Brukermanual for microdata.no

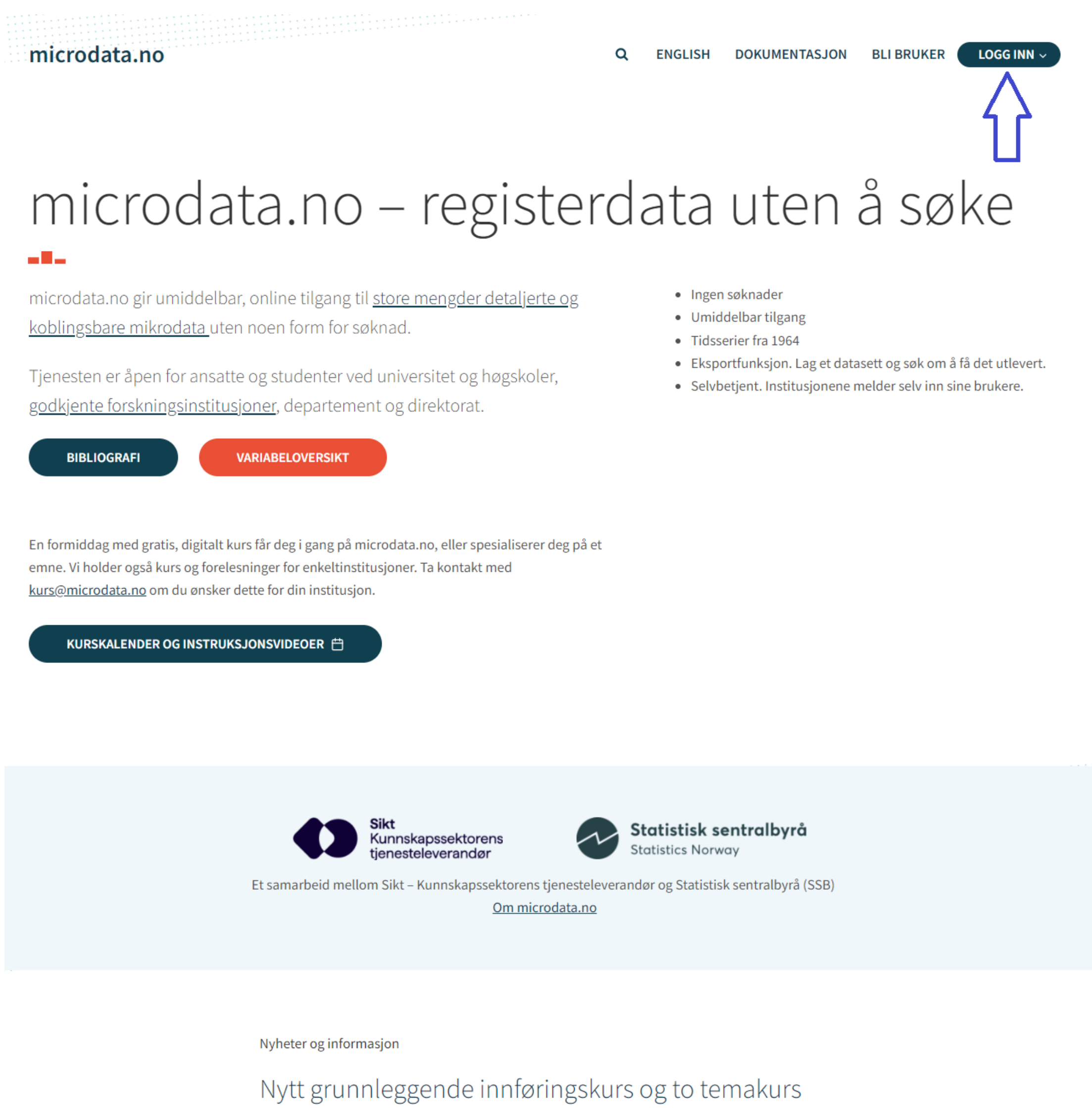
Skrevet av Trond Pedersen, Sikt

Innholdsliste

1. Om brukergrensesnittet

Microdata.no er et nettbasert analysesystem som baserer seg på en Stata-liknende kommandostruktur[1]. Det anbefales å bruke nettlesere som Chrome og Firefox for best mulig brukeropplevelse. Internet Explorer vil kunne gi diverse feil som at skjermen blir blank og/eller at det ikke går an å logge seg inn, og anbefales derfor ikke.

Innlogging i microdata.no gjøres via følgende nettside: http://microdata.no/



[1] Kommandoene er implementert ved bruk av Python og Pandas, og syntaksen likner på Stata. Dette for å gjøre det mest mulig brukervennlig.

Det som møter deg etter innlogging er en nettside bestående av et analyseområde, dvs. kommandovinduet (se kapittel 1.1). Dette brukes til å utforske variabler og teste ut kommandokjøringer:

En kan klikke på variabler for å gjøres seg kjent med innholdet, verdiformat, gyldighetsperiode m.m. Gjennom kommandoer som kjøres enkeltvis, kan variabler importeres til egne datasett for videre bearbeiding og analyser. I tillegg til deskriptive statistiske muligheter, kan en dessuten foreta avanserte statistiske operasjoner som regresjonsanalyser m.m. Se kapittler 1.1-1.4 for mer informasjon om hvordan jobbe i kommandovinduet.

Etter å ha utforsket og gjort seg kjent med data en ønsker å benytte i analyser, anbefales det på det sterkeste å benytte skript-funksjonaliteten for å systematisere analysearbeidet. Dette gjelder særlig om en har til hensikt å foreta en mer omfattende analyse (utover å bare lage enkel deskriptiv statistikk for et fåtall variabler). Bruk av skript har mange fordeler fremfor å utelukkende jobbe i kommandovinduet gjennom bruk av enkeltkommandoer:

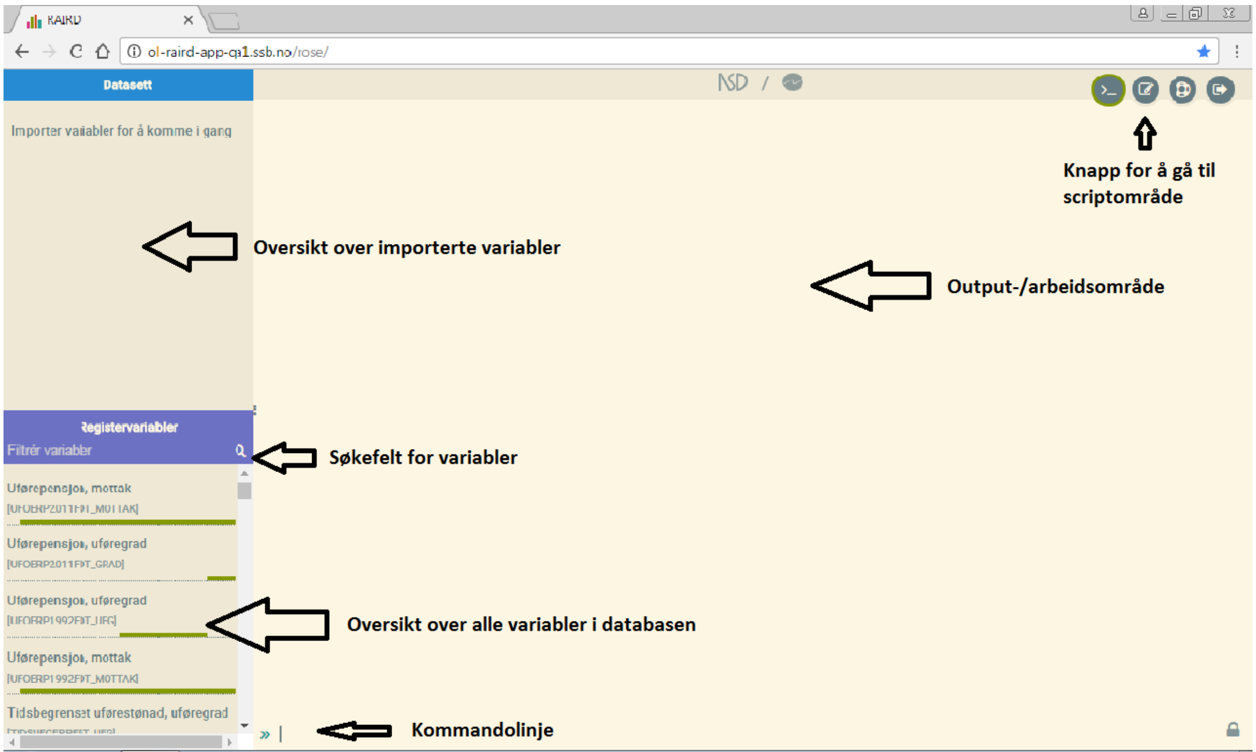
a) En kan sette opp kommandosekvenser som kan kjøres i én operasjon. Resultatet av skriptkjøringer vises fortløpende etterhvert som kommandoer kjøres, og kjøringen stoppes dersom feil oppdages i skript (syntax- eller logiske feil)
b) Kommandosekvenser kan redigeres og kjøres på nytt
c) Mye enklere å identifisere hvor eventuelle feil oppstår i en lang rekke med kommandosekvenser
d) Fungerer som dokumentasjon/logg av arbeidet
e) Forskjellige arbeidsøkter kan lagres med egne navn og enkelt kalles frem fra en meny
f) Innholdet i et skript kan kopieres over til egne dokumenter for ekstra backup

Arbeid en allerede har utført i kommandovinduet kan enkelt overføres til et skript for videre redigering. Det kan gjøres på tre måter:

- I skriptvinduet finnes det en menyknapp helt oppe til venstre. Der kan en velge “Nytt skript med historikk fra kommandolinjen”. Husk å legge inn et navn på skriptet i linjen over skriptvinduet. Skriptet vil da lagres med dette navnet.
- I kommandovinduet kan en skrive kommandoen `history`. Dette frembringer alle kommandoer en har kjørt i en enkelt liste som kan kopieres på vanlig måte ved å klikke på kopieringsknappen som vises når musepeker holdes over. Trykk så på ctrl+c og lim inn i et tomt skriptvindu. Husk også her å lage navn på skriptet, jfr. punktet over
- I kommandovinduet kan en skrive kommandoen `save` etterfulgt av et valgfritt navn som du bruker på skriptet. Navnet må ha fnutter rundt. F.eks. `save ‘Analyse av arbeidsledige’`. Ved denne fremgangsmåten trenger du ikke lage navn på skriptet i skriptvinduet etterpå

For mer informasjon om bruk av skript, se kapittel 1.5.

1.1 Kommandovinduet

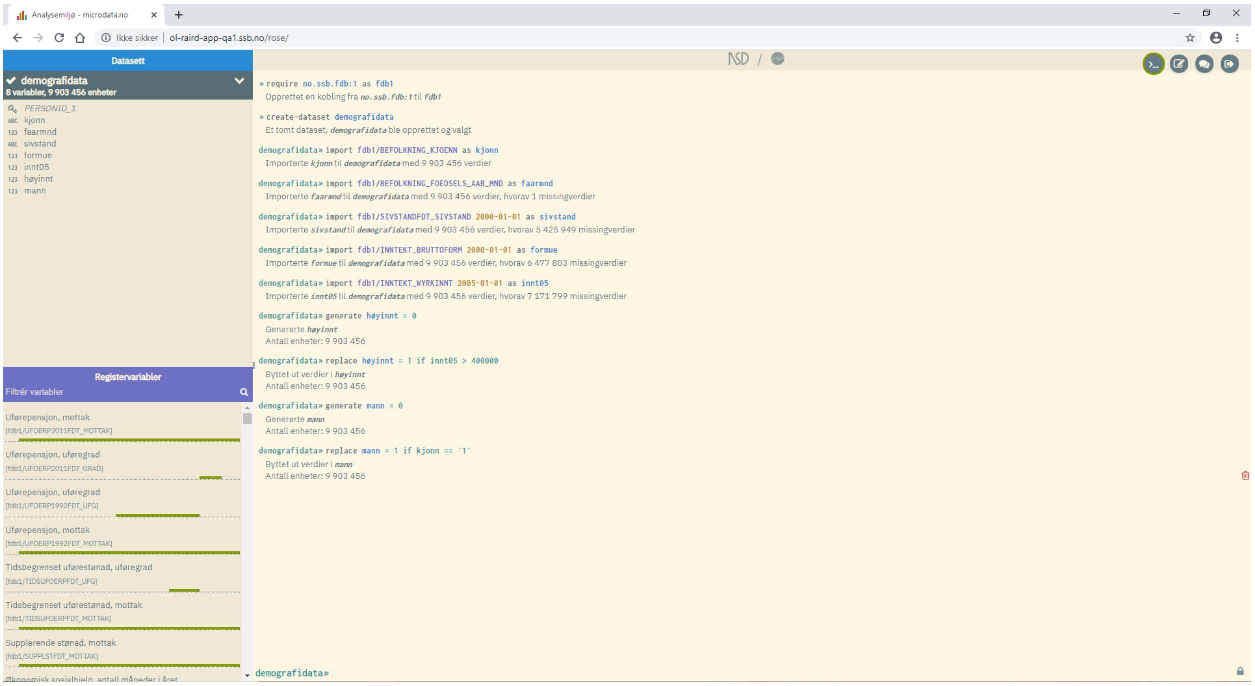


Kommandovinduet består av følgende:

- Arbeidsområdet (største feltet helt til høyre)
- Oversikt over tilgjengelige variabler (feltet nederst til venstre - feltet vil være blankt helt til en kobler seg mot databanken vha. kommandoen `require`, jfr. kapittel 2.1)
- Oversikt over variabler importert til egne datasett (feltet øverst til venstre)
- Kommandolinjen (nederst i arbeidsområdet)

Egne datasett bygges opp ved å importere, tilrettelegge og utvikle nye variabler basert på variabler fra datakatalogen. Datasett blir lagret i brukerens kommandovindu og forsvinner ikke uten at brukeren selv sletter dem. Se kapittel 2 for hvordan en oppretter datasett og importerer variabler.

Etter at kommandovinduet er blitt fylt opp med importerte variabler, kan det se slik ut:



I eksempelet over er det opprettet et datasett med navnet “demografidata” som har 8 variabler og 9 903 456 enheter (individer). Blant variablene finner en identifikasjonsnøkkelen PERSONID_1. Dette er en systemvariabel som alltid følger med ved import av variabler. Den angir en unik personidentifikator som brukes som koblingsnøkkel. Systemet kobler automatisk sammen variabler (“left join”) dersom en bare skal bruke data på personnivå, og da trenger en ikke forholde seg til denne variabelen[2].

