummer

Om man inte definierar om sin tidsvariabel till datum så kan man istället göra en fuling och skapa en ny variabel som representerar antal observationer och sedan använda denna variabel som tid eller "x"-variabel när man vill plotta tidsserien. Det kan enkelt göras med koden:

```
min_df$x <- 1:nrow(min_df)
```

Där funktionen `nrow()` räknar hur många rader man har i sin tabell. Slutresultatet blir att en ny variabel kallad för `x` som går från 1, 2,…n läggs till i tabellen. Där n = antal observationer man har totalt i sin tabell.

Efter dessa bearbetningar är vi nu redo att plotta tidsserien.

- ☐ Plotta tidsserien i en passande graf, tänk på att tidsserier inte brukar presenteras grafiskt på samma sätt som tvärsnittsdata.
- ☐ Kommentera diagrammet. Är det några år som sticker ut? Vad tror du (kort) att det kan bero på?

Del 3 - Samla in data om arbetslöshet från SCB:s hemsida

Att samla in och bearbeta data är ofta en väldigt viktig och tidskrävande del när man arbetar med statistik, inte minst när man behöver skriva sin kandidatuppsats (även om de flesta här inte behöver oroa sig för det just ännu). Du ska nu ladda ner data genom att söka dig fram på SCB:s hemsida scb.se (alltså inte via `API`).

- ☐ Leta fram serien om arbetslöshet bland personer 15-74 år för åren 2001-Januari till 2023-Mars genom att gå in på SCB:s hemsida under ämnet `Arbetsmarknad` följt av `Arbetskraftsundersökningar (AKU)` och sedan `Arbetslösa från 2001`.
- ☐ Det finns flera serier, Klicka på månadsserien som sträcker sig ända från 2001 till och med Oktober 2023.
- ☐ Om din dators språkinställningar är på svenska så kan du fortsätta som vanligt men om den är på engelska så bör du helst använda SCB:s engelskspråkiga inställning. Om du inte vet så kan du alltid gå tillbaka i ett senare skede och ändra till SCB:s engelskspråkiga alternativ.
- ☐ Du kommer nu få möjligheten att skapa en egen tabell genom olika tillval. Välj först `Procent` i `tabellinnehåll`, följt av alternativet `arbetslösa` under arbetskraftstillhörighet och markera alla alternativ under fliken `kön`. Välj sedan `totalt 15-74` under `ålder` och markera sedan samtliga månader.

- ☐ Tryck på fortsätt. Det bör nu dyka upp en färdig tabell. Innan du laddar ner filen bör du först pivotera tabellen (medsols) för att göra datahanteringen enklare i R. Tryck på `Verktyg` och sedan spara resultat som.
- ☐ Ladda ner datasetet som excelfil, csv-fil (Kommaavgränsad utan rubrik) och textfil (Relationstabell som textfil) och spara det i samma mapp som din Quarto fil befinner sig i.
- ☐ Öppna textfilen och se efter ifall allt ser ut som det ska. Om tecknen ser konstiga ut, speciellt ifall svenska vokaler har omvandlats till andra tecken, så har du troligtvis en engelskspråkig inställning på din dator. Det enklaste lösningen är därför att ladda ner den engelska versionen av samma dataset från SCB (alla namn kommer då att vara på engelska).
- ☐ Lägg märke till att spara sina filer i `Onedrive` kan ibland orsaka problem vid just inläsning av filer i R.

Del 4 - Läs in data från excel-, csv- och textfiler

Del 4a - excel

Vi har tidigare tittat på hur man kan läsa in data i excelformat. För att läsa in excelfiler behöver man först ladda paket såsom exempelvis `readxl` (alternativt `openxlsx` som användes i SDA1, fast som använder lite annorlunda argument än nedan). Vill man läsa in data från en excelfil kan man använda sig av koden:

```
df <- read_excel("Min_df.xlsx", sheet = 1, col_names = TRUE, skip =  0)
```

Även om du inte har tillgång till officepaketet så bör det gå att läsa in filen i R. SU erbjuder även alla studenter gratis tillgång till `office 365` (har man Linux så kan det dock finnas större begränsningar tyvärr).