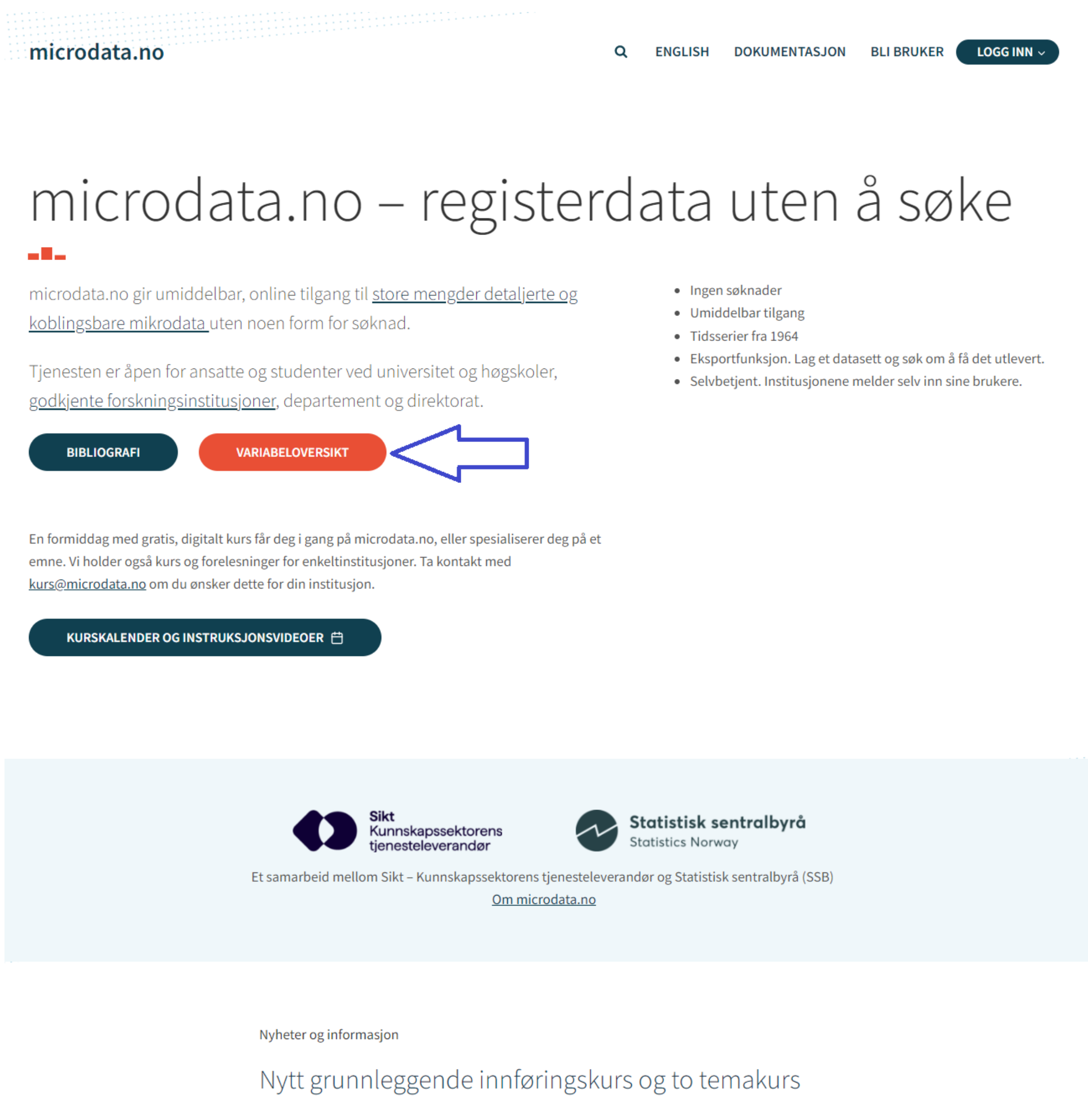
Arbeidsområdet viser en logg for hvilke kommandoer som har vært utført og hvilket svar man får, det være seg tabeller, figurer og annen feedback.

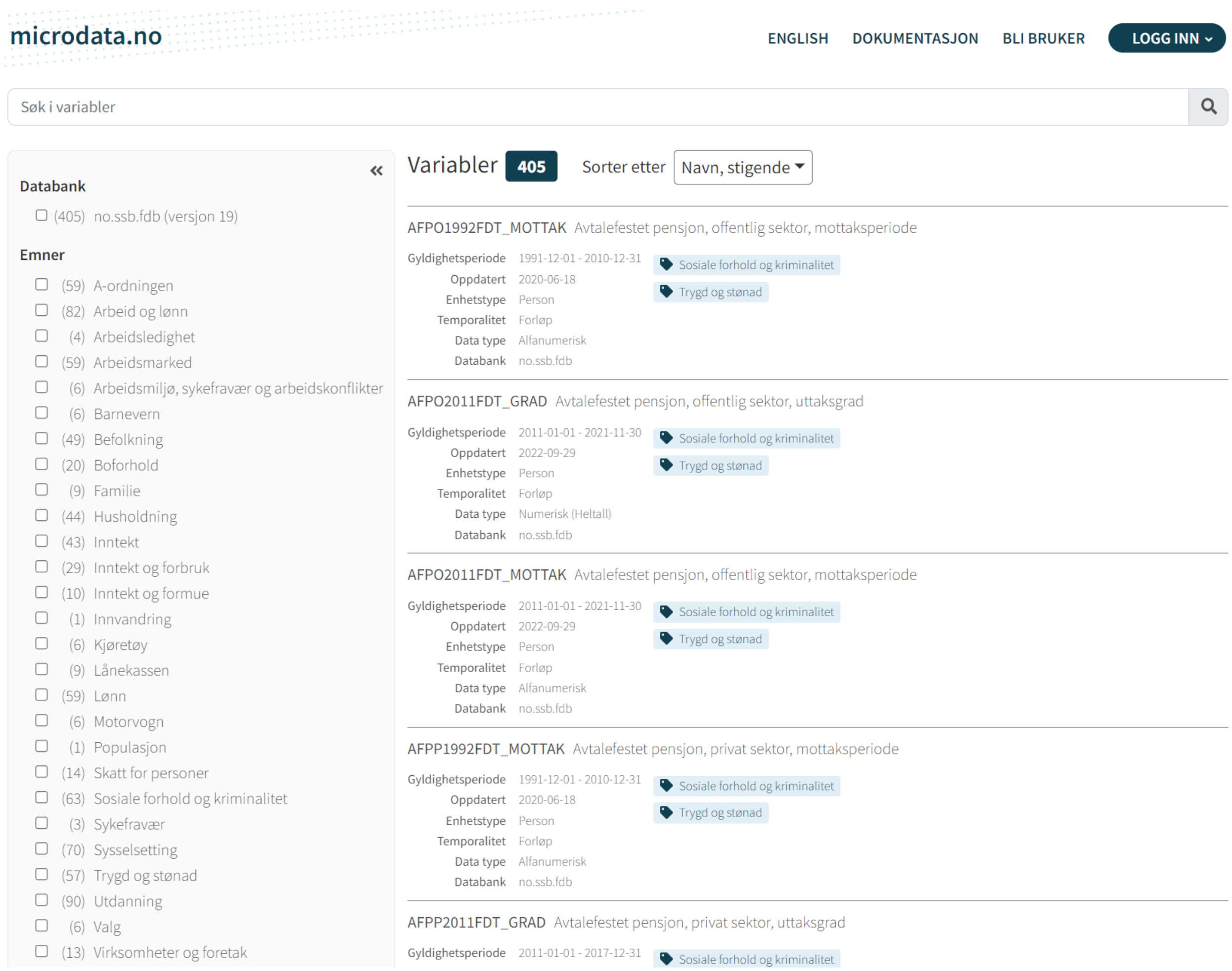
[2] Identifikasjonsnøkkelen PERSONID_1 trenger bare angis eksplisitt i de tilfeller hvor en bruker kommandoen `collapse` til å aggregere opplysninger fra et sub-individnivå opp til individnivå. Et typisk eksempel er når en bruker kommandoen `import-event` til å lage et datasett med “hendelser” som enhetstype (se kapittel 2.3.2). I praksis er dette datasett som kan inneholde flere verdimålinger per individ representert ved separate datarecords som peker til de ulike verdiene (en importerer alle verdimålinger mellom to tidspunkt for en gitt variabel), og som ikke uten videre kan kobles sammen med andre datasett. For å kunne koble sammen data basert på hendelsesopplysninger (`import-event`) med vanlige individnivå-datasett, må en bruke kommandoen `collapse` til å aggregere dataene opp til individnivå og deretter bruke kommandoen `merge` (se kapittel 2.8 og 3.4). Et annet eksempel på sub-individdata er såkalte kursvariabler (data om pågående studier) som kan inneholde flere verdimålinger per individ selv på tverrsnittsnivå (`import`), siden det er mulig å ta flere studier på samme tid.

Helt nederst står det nå “demografidata>>” i stedet for “>>”, for å markere at en nå står i datasettet “demografidata”. Om en oppretter flere datasett vil vinduet øverst til venstre inneholde flere variabeloversikter tilsvarende den som gjelder for “demografidata”. For å arbeide på ulike datasett kan en skifte gjennom å bruke kommandoen use <datasett>. Da vil ledeteksten i kommandolinjen endre navn til det datasettet en har forflyttet seg til.

1.2 Variabler og variabelbeskrivelser

Analysesystemet microdata.no har en lang rekke demografiske, utdanningsrelaterte, økonomiske, sysselsettingsrelaterte og trygderelaterte variabler i databanken. Disse kan en få en oversikt over gjennom å utforske variabeloversikten en finner på åpningssiden (https://microdata.no/discovery):

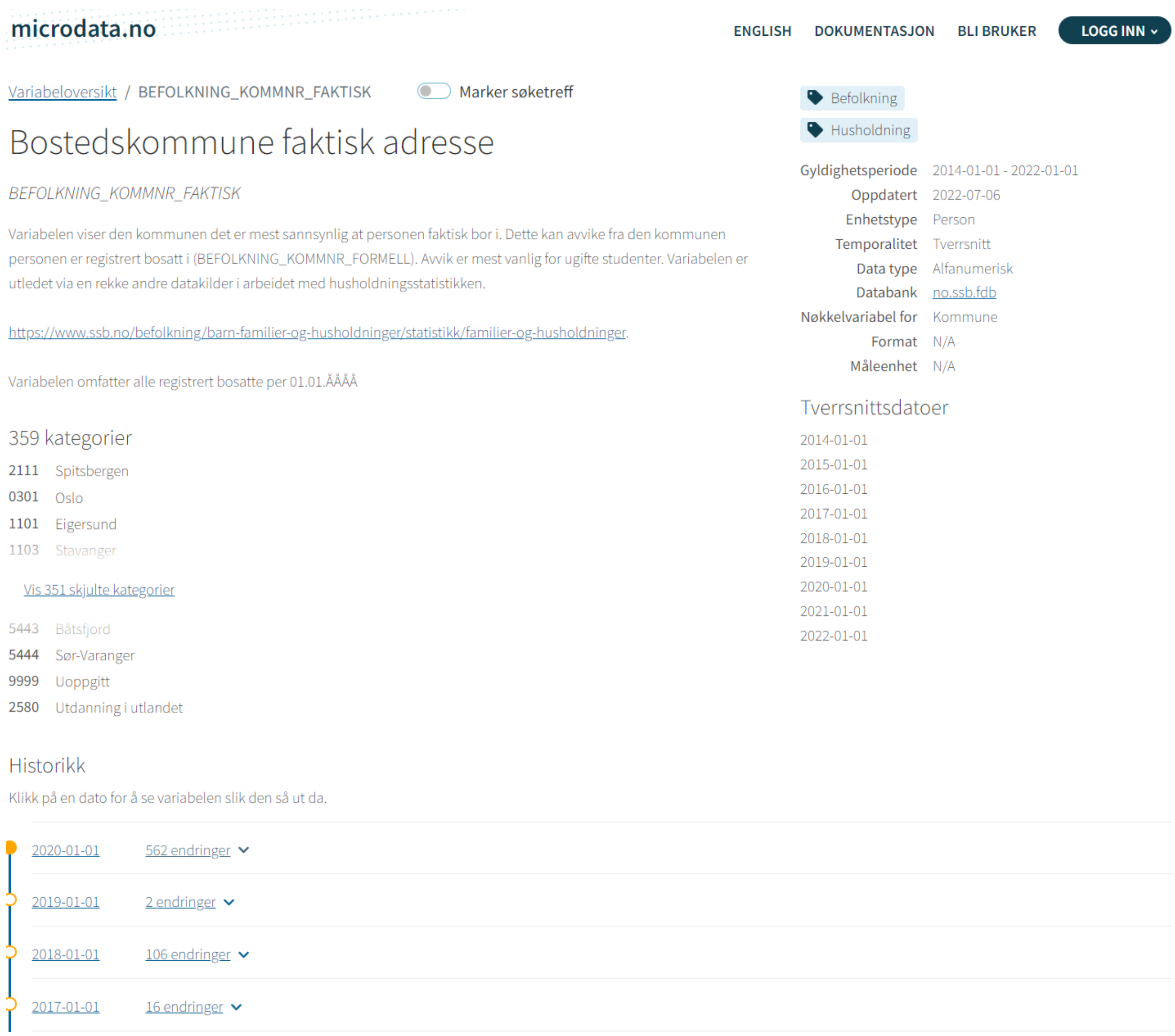




Variabellisten viser en oversikt over samtlige variabler i databanken (405 i SSBs databank per desember 2022), og en kan bruke søkefunksjonalitet for å lettere kunne finne variabler en leter etter.

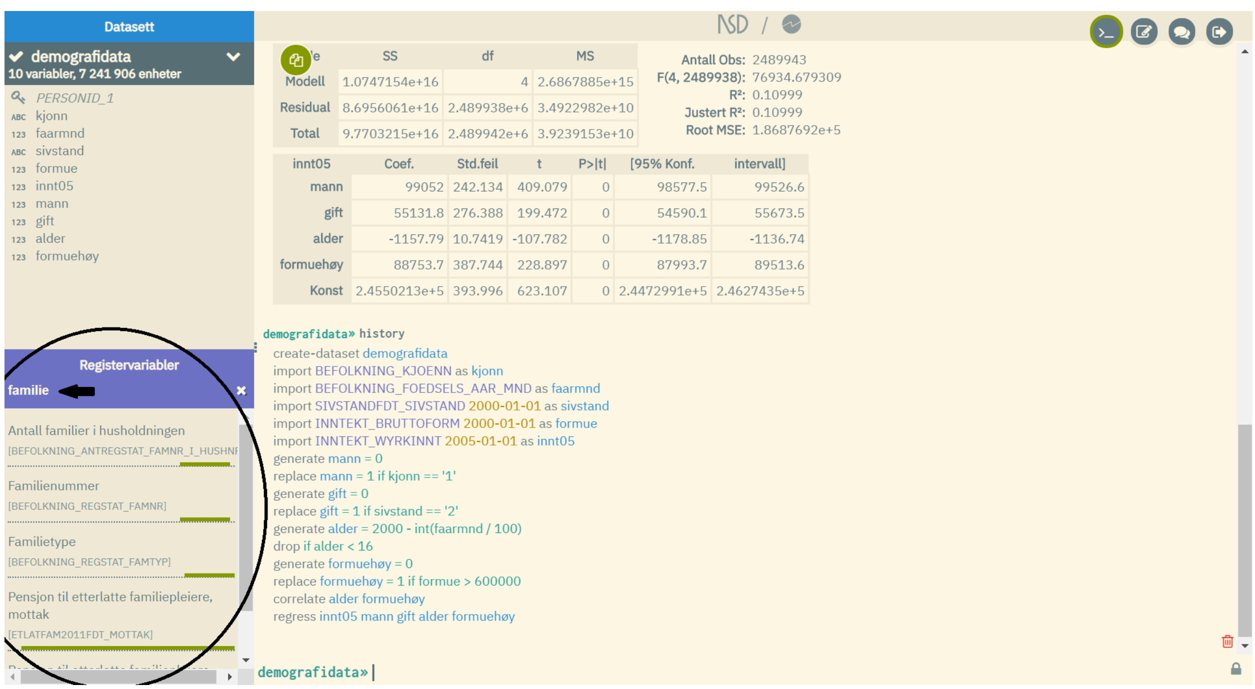
Det er også mulig å filtrere søk ved å huke av på ønsket databank, emneområde, datatype, enhetstype, nøkkelvariabel og/eller temporalitet.

Ved å klikke på en variabel i variabellisten, får en opp definisjoner, kodelister, endringshistorikk og annen nøkkelinformasjon knyttet til den spesifikke variabelen:



Merk at kodelisten for en gitt variabel endres over tid. Dette må en ta hensyn til dersom en jobber med lengre tidsserier som strekker seg bakover i tid. Det er nemlig gjeldende kodeliste for det aktuelle tidspunktet en må forholde seg til. Spesielt kommunekoder, utdanningskoder og næringskoder har relativt hyppige endringer i kodelistene. Ved å klikke på punktene foran tidspunktene som angir start for en ny kodeversjon, får en opp gjeldende kodeliste for den aktuelle tidsperioden. Om en klikker på angivelsen av antallet endringer, får en opp en oversikt over hvilke endringer som er blitt gjort for samme tidsperiode:

Som illustrasjonen nedenfor viser, finnes det også en fullstendig variabeloversikt nederst til venstre inne i selve kommandovinduet. Også her kan en filtrere variabellisten ved å skrive inn deler av et variabelnavn i søkefeltet. Som for variabelisten en finner på åpningssiden, fungerer søket både mot variabelbeskrivelse og selve navnet. På den måten blir det lettere å finne variabelen en ønsker.