Alternativt om man inte har sin fil sparad i samma mapp som ens Quarto-dokument behöver man ange hela sökvägen och inte bara namnet och formatet på filen exempelvis likt:

```
df <- read_excel("C:/Users/Mitt namn/Min mapp/Min_df.xlsx", sheet = 1,
                 col_names = TRUE, skip =  0)
```

Där argumentet `col_names = TRUE` innebär att den första raden används som variabelnamn och `skip` hoppar över så många rader som man har specificerat (default är 0).

Alternativt med paketet `openxlsx` kan man istället använda sig av funktionen:

```
df <- read.xlsx("Min_df.xlsx", sheet = 1, colNames = TRUE, skipEmptyRows = TRUE)
```

Bra att veta är att funktionen `read_excel()` omvandlar automatiskt din dataframe till en tibble.

Om excelfilen kommer direkt från internet utan att man har öppnat den tidigare så kan funktionen `read.xlsx()` krångla. En lösning är då att öppna excelfilen och trycka på redigera följt av spara. Detta funkar dock bara ifall man har tillgång till officepaketet. Har man inte det så går det smidigare att istället använda sig av funktionen `read_excel()` från paketet `readxl`.

- ☐ Läs in excelfilen i R med hjälp av `read_excel()`. Ta också gärna bort de första tomma raderna när den läses in och spara tabellen med namnet `AB_excel`.
- ☐ Ser tabellen ut som du hade förväntat dig? Skulle man behöva göra några ändringar i den? Vilka då i så fall? (Du behöver inte göra några ändringar i den om du inte vill, exemplet är endast till för att illustrera hur en rå, nedladdad fil kan se ut).
- ☐ Ifall du hinner: Testa också att läsa in datasetet med paketet `openxlsx`. Försök sedan justera argumenten i funktionen `read_excel` eller `read.xlsx` och beabeta ditt data så att det skulle bli mer funktionellt för en eventuell analys.

Del 4b - csv

För att läsa in filer av csv format (som står för comma separated values) behöver man inte ladda något paket. Istället kan man använda sig av den inbyggda funktionen i R `read.csv()` exempelvis med koden:

```
df = read.csv("Min_df.csv", header = TRUE, sep = ",", dec = ".")
```

Detta är default argumenten där det första argumentet anger sökvägen till filen. På liknande sätt som med excelfiler så räcker det med att man endast anger namnet på filen ifall den redan befinner sig i samma mapp som Quarto-filen. Det andra argumentet `header` säger till att den första raden i tabellen ska vara variabelnamn. Argumentet `sep=","` anger att variablerna separeras av komma och slutligen så anger `dec="."` att decimaler markeras med punkt. Lägg märke till att detta endast är default värden för `sep` och `dec` som kan tillskrivas andra tecken istället.

- ☐ Läs in csv-filen i R genom att använda dig av funktionen `read.csv()` och spara den under namnet `AB_csv`. Klicka på tabellen i `Environment`. Vad händer ifall man istället skriver `sep = ";"` i funktionen `read.csv()`?

Del 4c - txt

Textfiler är ofta väldigt enkla att använda sig av i olika programmeringsspråk. För att läsa in filer av txt.-format behövs heller inget paket. Man kan exempelvis använda sig av funktionen `read.delim()` alternativt av `read.table()` I R.

```
df <- read.delim("Min_df.txt", header = TRUE, sep = "\t", dec = ".")
```

Det första argumentet är filens sökväg. Precis som med excel- och csv-filer räcker det med att man endast skriver namnet på filen följt av en punkt och vilket format den har i en textsträng om filen finns sparad i samma mapp som Quarto-dokumentet. Det andra argumentet används precis som innan för att ange att den första raden ska användas som variabelnamn. Default argumentet `sep = "\t"` anger att variablerna i tabellen ska separeras med "tab" (stort mellanrum, ofta består ett tab av 4 vanliga mellanslag) och det sista argumentet som visas ovan `dec = "."` instruerar precis som innan att decimaler markeras med punkt.

- ☐ Läs in textfilen i R genom att använda dig av `read.delim()` funktionen och spara tabellen under namnet `AB_txt`. Ta en titt på tabellen. Kontrollera att dina variabler är definierade som de borde med `str()`-funktionen. Vad skulle hända med din tabell ifall du istället skrev `sep = " "` i funktionen `read.delim()`?
- ☐ Läs nu istället in csv-filen igen med funktionen `read.delim()`. Vilka ändringar behöver du göra i funktionen så du får önskat utseende på tabellen och att variablerna blir definierade så som de borde?

Del 5 - Samla in data om arbetslöshet med hjälp av API

I den här deluppgiften ska du ladda ner data om arbetslöshet direkt från R via API med hjälp av funktionen `interactive_pxweb()` istället för att gå in på SCB:s hemsida.

- ☐ Använd funktionen `interactive_pxweb()` för att ladda ner samma dataset som du fann i Del 3. Följ instruktionerna som motsvarar de fyra första punkterna i Del 2a. Men välj sedan `Arbetsmarknad` i listan som dyker upp. Försök sedan att identifiera den relevanta serien utifrån instruktionerna som gavs i Del 3.
- ☐ Ladda ner alla observationer och spara dem i en dataframe på liknande sätt som du gjorde med KPIF i del 2.
- ☐ Döp om tabellen till `df_AB` eller något annat lämpligt och gör eventuella passande namnbyten för variablerna samt definiera om datatypen för månadsvariabeln till datum-typ om så önskas.
- ☐ Illustrera tidsserien grafiskt och kommentera grafen. Kan man se några trender eller eventuella säsongsmönster? Finns det någon skillnad mellan arbetslösheten bland män och kvinnor?

Del 6 - Ladda ner och presentera data över civilstånd

I den här deluppgiften ska du ladda ner data om civilstånd för åren 1998-2022.

- ☐ Använd funktionen `interactive_pxweb()` för att påbörja navigeringen i SCB:s databas via API. Följ instruktionerna som motsvarar de fyra första punkterna i Del 2a. Men välj sedan `Befolkning` i listan över ämnesområden, följt av `Befolkningsstatistik`. Välj sedan ut `Partnerskap` i listan och därefter `Partnerskap efter region, civilstånd och kön` och sedan hela riket. Välj sedan alla 3 alternativ (registrerade-, separerade- och efterlevande partners) och därefter både män och kvinnor. Ta med samtliga år i din tabell och ladda ner data.
- ☐ Byt eventuellt namn på några av variablerna efter att ha laddat ner datasetet.
- ☐ Undersök ifall alla variabler har korrekt datatyp genom att använda dig av funktionen `str()`. Är de numeriska variablerna definierad som numeriska? Om inte så kan man använda sig av funktionen `as.numeric()` för att definiera om en variabel.
- ☐ Agggregera grupperna män och kvinnor och spara resultatet i en ny tabell så att endast totalvärdet (av män och kvinnor) finns med i den nya tabellen och plotta sedan de tre olika kategorierna registrerade-, separerade- och efterlevande partners tillsammans över tid i ett lämpligt diagram. Detta kan vara lite klurigt men tänk på att du vill ha kvar år, och civilståndsvariabeln (med de tre olika kategorierna) samt hur många det är totalt registrerade i den kategorin för det året. För att aggregara data kan man använda sig av funktioner i tidyverse eller alternativt `aggregate()` som finns inbyggd i R. I tidyverse kan man använda sig av funktionen `group_by()` och därefter `summarise()` likt nedan. Tänk på att variabel 1 och 2 bör motsvara civilstånds- och års-variabeln och variabel 3 bör motsvara partnerskapsvariabeln. Det sista argumentet `.groups = "drop"` används för att exkludera varningsmeddelande som annars kan printas ut.

```
min_df <- min_df %>%
  group_by(variabel_1, variabel_2) %>%
  summarise(variabel_3 = sum(variabel_3), .groups = "drop")
```