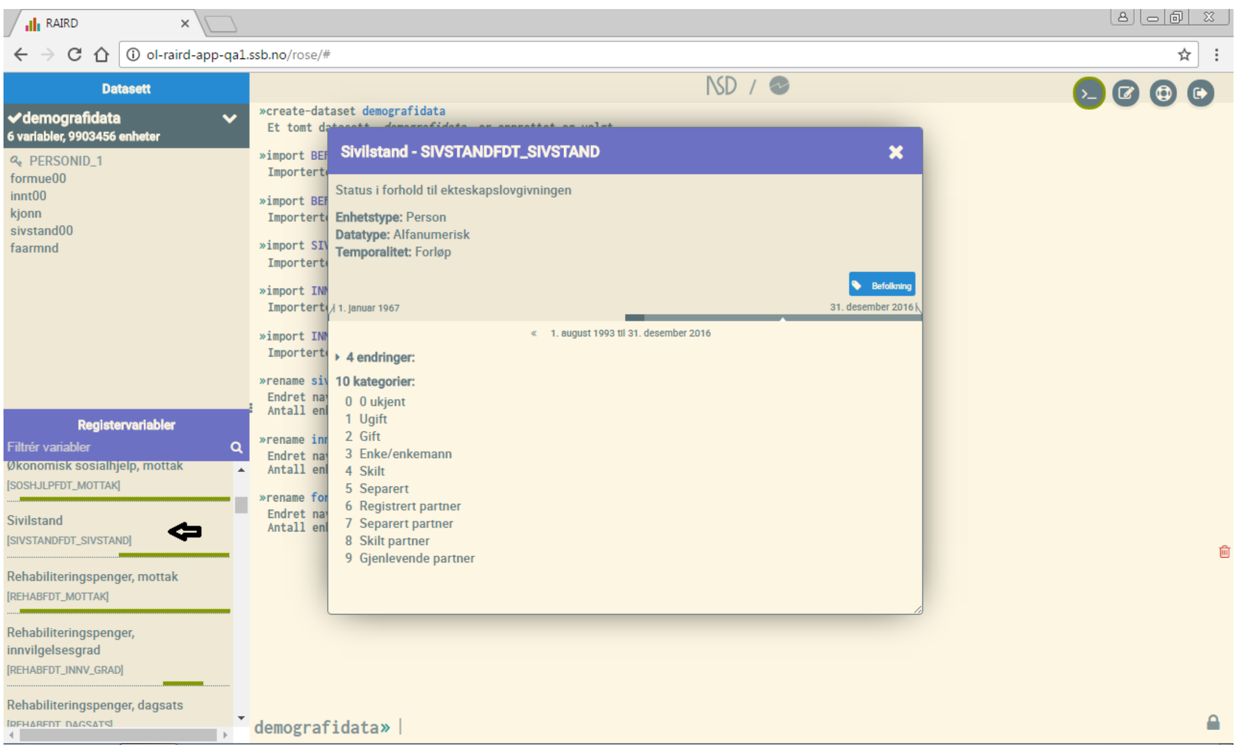
Alle variabler vises med en tilhørende tidslinje som markerer gyldighetsperioden(e), dvs. hvilket tidsspenn som er dekket. Variabler i microdata.no er tredimensjonale - de inneholder tid. Ved å “klikke” på variabler i listen kan en hente frem deskriptiv statistikk og annen informasjon som variabeltype o.l.

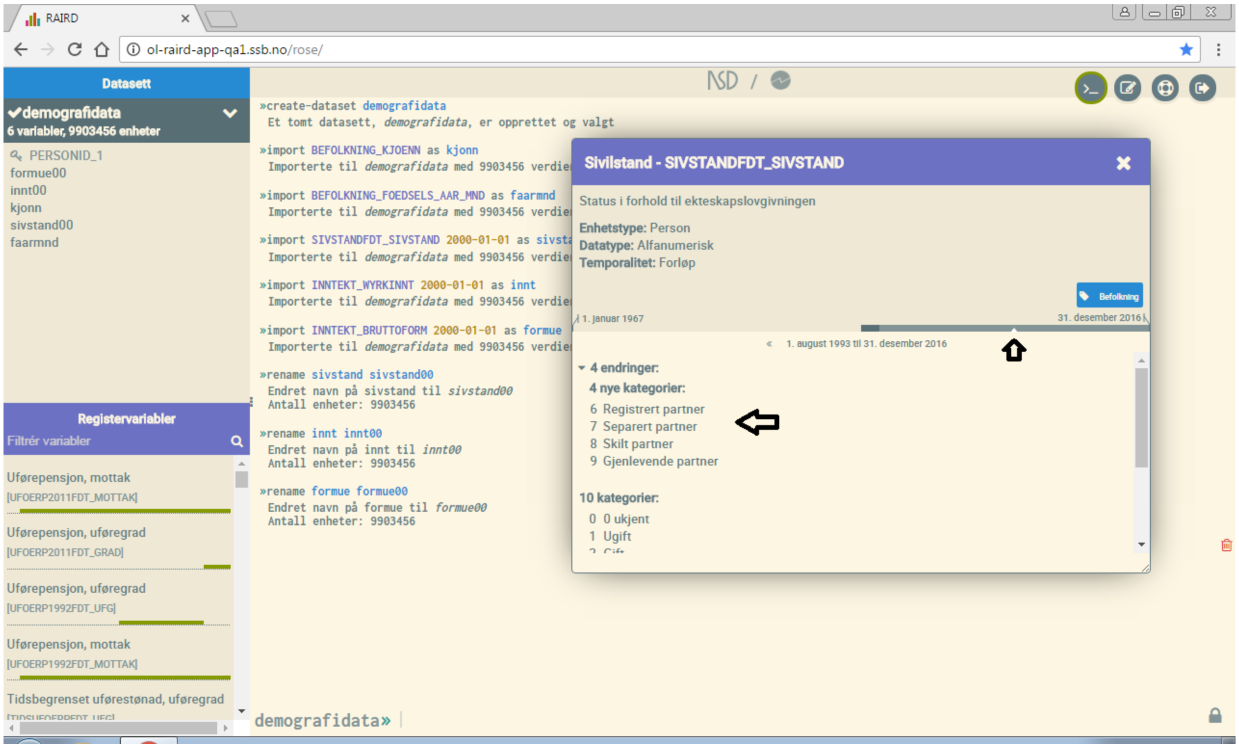
I eksempelet nedenfor blir dette eksemplifisert for variabelen “Sivilstand”. Variabelen presenteres da i et eget vindu som en kan flytte på og justere. Det gir detaljert informasjon om variabelen:

- Nøkkelinformasjon: Variabelnavn, variabel-label, variabelbeskrivelse, variabeltype

- Detaljert interaktiv tidslinje som gir mulighet til å studere endringer i kodingen over tid: Endringer i kodingen vises gjennom ulike farger som illustrerer hvilke tidsperioder de gjelder for. Klikker en på de ulike feltene i tidslinjen får en frem en liste over de kodene som var gyldige i den aktuelle perioden. I eksempelet er det markert i feltet som gjelder 1. august 1993 - 31. desember 2016, og det dukker da opp en liste på 10 kategorier

- Informasjon om endringer: I eksempelet vises det “4 endringer”. Dette er antallet endringer i forhold til forrige tidsperiode. En kan klikke på “4 endringer”, og få opp en liste over kodene som er nye



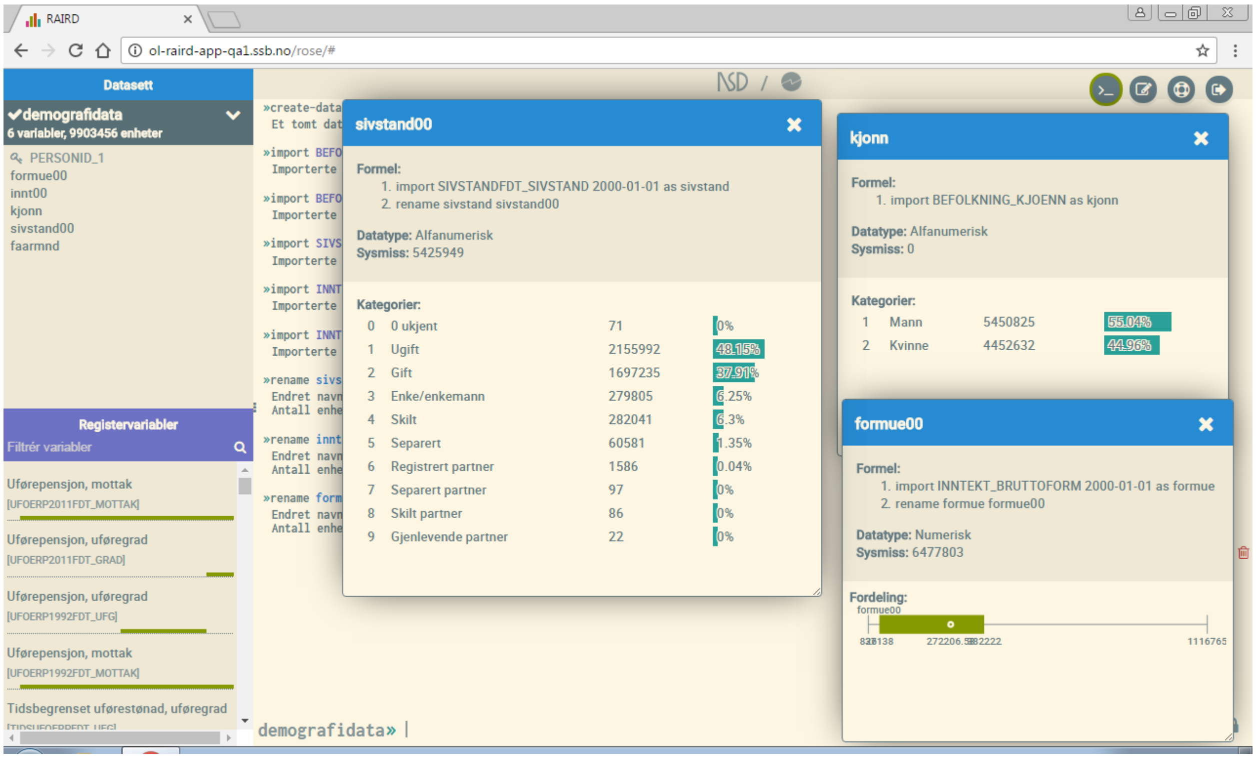


For variabler som er importert til brukerens datasett ("demografidata") vises det en litt annen type informasjon som kan være nyttig når en arbeider med mange ulike variabler. Denne informasjonen justerer seg fortløpende dersom en endrer på variablene, og vises i separate popup-vinduer når en klikker på variabler i listen tilhørende det aktuelle datasettet ditt:

- Formel: Øverst i vinduet finner en "skapelseshistorikken" til den aktuelle variabelen. Dette brukes til å slå opp hvordan en variabel har blitt laget eller omkodet

- Nøkkelinformasjon: Variabeltype og antall enheter som har verdi for manglende data (sysmiss)

- Frekvensfordeling og enkel statistikk: For kategoriske variabler vises frekvensfordelingen, mens det for kontinuerlige variabler vises en standard boksplot med boks for de to midterste kvartilene, gjennomsnitt og minimums- og maksimumsverdi (såkalte whiskers). Om verdier blir uleselige pga. overlapp, så kan dette popup-vinduet utvides.



1.3 Kommandolinjen

Kommandolinjen er den sentrale delen av brukergrensesnittet, og det er der en skriver inn kommandoer for hva en ønsker å gjøre. Dette omfatter import av variabler, lage deskriptiv statistikk for enkeltvariabler, kommandoer for bearbeiding og omkoding av variabler, administrative hjelpekommandoer (`help`, `history`, `clear` etc) eller analysekommandoer. Se vedlegg A for fullstendig liste over tilgjengelige kommandoer.

Analysesystemet microdata.no har en innebygget selvutfyllingsløsning som foreslår relevante kommandoer ut fra hva en starter å taste inn. Ofte er det nok å taste inn noen få tegn før systemet foreslår den ønskede kommando. Ved å trykke på <tab>-tasten på tastaturet, legges kommandoen korrekt inn.

De fleste kommandoer forutsetter at det legges inn tilleggsinformasjon, og selvutfylling fungerer også i slike sammenhenger. Kommandoen `import` trenger informasjon om variabelnavn og måletidspunkt for å kunne utføres. Ønsker en å importere variabelen *kjønn*, så kan en taste inn bokstaven “k”. Det vil være nok til å få opp en liste over alle variabler med bokstaven “k” i navnet - i praksis blir det svært mange. Fortsetter en med “j” vil systemet liste alle variabler med “kj” i navnet. Etter å ha tastet inn "ø" vil bokstavkombinasjonen være såpass unik at systemet har redusert antall alternativer til et fåtall, og brukeren kan velge den korrekte variabelen med piltastene på tastaturet og deretter bruke *<tab>*-tasten. Når variabel er valgt må man også angi et tidspunkt gitt at ikke det er snakk om en konstant opplysning slik som "kjønn" (da trenger en ikke angi tidspunkt). Default-verdi for systemet er den sist brukte datoen. Om dette er ok, brukes *<tab>*-tasten. Hvis ikke legger en inn aktuell ny dato i stedet. Systemet vil også foreslå aktuelle opsjoner til slutt. For `import` kan en bruke opsjonen `as.` Den brukes til å lage et alias til den aktuelle variabelen. Registervariablene i databanken har ofte litt upraktiske og lange navn som med fordel kan døpes om gjennom `as`- opsjonen for bl.a. å skape mer leselige analyse- og statistikkutskrifter.

1.4 Nyttige kommandoer

Om en lurer på hva slags kommandoer en kan bruke, kan en starte med `help.` Da listes alle tilgjengelige kommandoer. For dem som kjenner til analyseprogrammet Stata, vil de fleste kommandoene virke velkjente. Analysesystemet microdata.no benytter en Stata-liknende syntax, det innebærer også at kommandoer skrives på engelsk.

Ønsker en full informasjon om enkeltkommandoer, taster en inn `help` og deretter navn på kommando, f.eks. `help import.` Da vises en forklaring med eksempler på bruk. En komplett oversikt over kommandoer og funksjoner vises i vedlegg A og B.

En annen nyttig kommando er `history.` Den lister opp alle kommandoer som har vært brukt i en sesjon. Denne listen kan kopieres inn i systemets skripteditor for en samlet kjøring av alle kommandoene, jfr. kapittel 1.5, eller i et privat dokument.

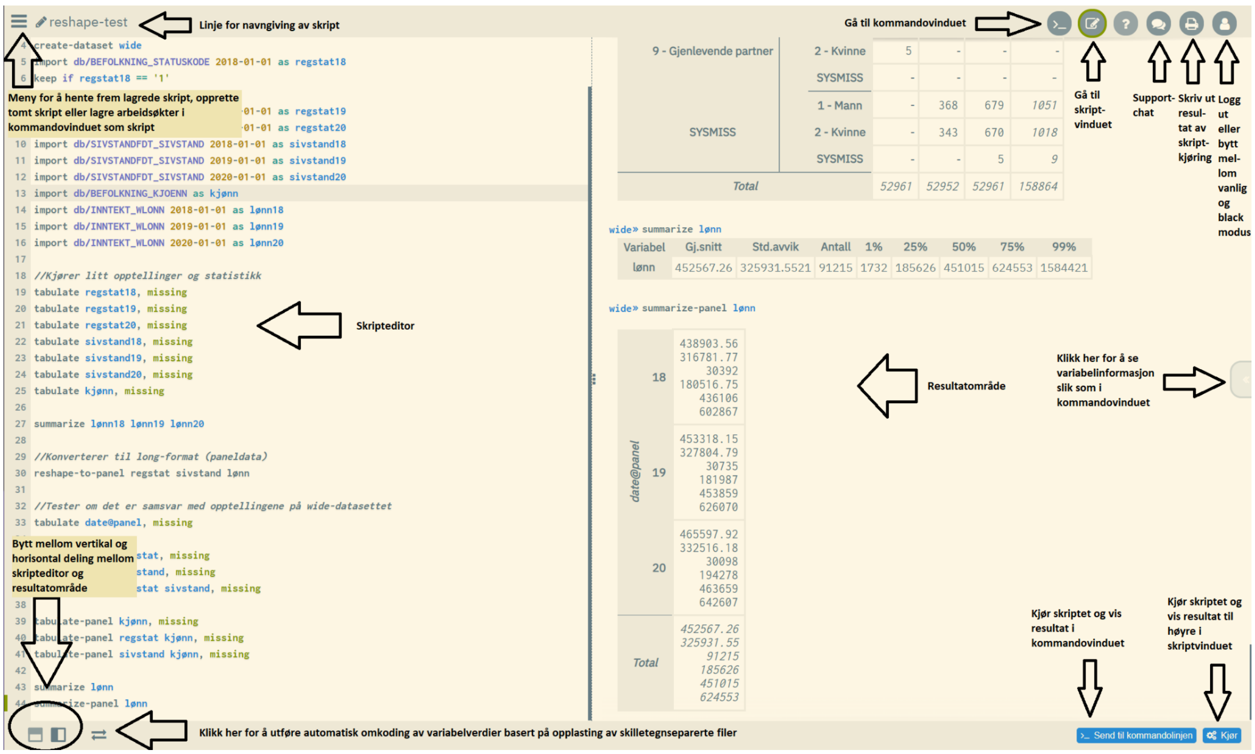
Kommandoen `clear` kan brukes til å slette alt innhold i arbeidsområdet dersom en ønsker å begynne helt på nytt. Denne bør derfor brukes med varsomhet!

Et alternativ til `clear` kan være å bruke den kjente tastekombinasjonen *<Ctrl>* + *<Z>*. Dette er en mye brukt angrefunksjon, der en gjør om det siste en gjorde (går altså ett steg tilbake). Dette kan gjøres så mange ganger en ønsker, helt til en står igjen med et tomt arbeidsområde. Tastekombinasjonen *<Ctrl>* + *<Y>* kan brukes for å gjøre om den siste "angringen". Brukere av Apple-PC'er bruker kombinasjonene *<Cmd>*+*<Z>* og *<Cmd>*+*<Y>*.

Et nyttig hjelpemiddel når en jobber i kommandolinjen er å bruke *<piltast opp>* på tastaturet. Da kan en scrolle gjennom alle tidligere kommandoer en har benyttet i kronologisk rekkefølge og velge den en ønsker å bruke på nytt. Da slipper en å legge inn manuelt samme kommandolinje flere ganger. Ofte ønsker en å kjøre ulike varianter av samme kommando (f.eks. med litt andre variabler eller parametre). Da kan en lete opp en allerede brukt kommando, og så endre litt på den før en kjører den på nytt.

1.5 Skriptvinduet

Skripteditoren befinner seg til venstre i skriptvinduet, og lar brukere legge inn sekvenser av kommandoer som kan kjøres samlet som et skript. Resultatet av en skriptkjøring vises i høyre del av skriptvinduet når en bruker “Kjør”-knappen.



1.5.1 Lage skript

Skript kan lages på følgende måter (kan kombineres):

- Skrive inn manuelt, linje for linje
- Kopiere kommandoer fra kommandovinduet (bruk kommandoen `history` og kopier de kommandoene du ønsker ved å markere dem og bruke ctrl+c)
- Lime inn et ferdig skript fra ulike kilder (husk å konvertere til ren tekstformat først - ikke lim inn skript som kommer fra formaterte formater som f.eks. Word)
- Importere/hente hele arbeidsøkten din fra kommandovinduet (se kapittel 1.5.2)

1.5.2 Lagre arbeidsøkter i kommandovinduet som skript

Arbeid en allerede har utført i kommandovinduet kan enkelt overføres til et skript for videre redigering. Det kan gjøres på fire måter:

- I skriptvinduet finnes det en menyknapp helt oppe til venstre. Der kan en velge “Nytt skript med historikk fra kommandolinjen”. Husk å legge inn et navn på skriptet i linjen over skriptvinduet. Skriptet vil da lagres med dette navnet.
- I kommandovinduet kan en skrive kommandoen `history`. Dette frembringer alle kommandoer en har kjørt i en enkelt liste som kan kopieres på vanlig måte ved å klikke på kopieringsknappen som vises når musepeker holdes over. Trykk så på ctrl+c og lim inn i et tomt skriptvindu. Husk også her å lage navn på skriptet, jfr. punktet over.
- I kommandovinduet kan en skrive kommandoen `save` etterfulgt av et valgfritt navn som du bruker på skriptet. Navnet må ha fnutter rundt (enkeltfnutter eller hermetegn/ dobbeltfnutter), f.eks. `save ‘Analyse av arbeidsledige’`. Ved denne fremgangsmåten trenger du ikke lage navn på skriptet i skriptvinduet etterpå.
- I stedet for `save`, kan man bruke kommandoen `edit`. Denne likner på `save` ved at arbeidet ditt i kommandovinduet automatisk legges inn i det eksisterende / aktive skriptet i skriptvinduet. Men vær obs på at det eksisterende skriptet overskrives med det nye innholdet. Denne kommandoen er praktisk når man ønsker å kjøre et skript der man sender resultatet til kommandovinduet (se kap. 1.5.3), for så å teste ut og legge til nye kommandoer der. `Edit` sørger da for at de nye kommandoene dine legges til skriptet du kjørte på en enkel måte.

1.5.3 Kjøring av skript

En har to muligheter når en er klar for å kjøre skriptet:

- Nederst til høyre er det to knapper, der den ene, “Kjør”, kjører gjennom skriptet på høyre side av skriptområdet (resultatområdet)
- Den andre knappen (“Send til kommandolinjen”) sender alle kommandoene til kommandovinduet og utfører dem der. Når kjøringen er fullført sendes du automatisk over til dette vinduet. Vær obs på at det du måtte ha av innhold i kommandovinduet erstattes av resultatet fra den nye skriptkjøringen, så husk å ta vare på arbeidet ditt i kommandovinduet i form av et skript før du kjører det nye.

Alle kommandolinjer i et skript eksekveres sekvensielt og viser resultatet fortløpende mens det kjøres. Ved eventuelle feil i syntaxen, stoppes kjøringen der hvor feilen befinner seg. En kan i slike tilfeller rette opp feilen og kjøre skriptet på nytt.

Merk at systemet “husker” tidligere kjørte kommandosekvenser. Så om en kjører akkurat samme kommandoskript på nytt uten endringer, vil det bare ta noen sekunder å hente frem resultatet av kjøringen. Dette prinsippet gjelder også dersom innledende deler av et skript er kjørt tidligere, der en kun har endret på siste delen. Da vil systemet “huske” det som tidligere er blitt kjørt, og kun bruke ressurser på å jobbe seg gjennom den delen av skriptet som er endret.

1.5.4 Kjøre deler av et skript

Det er mulig å kjøre deler av et skript. Dette kan gjøres på tre måter:

A) Markere ut enkeltlinjer ved å definere dem som hjelpetekst

- Legg inn tegnene “//” foran de aktuelle linjene en ønsker å holde utenfor. Systemet tolker alt som kommer bak “//” som hjelpetekst, og det vil derfor ikke kjøres

- I de neste trinn tar en så bort denne hjelpetekst-markeringen for flere og flere kommandolinjer helt til hele skriptet er ferdig kjørt. Husk at de eksekverte kommandolinjene ligger i minnet, så systemet kjører i praksis ikke alle linjene på nytt hver gang, kun de linjene der en fjerner “//”-tegnene

- En kan automatisk legge til “//” foran flere linjer ved å markere dem og bruke hurtigtastkombinasjonen alt + c. Ved å gjenta prosedyren for de samme linjene, vil “//”-tegnene tas bort igjen

```
//Kobler til databank
require no.ssb.fdb:12 as db

//Starter med å hente variablene en trenger
create-dataset demografidata
import db/BEFOLKNING_KJOENN as kjonn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/BEFOLKNING_STATUSKODE 2018-01-01 as regstat
// import db/SIVSTANDFDT_SIVSTAND 2018-01-01 as sivstand
// import db/INNTEKT_BRUTTOFORM 2018-01-01 as formue
// import db/INNTEKT_WYRKINNT 2019-01-01 as innt19
```

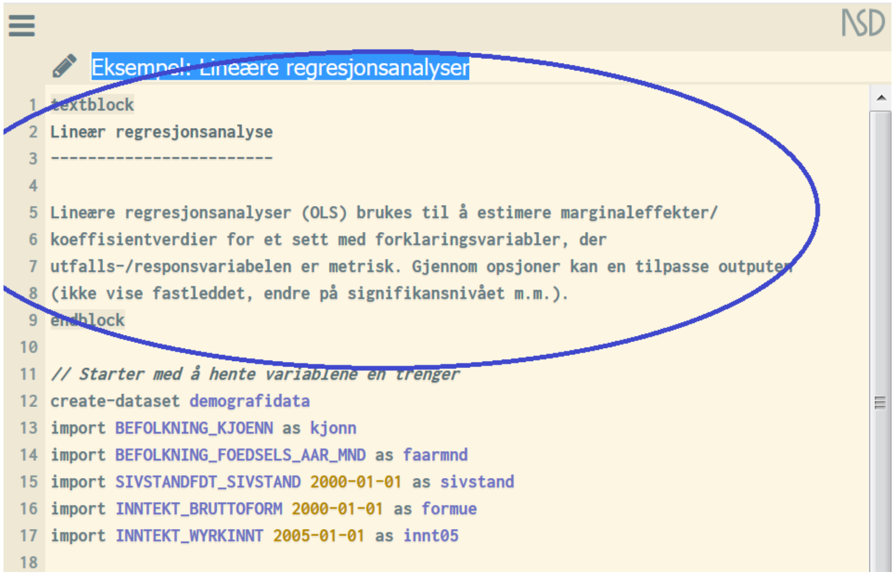
B) Markere ut større deler av et skript ved å definere blokker som hjelpetekst

- Legg inn hhv. `textblock` og `endblock` før/etter en blokk med kommandolinjer. Alt i mellom vil da bli holdt utenom kjøringen. Merk at meningen med `textblock` og `endblock` er å legge inn analysetekst/kommentarer til analyser utført i kommandovinduet. Kommandovinduet kan derfor fylles opp med de kommandolinjene som ble “markert ut”, og det kan se litt rotete ut. Men etterhvert som færre og færre linjer blir holdt utenfor, vil kommandovinduet inneholde gradvis mindre av dette

- Fordelen med `textblock` og `endblock` er at det er mindre tidkrevende dersom antallet linjer som skal markeres ut er omfattende

- På samme måte som ved bruk av tegnene “//”, vil systemet “huske” det som er kjørt fra før og vil hoppe rett ned til den delen av skriptet der en har tatt bort “textblock-markeringen”. Merk at dette ikke fungerer dersom det er begynnelsen av skriptet som først blir markert ut, for så å bli kjørt etterpå

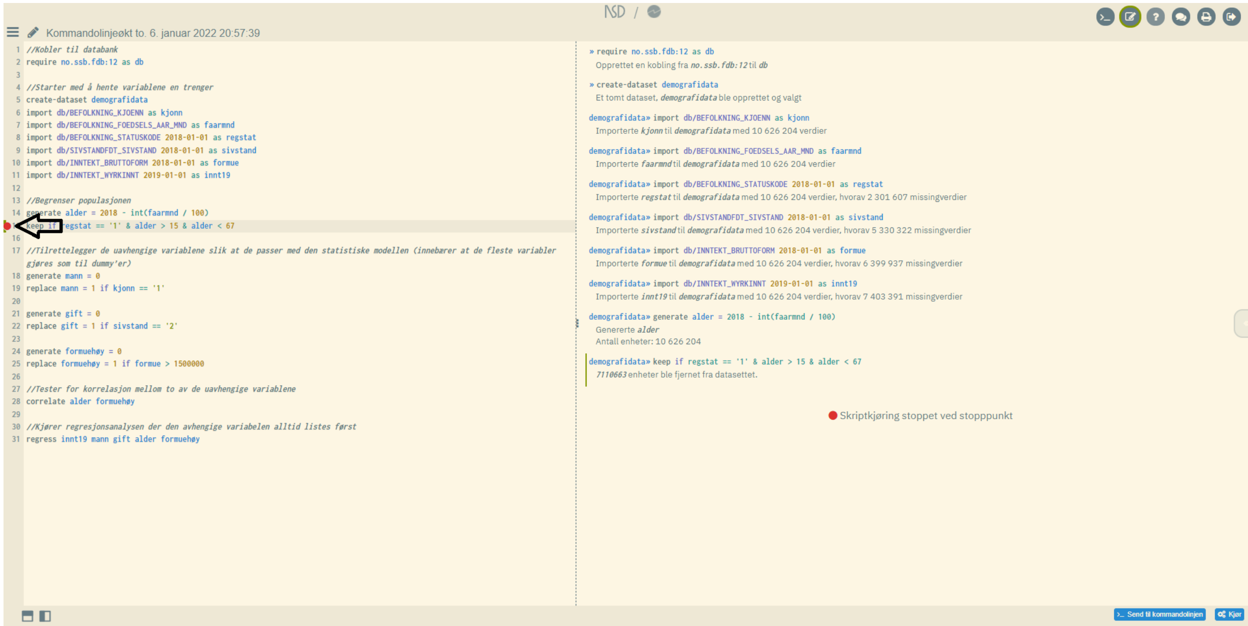
Eksempel: Bruk av textblock i skript



C) Klikke på et linjenummer i skriptet og deretter trykk på “Kjør” eller “Send til kommandolinjen”

Dette vil kjøre gjennom alle linjene i skriptet ditt frem til den linjen som er merket, og stoppe der. Denne metoden kan brukes til å gradvis kjøre gjennom mer og mer dersom du ikke er klar for å kjøre gjennom hele skriptet ditt.

Eksempel: Klikke på linjenummer for å kjøre alle linjene frem til dette punktet



1.5.5 Fordeler ved bruk av skripteditor

Det er mange fordeler ved å aktivt bruke skripteditoren til å eksekvere sekvenser av kommandoer:

- Fungerer som sikring av arbeid
    - Når en navngir et skript, lagres det i systemet og kan hentes frem igjen når en måtte ønske
    - En kan lagre skript i tekstformat ved å kopiere over til et tekstdokument. Dette fungerer som ekstra sikkerhet. Om en av ulike grunner mister arbeidet, kan en hente det frem, lime det inn i editoren og kjøre på nytt. Da gjenskapes alt analysearbeidet slik det opprinnelig var. NB! Ekstern lagring av skript bør gjøres i rent tekstformat av typen ".txt" via programmer som Notisblokk o.l. Mer avanserte tekstbehandlingsverktøyer som Word og Google Doc foretar tekstformateringer som gjør at enkelte tegn kan bli forandret, bl.a. enkeltfnutter. Når dette så limes inn i microdata.no igjen, risikerer en at systemet ikke gjenkjenner tegnene og låser seg.
- Skript er en måte å systematisere og huske arbeidet på. En kan endre på rekkefølgen i kommandosekvenser eller gjøre andre justeringer, og dessuten legge til hjelpetekst (se

kapittel 1.5.4) som gjør det lettere både for seg selv og andre å skjønne hva hensikten med de ulike operasjonene er

- Skript fungerer som en logg over arbeid (kan legges til analyserapporter for å dokumentere arbeidet)

- Det blir enkelt å gjøre justeringer i en analyse. Om en ønsker å gjøre ting litt annerledes, kan en endre skriptet og kjøre på nytt. Endrede skript kan lagres med nye navn. Det gjør det enklere å dokumentere og sammenlikne resultater

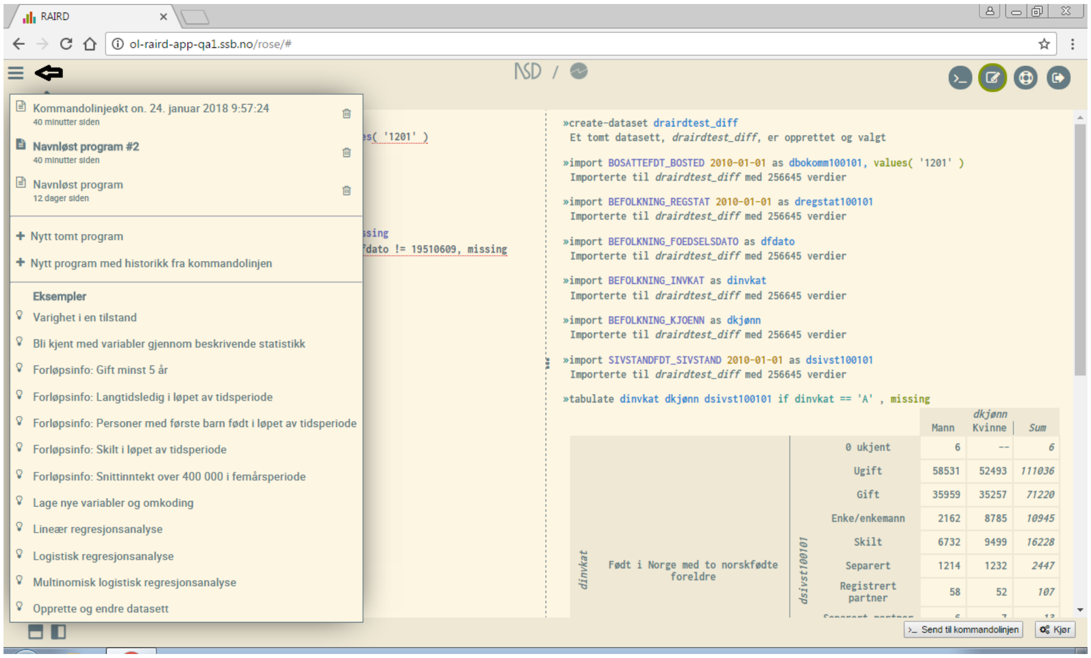
- Å bruke skriptmuligheten aktivt gjør det enklere å samarbeide med andre. En kan sende skript i tekstformat til andre kollegaer, f.eks. via mail

- En kan jobbe med skript på samme måte som i Google Doc eller andre tekstbehandlingsprogrammer som f.eks. Word: Det er mulig å redigere ved å klippe og lime tekst, markere tekstbolker og flytte rundt på dem etter behov

- Systemet “husker” tidligere kjørte skript gitt at de er uendret. Dermed vil det bare ta noen sekunder å gjenskape et tidligere kjørt resultat. Merk at dersom en endrer enkelte ting i et skript og kjører på nytt, så vil systemet behandle dette som et helt nytt sett med kommandoer, og det vil ta betraktelig mye lenger tid å kjøre gjennom det. Om en derimot har behov for å kun justere på kommandolinjer på slutten av et skript, vil systemet hoppe rett til den aktuelle delen og kun bruke ressurser på å prosessere dette. Det øvrige blir hentet frem fra minnet.