Om man använder sig av funktionen `aggreagate()` kan man istället skriva

```
min_df <- aggregate(variabel_3 ~ variabel_1 + variabel_2, data = min_df, sum)
```

Där det sista argumentet säger vilken typ av funktion som ska användas, i detta fall kommer alltså män och kvinnor att summeras och endast års-, civilstånds och parnerskapsvariabeln vara kvar i den nya tabellen. Efter dessa steg kan man enkelt plotta de tre kategorierna över tid i samma graf med hjälp av exempelvis `ggplot()`.

- ☐ Kommentera grafen. Vad tror du hände mellan 2005-2010 och varför tror du att grafen ser ut så som den gör?

Inlämningsuppgift - Del A

Varje grupp ska skicka in ett html- eller ett pdf-dokument samt även skicka in en qmd-fil för del A av inlämningsuppgiften. Om man vill kan man fortsätta arbeta med del B i samma dokument men det är tillåtet att ha separata dokument för del A och B om så önskas. **Läs igenom och undersök innan filerna skickas in ifall alla figurer och eventuella tabeller syns och att ingenting annat saknas.**

Instruktioner

I den här delen av inlämningsuppgiften ska ni undersöka och jämföra populationsutvecklingen bland män och kvinnor i en specifik kommun. De 20 största kommunerna i Sverige sett till ytan är

- Kiruna
- Jokkmokk
- Gällivare
- Gotland
- Arjeplog
- Härjedalen
- Strömsund
- Skellefteå
- Villhelmina
- Örnsköldsvik
- Åre
- Storuman
- Pajala
- Sorsele
- Älvdalen
- Krokom
- Berg
- Arvidsjaurs
- Lycksele

- Norrtälje

Varje grupp har blivit tilldelad en kommun (se Athena för detaljer) och ska undersöka hur befolkningen, både bland män och kvinnor, har förändrats över tid. Data går att finna på SCB:s statistikdatabas och ska laddas ner med hjälp av API. Det finns flera olika tidsserier som involverar befolkningsstatistik.

- ☐ Använd dig av funktionen `interactive_pxweb()` i consolen (ej i din code-chunk) för att navigera i databasen och finna det relevanta datasetet.
- ☐ Välj ämnet `Befolkning` bland de listade kategorierna som dyker upp och sedan `Befolkningsstatistik`. Välj seda ämnet `Folkmängd` och därefter serien `Folkmängden den 1 november efter region, ålder och kön. År 2002 - 2023`. Ta endast kommunen som er grupp blivit tilldelad följt av den totala åldern och slutligen både män och kvinnor. Välj därefter alternativet `folkmängd den 1 november` och sedan samtliga år i listan. Kopiera och klistra in koden som använts för att hämta ut tabellen i Quarto när allt är klart. Glöm inte att också kopiera koden som används för att citera källan.
- ☐ Om nödvändigt, bearbeta gärna tabellen så att den blir mer användarvänlig (exempelvis byt namn på variabler osv). Detta steg är mest till för att hjälpa er. Tänk på att om ni vill använda er av `ggplot()` kommer det bli enklast att plotta variabler med önskade namn och etiketter ifall de redan har fått dessa i tabellen, istället för att behöva ändra på namn och etiketter i efterhand i själva plotten.
- ☐ Presentera de två tidsserierna (män och kvinnor) över folkmängden i samma graf och kommentera utvecklingen för båda tidsserierna var och en för sig. Kommentera även eventuella skillnader och likheter mellan dem. Vad kan ni se för tendenser över tid?
- ☐ Skapa en ny rubrik **Referenser**, eller något annat passande, och kopiera och klistra in referensen till det nedladdade datasetet under denna rubrik. Klistra inte in den delen av referensen som avses för källhänvisning i LaTeX om ni inte tänker använda LaTeX referenshantering i ert dokument. (Se övning 2a i datorlaborationen ovan för ett exempel).

Viktigt! Glöm inte att kommentera er graf.