1.5.6 Organisering av kommandoskript

Bildet under viser de ulike mulighetene en har for å organisere skript (programmer). Menyknappen øverst til venstre i skriptvinduet gir mulighet til å opprette et nytt tomt program (om en vil starte fra scratch), eller å hente inn alle kommandoer en har brukt i det aktive arbeidsvinduet (“Nytt program med historikk fra kommandolinjen”).



Innholdet i aktive skript lagres fortløpende med et defaultnavn etterhvert som en arbeider (slik som i Google Doc). Ved å legge inn en selvvalgt tittel øverst i skriptet, der hvor det står “Navnløst program”, vil defaultnavn erstattes med dette. Alt arbeid en gjør på skriptet vil da automatisk lagres med dette navnet.

En kan lagre så mange skript en ønsker gjennom å navngi dem med nye navn. Systemet vil også foreta en automatisk lagring av gjeldende skript med jevne mellomrom (sikkerhetskopi).

Nederst i program-menyen finnes det eksempler på kommandosekvenser for ulike typer oppgaver. De kan brukes som aktive skript som kan kjøres direkte. De kan også redigeres og lagres med nye navn (kan brukes som en mal).

1.5.7 Problemløsninger ved bruk av skript

Det er nesten ikke til å unngå at systemet finner feil i kommandosyntaxen når skript kjøres. Dette kan være feilstavinger eller en logiske feil som ikke lar seg eksekvere. Systemet vil da stoppe kjøringen av skriptet der feilen befinner seg, og markere den aktuelle linjen. En passende feilmelding vil også bli gitt.

Løsning:

i) Sjekk hva som kan ha gått galt. Se spesielt på den linjen som er markert med feil. Kjør den delen av skriptet som ikke inneholder feil, altså frem til den linjen som viser feil (se kapittel 1.5.4 for hvordan en kjører deler av et skript)

ii) Dobbeltsjekk om syntaxen er riktig, at variabelnavnet er riktig skrevet, at datoen for import er gyldig (det kan tenkes at en variabel ikke har data for det aktuelle måletidspunktet)

iii) Bruk statistiske hjelpemidler som kommandoene `tabulate` eller `summarize`. Se om det er noe feil i måten de aktuelle variabler er kodet på. Sjekk også om verdiformatet er korrekt (numerisk eller alfanumerisk)

iv) Dummyvariabler eller kategoriske variabler der minst én av kategoriene har få observasjoner kan føre til uønskede analyseresultater:
   - Ved bruk av regresjonsanalyser vil kun de enhetene med gyldige verdier for samtlige variabler som inngår bli analysert
   - Enkelte dummyvariabler kan i utgangspunktet ha tilstrekkelig antall observasjoner for begge verdiene 0 og 1, men kan etter at enheter holdes utenfor regresjonsanalysen nå stå med observasjoner kun for én av verdiene
   - Analysen vil da bli stoppet og en feilmelding bli gitt. En løsning kan da være å kode om de aktuelle variablene slik at en får flere enheter i kategoriene med minst observasjoner i. Eventuelt kan en droppe de problematiske variablene fra analysen

2. Oppretting og endring av datasett

Dataanalyser i microdata.no krever at en først knytter seg opp mot en databank, deretter oppretter et tomt datasett, og så importerer de ønskede variabler inn i datasettet. Sistnevnte gjøres gjennom bruk av import-kommandoer (se kapittel 2.3.1, 2.3.2 og 2.4).

Om en ønsker å justere på utvalget eller fjerne variabler en ikke ønsker å jobbe videre med, brukes kommandoene `drop` eller `keep` (se kapittel 2.6 og 2.7). En kan spesifisere hvilken variabel en ønsker å slette. Dersom ingen variabel spesifiseres, slettes hele records i kombinasjon med en if-betingelse.

2.1 Koble til databank

Når en logger seg inn i microdata.no for første gang, eller oppretter en helt ny arbeidsøkt, vil innholdet i kommandovinduet være tomt. For å kunne lage datasett er det nødvendig å først koble seg opp mot tilgjengelig databanker. Som regel er det tilstrekkelig å koble seg opp mot databanken som inneholder en omfattende samling med registerdata fra SSB. Microdata.no vil etterhvert tilby databanker med data også fra andre datakilder enn SSB.

En kan selv bestemme hvilken versjon av databanken en vil koble seg mot. Siste versjon vil alltid inneholde de nyeste variabler og årganger, og anbefales derfor å bruke. Det er imidlertid mulig å koble seg mot tidligere versjoner også, om en f.eks. ønsker å avdekke eventuelle effekter som kan knyttes til forskjeller i versjonene. Når data oppdateres med nye årganger, vil dette kunne påvirke hendelser tilbake i tid for de ulike variablene. Derfor opprettes det alltid nye databank-versjoner i slike tilfeller.

I variabeloversikten vil en finne en oversikt over alle tilgjengelige databanker, versjoner av disse, og hvilke variabler de inneholder. Kapittel 1.2 beskriver dette nærmere.

Kommando for kobling mot databanker:

```
require <databank:versjon> as <alias>
```

Eksempel:

```
require no.ssb.fdb:15 as db
```

2.2 Opprette et datasett

For å kunne jobbe med data og analyser i microdata.no, trenger en å opprette et datasett som en kan fylle med variabler. Dette gjøres ved å skrive følgende kommando i kommandolinjen:

```
create-dataset <datasett>
```

Eksempel:

```
create-dataset mittdatasett
```

Det er tilstrekkelig å opprette ett datasett, men en kan i prinsippet opprette så mange en ønsker. Ett eksempel der dette er aktuelt er når en vil arbeide med variabler organisert ulikt (har forskjellige enhetsnivåer). I tillegg til et datasett på individnivå (én record per individ) kan en bruker ønske å analysere andre datasett organisert med enkelthendelser som enhet.

En kan i prinsippet opprette et datasett *før* en kobler seg mot en databank, men det vil ikke være mulig å hente variabler før oppkoblingen mot databanken er foretatt.

2.3 Hente variabler inn i et datasett

Neste trinn er å fylle datasettet med ønskede variabler.

Alle underliggende variabler i microdata.no er i utgangspunktet organisert på samme måte; på hendelsesnivå:

individnummer x verdi x startdato x stoppdato

I microdata.no har en tilgang på fire typer variabler, basert på temporalitet:

1) Forløpsvariabler med sekvenser av hendelser (hver observasjon representerer en tilstandsendring, dvs. at variabelen endrer verdi, og en har variable start- og stoppdatoer)

2) Faste variabler med kun én observasjon per enhet (f.eks. kjønn, fødselsdato, fødeland)

3) Tverrsnittsvariabler målt på faste tidspunkt (variabler brukt til statistikkproduksjon, der en kun vet verdien på det aktuelle tidspunktet, startdato=stoppdato)

4) Akkumulerte variabler - hovedsaklig økonomiske opplysninger som angir årlige beløp, f.eks. årlig inntekt, formue etc.

En bygger opp datasett gjennom å benytte kommandoen `import`, der en vanligvis spesifiserer en uttrekksdato/måledato (unntaket er variabler med konstante verdier som f.eks. kjønn).

I kapittel 2.3.1 vises det i detalj hvordan man bruker kommandoen `import` til å importere variabler inn i et datasett med person som enhetsnivå. Kapittel 2.3.2 viser alternativ import av variabler med hendelsesnivå som enhet (personer er representert ved flere observasjoner over tid, avhengig av antallet hendelser som har skjedd). Til dette brukes kommandoen `import-event`.

2.3.1 Datasett med tverrsnittsopplysninger

Kommandoen `import` brukes til import av følgende opplysninger (fire temporalitetstyper):

- Faste opplysninger (f.eks. kjønn, fødselsdato, fødeland)
- Forløpsopplysninger (gjør i praksis et valgfritt uttrekk fra forløps-/ hendelsesvariabler)
- Tverrsnittsopplysninger på forhåndsgitte måledatoer
- Akkumulerte opplysninger (hovedsaklig årlige økonomiske opplysninger som inntekt, formue etc)

En spesifiserer navnet på variabelen en ønsker å hente til sitt arbeidsdatasett, samt datering for når opplysningen skal måles. Om du jobber i kommandovinduet, foreslår systemet relevante variabler og dateringer gjennom en selvutfyllingsfunksjon som minimerer sjansen for å skrive feil.

For hver gang `import` kjøres, vil en ny variabel legges til arbeidsdatasettet (kobles på automatisk). Det resulterende datasettet vil bestå av én observasjon per enhet (individ), og med et valgfritt antall variabler.

Ved import av variabler med faste opplysninger trenger ikke måledatering oppgis:

```
import <variabel> as <alias>
```

For øvrige variabler må en oppgi en ønsket måledato formatet *YYYY-MM-DD:*

```
import <variabel> <måledato> as <alias>
```

For tverrsnittsvariabler må imidlertid aktuell statistikkdato benyttes siden verdiene i prinsippet kun vil gjelde bare for denne aktuelle datoen (en kjenner ikke til faktiske endringsdatoer for slike variabler). Om du jobber i kommandovinduet, foreslår analysesystemet i slike tilfeller de aktuelle statistikkdatoer gjennom selvutfyllingsfunksjonen slik at dette blir lett å oppgi. Ved import av forløpsopplysninger foreslås i stedet den sist brukte dateringen. For variabler med akkumulerte opplysninger er det den årlige verdien for det aktuelle året en importerer, så det har ikke noe å si konkret hvilken dato en oppgir så lenge en angir riktig år.

Eksempel: Datamatrise ved bruk av import (4 variabler)

ID	Variabel 1	Variabel 2	Variabel 3	Variabel 4
123456	1	200000	0301	1
135791	1	410000	0301	1
147036	2	515000	1201	sysmiss
159371	2	309011	1101	sysmiss
160505	2	357000	1101	1
173951	2	399000	0301	3

Viktig:

Kommandoen `import` gjør i praksis to operasjoner:

a) Henter verdier for en gitt variabel
b) Kobler variabel på eksisterende datasett gjennom en såkalt "left-join" kobling (standardvalg)

At variabler kobles på datasettet gjennom "left-join" innebærer at en kun importerer verdier for enheter (individer) i det eksisterende datasettet. Derfor er det viktig å starte med å importere en variabel der færrest mulig i en tenkt populasjon har manglende verdier, som f.eks. kjønn, landbakgrunn eller fødselsdato. Starter en derimot med å importere en variabel som angir sykefravær på en gitt dato, er det kun personene med sykefravær på denne datoen som en arbeider videre med ved etterfølgende import-steg. Det vil da ikke være mulig å hente opplysninger om andre personer i senere trinn. <u>Det første import-steget vil med andre ord definere utvalget en jobber videre med.</u>

Alternativ import-løsning: Import-kommandoen har en opsjon, outer_join, som kan benyttes om du ønsker å importere en ny variabel ved bruk av såkalt "outer join"-tilnærming. Dette innebærer at også nye enheter (individer) som ikke eksisterer fra før i ditt datasett blir lagt til, gitt at de har en gyldig verdi for den nye variabelen. Populasjonen din vil da øke i størrelse avhengig av hvor mange nye enheter som har en gyldig verdi for den nye variabelen. Dette kan være nyttig om du ønsker å lage en populasjon som dekker alle enheter (individer) som har hatt en verdi/status over et gitt tidsrom, og ikke bare koble på ny informasjon kun for dem som eksisterer i populasjonen definert av variabel nummer 1. Eksempel: `import lønn19, outer_join`

Merk at en har anledning til å "trimme" ned utvalgspopulasjonen underveis i prosessen med å bygge opp et datasett, gjennom kommandoene `drop` og `keep`, jfr. kapittel 2.6.

Personer i et eksisterende datasett som har manglende verdi for en importert variabel vil fortsatt være med i utvalget, men vil få en såkalt sysmiss-verdi (se kapittel 2.6).

Om en har en klar idé om hvilke enheter (individer) som skal inngå i en analysepopulasjon, kan det være lurt å "trimme" ned utvalget en jobber med så tidlig som mulig. Dette vil kunne gi betydelige forbedringer i hvor raskt systemet jobber.

Om første variabel som importeres er av universell karakter, f.eks. "Kjønn", vil datasettet ditt bestå av flest mulig individer fra den totale databanken, inkludert personer som er døde, emigrert eller ikke født på det aktuelle tidspunktet en er interessert i. Dette kan løses ved å importere variabelen `BEFOLKNING_STATUSKODE` målt ved det aktuelle tidspunktet, og deretter beholde alle individer som har verdien '1' (= bosatt i Norge). Eksempel:

```
require no.ssb.fdb:12 as db
create-dataset demografi
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_STATUSKODE 2015-01-01 as regstat15
keep if regstat15 == '1'
```

2.3.2 Datasett med hendelsesopplysninger

I tillegg til å hente ut opplysninger på valgte eller gitte datoer, kan en foreta beregninger basert på hendelser over tid. F.eks. kan en være interessert i å finne individer som giftet seg i løpet av en lengre tidsperiode, som ble arbeidsledig over et gitt tidsspenn, eller som var arbeidsledige i over 6 måneder i en gitt periode. Til dette benyttes kommandoen `import-event` som importerer *alle* records (= hendelser) per enhet (= individ) over et angitt tidsspenn. I tillegg til variabelnavnet angis 2 tidspunkter for datauttrekkets hhv. start- og stopp-tidspunkt. Da vil alle hendelser som har skjedd mellom de to tidspunktene hentes til ditt datasett. Datasettet vil inneholde et varierende antall records for hver enhet (= individ), avhengig av hvor mange endringshendelser som har skjedd for hver enkelt.

Merk at en bare kan importere én hendelsesorganisert variabel til et gitt datasett, og at et slikt datasett ikke kan inneholde andre variabler. En må altså opprette ett arbeidsdatasett for hver hendelsesorganiserte variabel en trenger å arbeide med. Importen gjøres på følgende måte:

```
create-dataset <dataset>
import-event <variabel> <startdato> to <stoppdato> as <alias>
```

Eksempel: Datamatrise ved bruk av import-event (tidsintervall: 2000-01-01 - 2003-01-01)[1]

ID	Start	Stopp	Variabel
123456	2000-01-01	2000-05-30	1
123456	2000-05-31	2001-12-31	4
123456	2002-01-01	2003-08-15	2
135791	2000-04-10	2002-03-03	2
135791	2002-03-04	2002-11-11	3
147036	2002-02-28	2004-07-16	1

[1] Merk at alle hendelser som overlapper med perioden 2000-01-01 - 2003-01-01 tas med ved import

2.3.3 Tverrsnitts- vs. hendelsesorganiserte datasett

Til forskjell fra tverrsnittsorganiserte datasett som bygges opp gjennom kommandoen `import`, kan en ikke bygge opp datasett gjennom `import-event`. Grunnen til dette er at datauttrekk på hendelsesnivå alltid vil ha ulikt antall records per individ, og det vil gi liten mening å koble slike uttrekk sammen til et felles datasett. Derfor må en opprette et nytt datasett for hvert hendelsesbaserte datauttrekk (se kapittel 2.3.2).

Hensikten med hendelsesorganiserte datauttrekk er som nevnt å foreta beregninger basert på hendelser over tid, gjennom kommandoen `collapse`. Dette vil transformere det hendelsesbaserte datasettet til et datasett på enhetsnivå (én record per individ), med det aggregerte målet som variabelverdi (målt over det angitte tidsspenn), noe som gjør det mulig å koble variabelen sammen med tverrsnittsorganiserte datasett for videre analyser.

I kapittel 2.8 beskrives metoden for å koble sammen datasett.

2.4 Datasett med regelmessige målinger over tid (paneldata)

For å kunne foreta avanserte regresjonsanalyser i form av paneldataanalyse, må data organiseres på en annen måte enn ved vanlige regresjonsanalyser. Paneldata er datasett der hver enhet har oppgitt verdier for samtlige variabler målt over et gitt antall måletidspunkt. Dette har den fordelen at en kan ta med tidskomponenten i analyser, og at en får mye større datagrunnlag og gjerne analyser av en bedre kvalitet.

Det finnes et stort batteri av teknikker for paneldataanalyse, skillet går på hvilke antakelser som gjøres om variablenes variasjon over tid. Vanlige varianter som brukes er “fixed effect”- og “random effect”-analyser. Denne analyseformen vil bli gjennomgått i kapittel 5.9.

Data som skal brukes i paneldataanalyse må importeres på følgende måte:

```
create-dataset <dataset>
import-panel <variabelliste> <måledatoliste> as <alias>
```

Eksempel: Datamatrise ved bruk av import-panel (3 variabler, 3 måletidspunkt)

ID	**Tid**	**Variabel 1**	**Variabel 2**	**Variabel 3**
123456	2000-01-01	1	200000	0301
123456	2001-01-01	1	210000	0301
123456	2002-01-01	2	215000	1201
135791	2000-01-01	2	305011	1101
135791	2001-01-01	2	301000	1101
135791	2002-01-01	3	299000	0301
147036	2000-01-01	1	150000	2030
147036	2001-01-01	1	159000	2030
147036	2002-01-01	3	199000	0301

Merk:

- Paneldatasett blir fort veldig store ettersom alle enheter/individer i datasettet måles T ganger, der T står for antall målinger. Dette gjelder særlig om en importerer mange variabler i tillegg
- En god praksis ved opprettelse av paneldatasett er å først lage en populasjon av passende størrelse, så duplisere denne og til slutt importere paneldata inn i det tomme datasettet med den dupliserte populasjonen

Eksempel: Lage populasjon, duplisere enheter inn i nytt datasett, og til slutt importere paneldata for den gitte populasjonen (= bosatte i Oslo per 1/1 2010 i alderen 18-39 år)[1]

```
require no.ssb.fdb:12 as db

create-dataset populasjon
import db/BOSATTEFDT_BOSTED 2010-01-01 as bosted
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate alder = 2010 - int(faarmnd/100)
keep if alder >= 18 & alder < 40 & bosted == '0301'

clone-units populasjon paneldata

use paneldata
import-panel db/INNTEKT_WLONN db/SIVSTANDFDT_SIVSTAND
db/BOSATTEFDT_BOSTED 2011-12-31 2012-12-31 2013-12-31 2014-12-31
```

[1] Paneldatasett lages ved hjelp av en enkelt import-panel-kommando. En kan ikke importere i flere omganger til det samme paneldatasettet. En kan heller ikke mikse vanlige tverrsnittsdata og/eller forløpsdata med paneldata.

Det er også mulig å lage et paneldatasett gjennom å konvertere et eksisterende tverrsnittsdatasett til panel-/long-format ved bruk av kommandoen `reshape-to-panel`. Se kapittel 2.9.1 for en gjennomgang av denne kommandoen.

2.5 Forflytte seg mellom datasett

For å flytte seg mellom datasett brukes følgende kommando:

```
use <dataset>
```

Unntaket er når en oppretter et nytt datasett gjennom `create-dataset`. Da flyttes en automatisk til det nye datasettet.

2.6 Filtrering av datasettutvalg

Man kan benytte en filteropsjon til å spesifisere hvilke verdier som skal inngå i datasettet, f.eks. om en bare vil importere kvinner:

```
import BEFOLKNING_KJOENN as kjønn, values ( '2' )
```

Alternativt kan en først importere variabler på vanlig måte og så trimme ned utvalget etterpå gjennom kommandoene drop eller keep:

```
import BEFOLKNING_KJOENN as kjønn
drop if kjønn == '1'
```

IF-betingelser kan brukes i mange sammenhenger i microdata.no, også i forbindelse med `drop` og `keep`, og kan bygges opp med de vanlige logiske operatorene:

- Større enn >
- Mindre enn <
- Er lik ==
- Større enn eller lik >=
- Mindre enn eller lik <=
- Er ulik !=
- Eller |
- Og &

For å fjerne personer under 18 år fra utvalget, kan en skrive følgende:

```
keep if alder >= 18
```

Verdi for manglende data ("missingverdier") kan angis på følgende måte:

```
sysmiss(<variabel>)
```

For å fjerne alle individer uten oppgitt lønnsinntekt, kan en da skrive:

```
drop if sysmiss( lonn )
```

Det er også mulig å trekke et tilfeldig utvalg av en datapopulasjon. Dette gjøres med kommandoen `sample`. For mer om syntax og eksempler, bruk kommandoen `help sample`.

2.7 Fjerning av variabler fra datasett

I en analysesituasjon ser en ofte underveis at en ikke trenger alle variabler en først har importert. En del variabler brukes gjerne bare som grunnlag for å avlede verdier for nye variabler, og da trenger en ikke dra med seg alle de underliggende.

Å rydde et datasett gjøres gjennom kommandoen `drop`, der en legger til navnet på den variabelen en vil fjerne:

```
drop <variabel>
```

Som vi har sett, kan denne kommandoen brukes både til å fjerne enheter (= rekker i datamatrisen), jfr kapittel 2.6, **og** variabler (= kolonner i datamatrisen).

2.8 Aggregering og kobling av datasett

Datasett bygges vanligvis opp gjennom kommandoen `import,` der en legger til én og én variabel målt ved gitte tidspunkter. Variablene må da ha samme enhetstype, vanligvis person gitt ved nøkkelidentifikatoren `PERSONID_1`. Sammenkoblingen gjøres automatisk av analysesystemet, og en trenger bare å forholde seg til `import`-kommandoen der en spesifiserer variabelnavn, uttrekksdato og eventuelt alias.

Analysesystemet gjør det imidlertid mulig å analyse data med andre enhetstyper. Det kan være på hendelsesnivå, kommunenivå, familienivå, kursnivå, jobbnivå etc. Slike data med andre enhetstyper enn person kan ikke uten videre importeres rett inn i et persondatasett (datasett med enhetstypen person gitt ved `PERSONID_1`). De må først bearbeides til passende enhetsnivå gitt ved variabelen en ønsker å bruke som koblingsnøkkel i måldatasettet, og deretter kobles på datasettet ved bruk av kommandoen `merge.`

Data på et lavere enhetsnivå enn person, f.eks. hendelses- eller kursnivå[3], må aggregeres til personnivå ved bruk av kommandoen `collapse()` før du kobler det på et persondatasett ved bruk av `merge`. Kommandoen `collapse()` gjør to ting:

- Aggregerer data opp til et høyere enhetsnivå gitt ved en identifikatorvariabel. I prinsippet kan alle kategoriske variabler brukes som identifikatorvariabler, f.eks. ved å bruke bostedskommune som identifikator kan en aggregere data opp til kommunenivå
- Foretar en oppsummerende kalkulering på tvers av enhetene, for hver av de nye aggregerte enhetene. Kalkuleringstype angis i parentes etter kommandoen, og kan være summering, maximumsverdi, snittverdi, antall verdier m.m.

Data på samme eller høyere enhetsnivå enn person, f.eks. kommune- eller familienivå, kan imidlertid kobles på et persondatasett ved bruk av aktuell variabel på samme nivå i måldatasettet (brukes som koblingsnøkkel).

Eksempel der en aggregerer kursdata[4] (data om pågående utdanning) opp fra kursnivå til personnivå og kobler på et persondatasett:

```
collapse(max) utdanningsnivå, by(fnr)
rename utdanningsnivå høyeste_utdanningsnivå
merge høyeste_utdanningsnivå into persondatasett
```

Eksempel der en aggregerer opp fra personnivå til familienivå (summerer familiemedlemmers inntekter og lager familieinntekt) og kobler familiedataene på et persondatasett:

```
collapse(sum) inntekt, by(familienummer)
rename inntekt familieinntekt
merge familieinntekt into persondatasett on familienummer
```

Merk at det i eksempelet over spesifiseres koblingsvariabel gjennom uttrykket `on familienummer`. Dette må alltid gjøres dersom en bruker andre koblingsvariabler enn nøkkelidentifikatoren `PERSONID_1`.

Se kapittel 2.11 for eksempler på hvordan koble på opplysninger om hhv. foreldre, familier og kurs. Sistnevnte illustrerer sammenkoblinger av data på lavere nivå enn person (kursdata er opplysninger om pågående utdanning gitt ved aktuelt kurs/fag som tas på et gitt tidspunkt, der

[3] Kursdata er litt annerledes enn øvrige persondata ettersom disse dataene selv etter uttrekk på et gitt tidspunkt innholder persondata med flere observasjoner per individ. Dette reflekterer det faktum at det er mulig å delta på flere ulike kurs/studier på samme tid. Samme prinsipp gjelder for jobbdata der det er mulig å ha flere jobber på samme tid.
[4] Kursdata sammen med jobbdata er litt spesielle i forhold til øvrige persondata. Se fotnote 3 på forrige side.

personer kan ta flere kurs samtidig). Data om foreldre og familier illustrerer sammenkoblinger av data på *høyere* nivå enn person.

2.9 Restrukturere datasett

Vanlige tverrsnittsdatasett lages i microdata.no ved å bruke kommandoen `import` til å legge til en og en variabel. Datasettet vil da inneholde opplysninger på variabelnivå, der hver enkelt enhet (individ) har én record hver. For å legge til repeterende målinger av en variabel, må dette gjøres ved å importere den aktuelle variabelen flere ganger med nye tidsangivelser. Dette dataformatet kalles ofte for “wide-format” siden opplysninger organiseres horisontalt.

Data kan også organiseres vertikalt som paneldata, også kalt “long-format”. Kommandoen `import-panel` kan brukes til dette. Man angir da et sett med variabler, samt et sett med måletidspunkter man ønsker at opplysningene skal måles på. Hver enhet (individ) vil i dette tilfellet ha flere enn én record, avhengig av hvor mange måletidspunkter som angis.

Kapittel 2.3.1 og 2.4 forklarer prinsippene rundt disse to hovedtypene av dataorganisering mer detaljert.

Det er også mulig å restrukturere dataorganiseringen fra tverrsnittsformat (wide) til paneldataformat (long) og vice versa gjennom å bruke hhv. kommandoene `reshape-to-panel` og `reshape-from-panel`. De følgende underkapitler vil forklare hvordan disse to kommandoene brukes.

2.9.1 Restrukturere fra tverrsnittsdata til paneldata

Til statistikk og analyser i microdata.no brukes vanligvis datasett opprettet gjennom kommandoen `import`. Dette er datasett av typen “wide”, hvor opplysninger om alle enheter i en populasjon struktureres horisontalt på variabelnivå. Den nye kommandoen `reshape-to-panel` gjør det nå mulig å endre datastrukturen til long-format (panel-format), hvor opplysninger om hver enhet struktureres vertikalt på observasjons-/record-nivå.

Variabler som måles over flere tidspunkt og som man ønsker på long-/panel-format, må navngis gjennom `reshape-to-panel` med angitte prefiks som består av bokstavene (prefikset) fra den opprinnelige variabelen i wide-datasettet. Øvrige variabler som det ikke angis prefiks for, typisk opplysninger som bare måles én gang (kjønn, fødeland etc), defineres automatisk som faste opplysninger og verdiene for disse repeteres for alle undernivåer for hver enhet.

Suffiksene til de opprinnelige “wide”-variablene med repeterende målinger må bestå av heltall. Disse vil danne undernivået til long- / panel-datasettet. Typiske eksempler på suffikser vil være to- eller firesifrede år, eller andre typer tidsangivelser som også peker på måned eller kvartal, f.eks. 201901, 201902 osv. Du står fritt til å velge andre typer suffikser så lenge det består av sifre[5]. Suffikser av type 1, 2, 3, 4 osv. er også tillatt.

Illustrasjonen nedenfor viser hvordan restruktureringen logisk foregår under panseret. Eksempelet viser et datasett med wide-format som inneholder variablene sivstand18-sivstand20, lønn18-lønn20, og kjønn. Sivilstand (sivstand) og lønn måles altså for årene 2018-2020, mens kjønn er en fast opplysning som bare måles en gang. Datasettet konverteres til long-format ved hjelp av kommandoen `reshape-to-panel sivstand lønn`. Variabelen date@panel opprettes automatisk og inneholder undernivået som i dette tilfellet er tosifret årstall.

ID	sivstand18	sivstand19	sivstand20	lønn18	lønn19	lønn20	kjønn
1	1	1	2	120000	150000	180000	1
2	1	1	1	300000	340000	360000	2
3	2	2	2	400000	600000	630000	1
4	2	4	4	160000	170000	175000	2

ID	date@panel	sivstand	lønn	kjønn
1	18	1	120000	1
2	18	1	300000	2
3	18	2	400000	1
4	18	2	160000	2
1	19	1	150000	1
2	19	1	340000	2
3	19	2	600000	1
4	19	4	170000	2
1	20	2	180000	1
2	20	1	360000	2
3	20	2	630000	1
4	20	4	175000	2

Kommandoen `reshape-to-panel` har flere bruksområder:

[5] Også tegnet “_” er tillatt, f.eks. “sivstand2019_01_01”. Men etter at reshape-operasjonen er fullført, vil tegnet bli fjernet fra undernivåene. F.eks. ved bruk av suffikset “2019_01_01” vil tilhørende undernivå bli endret til “20190101” i det transformerte datasettet.

- Et mer fleksibelt alternativ til `import-panel` som også lager paneldatasett, men som har en del begrensninger. Blant annet må alle variabler her ha gyldige måletidspunkter for alle måletidspunkter, noe som kan være utfordrende dersom tverrsnittsvariabler inngår i datasettet (variabler som bare har verdier på gitte årlige, kvartalsmessige eller månedlige datoer). Kommandoen `reshape-to-panel` tillater alle kombinasjoner av variabler.

- En del analyser krever long-format, og støtten for dette blir nå forbedret. I tillegg har man tilgang til all fleksibilitet og funksjonalitet knyttet til wide-datasett, og kan gjøre hele tilretteleggingen i dette formatet før man enkelt restrukturerer til long-format etterpå. Dette er nyttig om man har behov for å sammenlikne og gjøre operasjoner over variabelverdier på tvers av undernivå (over tid), f.eks. sammenlikne verdien på lønn i 2020 i forhold til 2019.

Se kapittel 2.12 for eksempel på hvordan dette gjøres i praksis.

2.9.2 Restrukturere fra paneldata til tverrsnittsdata

Datasett opprettet gjennom en av kommandoene `import-panel` eller `reshape-to-panel` er av typen panel-/long-format hvor gjentakende variabelobservasjoner organiseres vertikalt på record-nivå. Den nye kommandoen `reshape-from-panel` gjør det mulig å endre datastrukturen til wide-format der opplysningene struktureres horisontalt på variabelnivå med én record per enhet.

Alle variabler i paneldatasettet du står i restruktureres til wide-format etter at kommandoen er kjørt, og variablene får suffiks basert på undernivået gitt ved hjelpevariabelen date@panel[6]. Merk at også variabler for faste opplysninger vil dupliseres med suffiks knyttet til undernivå (selv om de ikke endrer seg over tid). Dette kan løses ved å slette overflødige variabler etter at datasettet er ferdig konvertert.

Illustrasjonen nedenfor viser hvordan restruktureringen logisk foregår under panseret. Eksempelet viser et datasett med long-format som inneholder variablene sivstand, lønn og kjønn, i tillegg til hjelpevariabelen date@panel som inneholder verdien til undernivået, i dette tilfellet årene 2018-2020 (tosifret). Datasettet konverteres til wide-format ved hjelp av kommandoen `reshape-from-panel`. Merk at man ikke spesifiserer variabler eller prefiks.

---

[6] For klassiske paneldatasett som lages ved bruk av kommandoen `import-panel`, vil suffiksene bli litt annerledes enn forventet. Når man bruker `tabulate-panel` eller `summarize-panel` på slike datasett, vil det se ut som at undernivået har verdier av typen «YYYY-MM-DD», men dette gjelder bare som visningsformat. De faktiske verdiene for date@panel bruker i dette tilfellet referansedatoer som verdiformat (antall dager målt fra 1/1 1970). Dette løser man ved å døpe om variabelnavnene med kommandoen `rename` etterpå.

Alle variabler gjøres om til wide-format med tilhørende suffiks, inkludert variabler som måler faste opplysninger som kjønn.

ID	date@panel	sivstand	lønn	kjønn
1	18	1	120000	1
2	18	1	300000	2
3	18	2	400000	1
4	18	2	160000	2
1	19	1	150000	1
2	19	1	340000	2
3	19	2	600000	1
4	19	4	170000	2
1	20	2	180000	1
2	20	1	360000	2
3	20	2	630000	1
4	20	4	175000	2

ID	sivstand18	sivstand19	sivstand20	lønn18	lønn19	lønn20	kjønn18	kjønn19	kjønn20
1	1	1	2	120000	150000	180000	1	1	1
2	1	1	1	300000	340000	360000	2	2	2
3	2	2	2	400000	600000	630000	1	1	1
4	2	4	4	160000	170000	175000	2	2	2

Kommandoen `reshape-from-panel` kompletterer ved å gjøre det mulig å konvertere frem og tilbake mellom wide- og long-format, noe som gir følgende muligheter:

- Det er ikke mulig å importere nye variabler inn i et datasett opprettet ved hjelp av `import-panel`. Dette kan løses ved å bruke `reshape-from-panel` til å konvertere til wide-format, for så å importere nye variabler etter behov ved hjelp av `import`. Når man har de variablene man trenger, kan man konvertere tilbake til panel-/long-format igjen gjennom å bruke kommandoen `reshape-to-panel`.

- Paneldatasett gir mindre fleksibilitet når man skal sammenlikne og gjøre operasjoner over variabelverdier på tvers av undernivå (over tid). Eksempler på dette er når man vil lage en variabel som består av gjennomsnittet av lønn i 2019 og 2020, eller når man vil lage en betingelse som baserer seg på tilfeller der lønn i 2020 er større enn 2019. Også dette kan løses ved å konvertere til wide-format, for så å gjøre de ønskede operasjoner og konvertere tilbake etterpå.

Se kapittel 2.13 for eksempel på hvordan dette gjøres i praksis.

2.10 Eksempler: Oppretting og justering av et datasett

```
textblock
Henter først variablene en trenger

Først kobler en seg mot databanken, deretter opprettes et datasett som kalles `demografidata` ved hjelp av `create-dataset <dataset>`. Da vil all import og all bearbeiding foregå mot dette med mindre en aktivt skifter datasett med kommandoen `use <dataset>`

Tilhørende start- og stoppdatoer følger også med ved import
endblock

require no.ssb.fdb:12 as db

create-dataset demografidata
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/SIVSTANDFDT_SIVSTAND 2015-01-01 as sivstand
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue

// Endrer navn på variabler ved å legge til årsangivelse
rename sivstand sivstand15
rename formue formue15

// Sletter variabelen kjonn fra datasettet
drop kjønn

// Beholder kun gifte personer i datasettet
keep if sivstand15 == '2'
```

2.11 Eksempler: Sammenkoblinger av data på andre nivå enn person

Eksempel: Sammenkobling av foreldreopplysninger

```
textblock
Hvordan benytte foreldreopplysninger i analyser
-----------------------------------------------------------

I databanken finnes det variabler for fars og mors fødselsnumre, noe som gjør at en kan koble
opplysninger om foreldre på et persondatasett.

Gjennom kommandoen `merge` kan en koble sammen datasett. Nøkkelidentifikatorvariabelen i
måldatasettet brukes som standard. Dette kan imidlertid overstyres gjennom en "on-opsjon".

I eksempelet lages det et eget datasett for fedre og mødre, som kobles sammen med et
persondatasett via koblingsnøklene `fnr_far` og `fnr_mor`.
endblock

//Kobler til databank
require no.ssb.fdb:12 as db

//Lager et persondatasett med lenker til far og mor
create-dataset persondata
import db/INNTEKT_WYRKINNT 2019-01-01 as inntekt
import db/BEFOLKNING_KJOENN as kjønn
import db/NUDB_BU 2019-01-01 as utd
import db/BEFOLKNING_FAR_FNR as fnr_far
import db/BEFOLKNING_MOR_FNR as fnr_mor

//Henter opplysninger om far og kobler på persondatasett
create-dataset foreldredata
import db/INNTEKT_WYRKINNT 2019-01-01 as inntekt_far
import db/NUDB_BU 2019-01-01 as utd_far
clone-variables inntekt_far -> inntekt_mor
clone-variables utd_far -> utd_mor
merge inntekt_far utd_far into persondata on fnr_far
merge inntekt_mor utd_mor into persondata on fnr_mor

//Kjører en enkel lineær regresjon for å teste sammenheng mellom egen og foreldres inntekt
```

```
use persondata
generate mann = 0
replace mann = 1 if kjønn == '1'

destring utd
generate høyutd = 0
replace høyutd = 1 if utd >= 700000 & utd < 900000
replace høyutd = utd if sysmiss(utd)

destring utd_far
generate høyutd_far = 0
replace høyutd_far = 1 if utd_far >= 700000 & utd_far < 900000
replace høyutd_far = utd_far if sysmiss(utd_far)

destring utd_mor
generate høyutd_mor = 0
replace høyutd_mor = 1 if utd_mor >= 700000 & utd_mor < 900000
replace høyutd_mor = utd_mor if sysmiss(utd_mor)

summarize inntekt inntekt_far inntekt_mor
histogram inntekt_far, percent
histogram inntekt_mor, percent
correlate inntekt_far inntekt_mor
tabulate høyutd_far høyutd_mor

regress inntekt mann inntekt_far inntekt_mor høyutd høyutd_far høyutd_mor
```

Eksempel: Sammenkobling av data på familienivå

```
textblock
Aggregere opplysninger til familienivå
----------------------------------------

Individer kan knyttes opp mot et familienummer som kan brukes til å aggregere opplysninger på familienivå. Individer tilhørende samme familie vil være registrert med det samme familienummeret som består av person-id'en til den eldste personen i familien.

I eksempelet opprettes først et persondatasett der en filtrerer ned på person i familier bestående av ektepar med små barn (kode 2.1.1). Deretter kobles det på demografiske opplysninger.

Familieinntekt er en opplysning på familienivå, dvs. familie = enhet. Derfor må en opprette et nytt datasett for dette formålet (datasett kan ikke bestå av variabler med ulike enhetstyper). En importerer da yrkesinntekt på personnivå, og bruker så kommandoen `collapse (sum)` til å summere inntektene på familienivå (`by(famnr)`). Resultatet blir et datasett med familie som enhet.

Til slutt kobles familieinntekt på persondatasettet vha. kommandoen `merge`.
endblock

//Kobler til databank
require no.ssb.fdb:12 as db

//Oppretter først et persondatasett for personer i familier bestående av ektepar med små barn
create-dataset persondata
import db/BEFOLKNING_REGSTAT_FAMTYP 2017-01-01 as famtype
tabulate famtype
keep if famtype == '2.1.1'

//Legger til diverse demografiske opplysninger
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate alder = 2017 - int(faarmnd/100)

import db/BEFOLKNING_KOMMNR_FAKTISK 2017-01-01 as bosted
generate fylke = substr(bosted, 1, 2)

import db/BEFOLKNING_BARN_I_HUSH 2017-01-01 as antbarn


//Oppretter datasett for generering av total yrkesinntekt per familie => enhet = familie
create-dataset familiedata
```

```
import db/BEFOLKNING_REGSTAT_FAMNR 2017-01-01 as famnr
import db/INNTEKT_WYRKINNT 2017-01-01 as yrkesinnt
collapse (sum) yrkesinnt, by(famnr)
rename yrkesinnt familieinnt

//Kobler familieinntekt på persondatasettet (enhet = personer)
merge familieinnt into persondata on PERSONID_1

//Lager familiestatistikk. Familienummeret består av person-id til eldste person i familien, så når en fjerner individer med manglende familieinntekt sitter en igjen med et datasett med familie som enhet. Alle personopplysninger vil da gjelde for eldste person i familien
use persondata
drop if sysmiss(familieinnt)

rename alder alder_eldst
rename kjønn kjønn_eldst

define-labels fylketekst '01' Østfold '02' Akershus '03' Oslo '04' Hedmark '05' Oppland '06' Buskerud '07' Vestfold '08' Telemark '09' 'Aust-Agder' '10' 'Vest-Agder' '11' Rogaland '12' Hordaland '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag' '18' Nordland '19' Troms '20' Finnmark '21' Spitsbergen '25' 'Studerer i utlandet' '99' Uoppgitt
assign-labels fylke fylketekst

tabulate fylke

histogram alder_eldst, discrete
histogram antbarn, discrete percent

tabulate antbarn
tabulate antbarn, cellpct
tabulate antbarn kjønn_eldst

summarize familieinnt
barchart (mean) familieinnt, by(fylke)
barchart (mean) familieinnt, by(antbarn)
histogram familieinnt, freq
histogram familieinnt, by(antbarn) percent
```

Eksempel: Sammenkobling/uthenting av data på kursnivå

```
textblock
Hente ut informasjon om pågående utdanning (kurs)
-------------------------------------------------

Opplysninger om pågående utdanning (såkalte kursdata) foreligger med kurs som enhetsnivå (med tilhørende kurs-identifikator). Kurs er gitt ved kombinasjonen person x kurstype, der hvert individ i praksis kan være representert med flere kurstyper til samme tid.

Siden kursdata ikke har person som enhet, kan ikke disse importeres inn i persondatasett på vanlig måte, men må kobles på vha. kommandoen `merge`.

Først må en legge til en lenke mellom kurs-id og person-id på kursdataene. Deretter må en aggregere opp til personnivå vha. kommandoen `collapse` før en tilslutt kobler på persondatasettet.

I eksempelet opprettes det først et persondatasett bestående av personer bosatt i Norge (regstatus == '1') per 2019-01-01. Deretter hentes det ut historikk over pågående utdanning for hele året 2019, der en tar vare på utdanning på høyere nivå (master eller høyere, nivå 7 og 8). Kommandoen `collapse (count)` brukes til å telle opp antall observasjoner med pågående utdanning per individ over året 2019, og resultatet kobles så på persondatasettet for videre analyse.

NB! Merk at variabelen `kurstype` etter bruk av `collapse` vil bestå av verdier for den aktuelle statistikken som blir kjørt, i dette tilfellet antall observasjoner (`count`).
endblock

//Kobler til databank
require no.ssb.fdb:12 as db

//Oppretter persondatasett for bosatte per 2019-01-01
create-dataset bosatte
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_STATUSKODE 2019-01-01 as regstatus
keep if regstatus == '1'

//Henter personer som tar høyere utdanning i løpet av 2019
create-dataset kursdata
import-event db/NUDB_KURS_NUS 2019-01-01 to 2019-12-31 as kurstype
destring kurstype
keep if kurstype >= 700000 & kurstype < 900000
```

```
//Kobler på lenke mellom kurs-id og fødselsnumre
create-dataset lenke_kurs_person
import db/NUDB_KURS_FNR as fnr
merge fnr into kursdata

//Lager statistikk (collapser) over antall hendelser med høy utdanning per individ, og kobler dette på
persondatasettet
use kursdata
collapse (count) kurstype, by(fnr)
rename kurstype ant_kurs
merge ant_kurs into bosatte

//Lager tabell over antall personer som tok høy utdanning i løpet av 2019
use bosatte
generate utdanning_høy = 0
replace utdanning_høy = 1 if ant_kurs >= 1
tabulate utdanning_høy kjønn
```

Eksempel: Sammenkobling/uthenting av data på kursnivå for en gitt dato

```
textblock
Hente ut informasjon om pågående utdanning (kurs) for en gitt dato
------------------------------------------------

Opplysninger om pågående utdanning (såkalte kursdata) foreligger med kurs som enhetsnivå (med
tilhørende kurs-identifikator). Kurs er gitt ved kombinasjonen person x kurstype, der hvert individ i
praksis kan være representert med flere kurstyper til samme tid.

Siden kursdata ikke har person som enhet, kan ikke disse importeres inn i persondatasett på vanlig
måte, men må kobles på vha. kommandoen merge.

Først må en legge til en lenke mellom kurs-id og person-id på kursdataene. Deretter må en aggregere
opp til personnivå vha. kommandoen collapse før en tilslutt kobler på persondatasettet.

I eksempelet opprettes det først et persondatasett bestående av personer bosatt i Norge (regstatus ==
'1') per 2019-01-01. Deretter opprettes et nytt datasett der en importerer status for pågående
utdanning per 2019-11-01. Kommandoen collapse (count) brukes til å telle opp antall observasjoner
med pågående utdanning per individ for den gitte datoen, og resultatet kobles så på persondatasettet
for videre analyse.
```

```
NB! Merk at variabelen kurstype etter bruk av collapse vil bestå av verdier for den aktuelle statistikken som blir kjørt, i dette tilfellet antall observasjoner (count).
endblock

//Kobler til databank
require no.ssb.fdb:12 as db

//Oppretter persondatasett for bosatte per 2019-01-01
create-dataset bosatte
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_STATUSKODE 2019-01-01 as regstatus
keep if regstatus == '1'

//Henter personer som tar utdanning per 1. november 2019, og kobler dette på persondatasettet. Siden kursdata kan ha flere observasjoner per individ, må kommandoen collapse brukes til å aggregere opp til personnivå. Vi bruker count som aggregeringsverdi (antall records)
create-dataset kursdata
import db/NUDB_KURS_NUS 2019-11-01 as kurstype
import db/NUDB_KURS_FNR as fnr
collapse (count) kurstype, by(fnr)
rename kurstype ant_kurs
merge ant_kurs into bosatte

//Lager tabell over antall personer som studerer per 1. november 2019
use bosatte
generate studerer = 0
replace studerer = 1 if ant_kurs >= 1
tabulate studerer kjønn
```

2.12 Eksempler: Hvordan restrukturere datasett fra tverrsnitts- til paneldata-format (fra “wide” til “long”)

```
require no.ssb.fdb:17 as db

//Lager først et vanlig wide-datasett bestående av et 1% utvalg av alle bosatte per 1/1 2018
create-dataset wide
import db/BEFOLKNING_STATUSKODE 2018-01-01 as regstat18
keep if regstat18 == '1'
```

```
sample 0.01 333
import db/BEFOLKNING_STATUSKODE 2019-01-01 as regstat19
import db/BEFOLKNING_STATUSKODE 2020-01-01 as regstat20
import db/SIVSTANDFDT_SIVSTAND 2018-01-01 as sivstand18
import db/SIVSTANDFDT_SIVSTAND 2019-01-01 as sivstand19
import db/SIVSTANDFDT_SIVSTAND 2020-01-01 as sivstand20
import db/BEFOLKNING_KJOENN as kjønn
import db/INNTEKT_WLONN 2018-01-01 as lønn18
import db/INNTEKT_WLONN 2019-01-01 as lønn19
import db/INNTEKT_WLONN 2020-01-01 as lønn20

//Kjører litt opptellinger og statistikk
tabulate regstat18, missing
tabulate regstat19, missing
tabulate regstat20, missing
tabulate sivstand18, missing
tabulate sivstand19, missing
tabulate sivstand20, missing
tabulate kjønn, missing

summarize lønn18 lønn19 lønn20

//Konverterer til long-format (paneldata)
reshape-to-panel regstat sivstand lønn

//Tester om det er samsvar med opptellingene på wide-datasettet
tabulate date@panel, missing

tabulate-panel regstat, missing
tabulate-panel sivstand, missing
tabulate-panel regstat sivstand, missing

tabulate-panel kjønn, missing
tabulate-panel regstat kjønn, missing
tabulate-panel sivstand kjønn, missing

summarize lønn
summarize-panel lønn
```

2.13 Eksempler: Hvordan restrukturere datasett fra paneldata- til tverrsnitts-format (fra “long” til “wide”)

```
require no.ssb.fdb:17 as db

//Lager et wide-datasett bestående av 1% av alle bosatte per 1/1 2019
create-dataset wide
import db/BEFOLKNING_STATUSKODE 2019-01-01 as regstat19
keep if regstat19 == '1'
sample 0.01 333
import db/BEFOLKNING_STATUSKODE 2020-01-01 as regstat20
import db/SIVSTANDFDT_SIVSTAND 2019-01-01 as sivstand19
import db/SIVSTANDFDT_SIVSTAND 2020-01-01 as sivstand20
import db/BEFOLKNING_KJOENN as kjønn
import db/INNTEKT_WLONN 2019-01-01 as lønn19
import db/INNTEKT_WLONN 2020-01-01 as lønn20

tabulate regstat19, missing
tabulate regstat20, missing
tabulate sivstand19, missing
tabulate sivstand20, missing
tabulate kjønn, missing

summarize lønn19 lønn20

//Restrukturerer til panel-/long-format
reshape-to-panel regstat sivstand lønn

tabulate-panel regstat, missing
tabulate-panel sivstand, missing
tabulate-panel kjønn, missing
summarize-panel lønn

//Restrukturerer tilbake til wide-format
reshape-from-panel
drop kjønn20
rename kjønn19 kjønn
```

```
tabulate regstat19, missing
tabulate regstat20, missing
tabulate sivstand19, missing
tabulate sivstand20, missing
tabulate kjønn, missing
summarize lønn19 lønn20


//Lager et nytt paneldatasett for samme populasjon vha. import-panel
clone-units wide paneltest
use paneltest
import-panel db/BEFOLKNING_STATUSKODE db/SIVSTANDFDT_SIVSTAND db/INNTEKT_WLONN
db/BEFOLKNING_KJOENN 2019-01-01 2020-01-01
rename BEFOLKNING_STATUSKODE regstat
rename SIVSTANDFDT_SIVSTAND sivstand
rename INNTEKT_WLONN lønn
rename BEFOLKNING_KJOENN kjønn

tabulate-panel regstat, missing
tabulate-panel sivstand, missing
tabulate-panel kjønn, missing
summarize-panel lønn

//Restrukturerer til wide-format
reshape-from-panel
drop kjønn18262
rename kjønn17897 kjønn

rename regstat17897 regstat19
rename regstat18262 regstat20

rename sivstand17897 sivstand19
rename sivstand18262 sivstand20

rename lønn17897 lønn19
rename lønn18262 lønn20

summarize lønn19 lønn20
```

3. Tilrettelegging av variabler

De fleste variabler som importeres til et datasett må kodes om før videre analyse. Dessuten har en som regel behov for å lage egne variabler basert på de importerte opplysningene. Dette gjøres gjennom kommandoene `generate`, `replace` og `recode`.

3.1 Opprettelse av nye variabler og omkoding: generate/replace

For å lage en ny variabel brukes kommandoen `generate,` der en spesifiserer navnet og hvilken verdi den skal ha. Dette kan være en konkret verdi eller en verdi basert på en likning/formel. IF-betingelser brukes til å angi hvilke tilfeller/enheter som skal få en verdi.

Merk at kommandoen `generate` bare kan brukes til å angi én verdi. Vil en legge til flere koder, kan `replace`-kommandoen benyttes i videre steg for å fullføre prosessen.

En kan også bruke `generate` til å kopiere andre variabler: `generate` <*nyvar*> = <*gammelvar*>. Også dette kan kombineres med IF-betingelser.

Eksempel på koding av dummyen "mann":

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 1
replace mann = 0 if kjønn != '1'
```

Det er mange mulige måter å lage logiske betingelser på, som alle vil gi samme resultat. En kunne alternativt kodet på følgende måte:

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 0
replace mann = 1 if kjønn == '1'
```

Merk følgende når en koder om eller lager nye variabler:

- "=" brukes når verdier settes gjennom generate eller replace. "==" brukes ved logiske IF-betingelser.

- Verdier for alfanumeriske variabler må angis med "enkeltfnutter" (`'1', '2', …` etc), mens for numeriske variabler angis verdiene som rene tall uten "fnutter" (`1, 2, ….` etc).
  - Variabelformatet finner en ved å se på den aktuelle variabelen øverst til venstre (datasettvinduet) eller nederst til venstre (registervariabelvinduet).

- Kode for manglende data ("missingverdi") angis på følgende måte: `sysmiss( <variabel> )`
  - Eksempel (en filtrerer bort records/enheter med manglende verdier for kjønn):

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 1
replace mann = 0 if kjønn != '1'
drop if sysmiss( kjønn )
```

- Følgende logiske operatorer kan brukes i forbindelse med IF-betingelser:

  - Større enn >
  - Mindre enn <
  - Er lik ==
  - Større enn eller lik >=
  - Mindre enn eller lik <=
  - Er ulik !=
  - Eller |
  - Og &

- Dummyvariabler MÅ av metodiske grunner være numeriske og bestå av verdiene 1 og 0. En kan altså ikke ha en dummyvariabel med kun verdien 1. Dette vil generere uønskede resultat, eller feilmelding når en kjører regresjonsanalyser. I praksis må en derfor passe på å kode alle enheter som ikke har "suksess"-verdien med verdien 0 (se eksempel øverst på forrige side).

- Ved bruk av dummyvariabler i IF-betingelser, trenger en ikke angi verdien 1.
  - Eksempel: I stedet for `tabulate sivstand if mann == 1`, kan en skrive `tabulate sivstand if mann`

- Om hensikten med tilretteleggingen av variablene er å benytte regresjonsanalyser, bør kategoriske verdier kodes på numerisk form. Hvis ikke risikerer en at systemet ikke godtar variabelinputen, og at en får feilmelding når en kjører kommandoer som `regress`, `logit`, etc.

- Av metodiske grunner bør kategoriske variabler vanligvis tilrettelegges som dummyvariabler slik som i eksempelet med variabelen “mann” ovenfor. Dette gjelder også flerkategorivariabler (mer enn to kategorier) som f.eks. “Utdanningsnivå”. I slike tilfeller lager en et sett med dummyvariabler som i kombinasjon tilsvarer flerkategorivariabelen. I praksis vil hver kategori minus referanse-/basis-kategorien representeres ved separate dummyvariabler, der en tolkningsmessig måler effekten av de enkelte kategorier sammenliknet med referansekategorien. Prosessen med å lage sett av dummyvariabler kan automatiseres gjennom å bruke prefikset “i.” foran variabelnavnet i de ulike regresjons-uttrykkene. Da benyttes automatisk den laveste verdi som referanseverdi.

- Missingverdier: Vær obs på at alle enheter der minst én av variablene har en missingverdi blir ekskludert fra regresjonskjøringer. Variabler med mange missingverdier som ikke blir kodet om vil da kunne føre til at regresjonsanalysen blir utført på et mye mindre datasett enn planlagt. Dette er noe en bør være klar over under tilretteleggingen. I eksempelet med kjønn vil det typisk være få enheter/individer med missingverdi, men det kan være andre variabler som angir f.eks. trygdeytelser som “Uføregrad”. Et flertall vil her ha missingverdi, og bare dem som er uføre vil ha en gyldig verdi. En bør da kode på følgende måte:

```
import PENSJONER_UFOERGRAD 2010-01-01 as uførgrad
generate ufør = 1
replace ufør = 0 if sysmiss( uførgrad )
```

- Missingverdier for inntektsvariabler: Dette vil typisk gjelde alle personer med inntekt = 0. Hvis også disse bør være med i analysen, må de kodes om til 0-verdier:

```
replace inntekt = 0 if sysmiss(inntekt)
```

3.1.1 Komprimert omkoding: inlist

Ved koding/omkoding av variabler basert på lange if-betingelser, kan funksjonen `inlist` være nyttig å bruke. Et eksempel er om man ønsker å lage en variabel som tar verdien 1 for personer som er bosatt i et utvalg av 100 kommuner. Dette ville gitt en generate- evt. replace-kommando med et svært langt if-uttrykk. Ved bruk av `inlist` kan man gjøre dette mer komprimert ved å liste opp alle kommunenumrene separert med komma inni en funksjonsparentes.

Den logiske funksjonen `inlist` settes til 1 ("true") dersom verdien i det første argumentet finnes blant de resterende argumentene.

Argumentene til funksjonen kan være både variabler og verdier (alle typer).

Eksempler:

- `generate var = 1 if inlist(sivstand, 1, 3, 5)`

  (inlist-funksjonen = 1 ("true") dersom sivstand = 1, 3 eller 5)

- `generate var = 1 if inlist('1', regstat09, regstat10, regstat11)`

  (inlist-funksjonen = 1 ("true") dersom minst en av regstat-variablene = '1')

Eksemplene over tilsvarer disse uttrykkene:

- `generate var = 1 if sivstand == 1 | sivstand == 3 | sivstand == 5`

- `generate var = 1 if regstat09 == '1' | regstat10 == '1' | regstat11 == '1'`

3.1.2 Komprimert omkoding: inrange

I tillegg til `inlist`, finnes det en annen logisk funksjon som kan gjøre det lettere å lage betingelser basert på intervaller: `inrange`. If-betingelser som baserer seg på intervaller blir sjeldent særlig lange, siden man som oftest bare trenger å spesifisere en minimums- og en maksimumsverdi. Men dersom man opererer med flere intervaller i ett og samme if-uttrykk, kan `inrange` være nyttig. Uansett blir generate- evt. replace-uttrykket mer komprimert i forhold til vanlige if-uttrykk ved bruk av denne funksjonen.

Den logiske funksjonen `inrange` settes til 1 ("true") dersom verdien i det første argumentet er høyere enn eller lik verdien i det andre argumentet og lavere enn eller lik verdien i det tredje argumentet.

Argumentene til funksjonen kan være både variabler og verdier (alle typer).

Eksempel:

- `generate var = 1 if inrange(formue, 500000, 1000000)`

  (inrange-funksjonen = 1 ("true") dersom 500000 <= formue <= 1000000)

Eksempelet over tilsvarer dette uttrykket:

- `generate var = 1 if formue >= 500000 & formue <= 1000000`

3.2 Omkoding av variabler: recode

Alternativt til kommandoen `replace`, kan `recode` benyttes til å kode om variabler. For variabler med mange verdier som skal omkodes, er denne kommandoen et nyttig verktøy. Omkodingen kan da i mange tilfeller fullføres med en enkelt kommandolinje, noe som bidrar til kortere tidsbruk for prosesseringen. En kan dessuten bruke kommandoen til å kode om flere variabler om gangen.

Eksempel på koding av variabelen "mann" ved hjelp av kommandoen `recode`:

```
import BEFOLKNING_KJOENN as mann
destring mann
recode mann (2 = 0)
```

Det er mulig å bruke `recode` både på numeriske og alfanumeriske variabler, og man kan også opprette verdi-labler for de omkodete verdiene inni selve recode-uttrykket.

Eksempler på måter å kode om grupper av tall for variablene *var1* og *var2*:

```
recode var1 var2 (1 2 3 = 0)
```
(verdiene 1-3 kodes til 0)

```
recode var1 var2 (1/7 = 0)
```
(verdiene 1-7 kodes til 0)

```
recode var1 var2 (1/7 = 0) (nonmissing = 1) (missing = 99)
```
(øvrige gyldige verdier kodes til 1, missingverdier kodes til 99)

```
recode var1 var2 (1/7 = 0)(* = 99)
```
(alle andre verdier kodes til 99)

```
recode var1 var2 (min/100 = 1)(101/max = 2)
```
(alle verdier til og med 100 kodes til 1, alle verdier fra og med 101 kodes til 2)

```
recode regstat ('3' '5' = '0' 'ikke-bosatt')
```
(verdiene ‘3’ og ‘5’ for den alfanumeriske variabelen `regstat` kodes til ‘0’ og gis verdi-labelen “ikke-bosatt”)

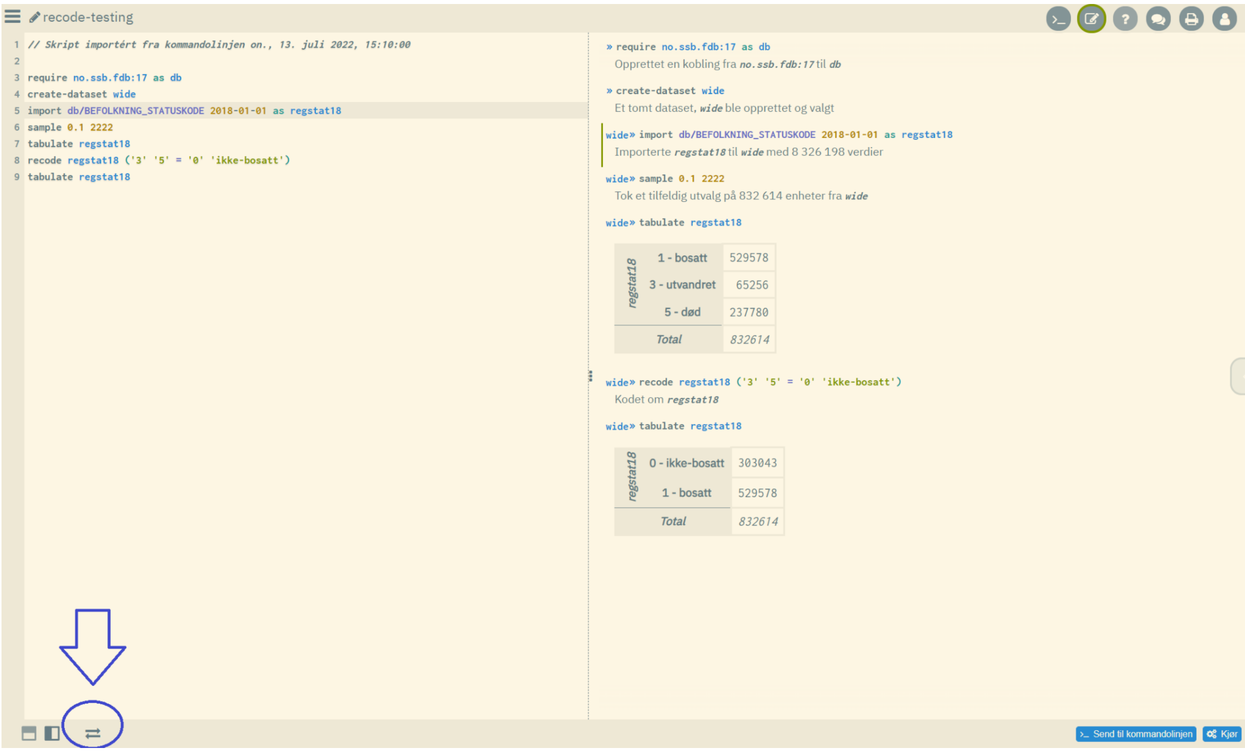
Merk følgende:
- Parametrene `min` og `max` kan bare benyttes i forbindelse med intervallangivelser, slik som i eksempelet over.
- Alfanumeriske verdier kan ikke inngå i intervallangivelser, bare som enkeltverdier, jamfør eksempelet over.
- Alfanumeriske verdier og verdi-labler kan angis både med enkeltfnutter og dobbeltfnutter/hermetegn

For mer informasjon om `recode`, bruk kommandoen `help recode`. Dette viser også en oversikt over tilhørende opsjoner.

3.2.1 Automatisk omkoding ved hjelp av opplasting av skilletegnseparerte filer

I tilfeller hvor man ønsker å omkode mange variabelverdier, f.eks. kode om alle kommunekoder og erstatte med koder for sentralitet, eller kode om fra en næringsstandard til en annen, vil det være tidkrevende og tungvint å legge inn hver enkelt kode i et `recode`-uttrykk. Slike uttrykk vil typisk utgjøre mange linjer avhengig av hvor mange koder som skal kodes om.

I microdata.no finnes det en løsning for dette. Ved å klikke på pilsymbolet nede til venstre i skriptvinduet, vises det en dialogboks som gir deg mulighet til å laste opp en skilletegnseparert fil som du på forhånd har lastet ned fra f.eks. SSB sin standardklassifikasjonsside: ssb.no/klass


recode-testing
// Skript importért fra kommandolinjen on., 13. juli 2022, 15:10:00
require no.ssb.fdb:17 as db
create-dataset wide
import db/BEFOLKNING_STATUSKODE 2018-01-01 as regstat18
sample 0.1 2222
tabulate regstat18
recode regstat18 ('3' '5' = '0' 'ikke-bosatt')
tabulate regstat18
» require no.ssb.fdb:17 as db
Opprettet en kobling fra no.ssb.fdb:17 til db
» create-dataset wide
Et tomt dataset, wide ble opprettet og valgt
wide» import db/BEFOLKNING_STATUSKODE 2018-01-01 as regstat18
Importerte regstat18 til wide med 8 326 198 verdier
wide» sample 0.1 2222
Tok et tilfeldig utvalg på 832 614 enheter fra wide
wide» tabulate regstat18
regstat18
1 - bosatt 529578
3 - utvandret 65256
5 - død 237780
Total 832614
wide» recode regstat18 ('3' '5' = '0' 'ikke-bosatt')
Kodet om regstat18
wide» tabulate regstat18
regstat18
0 - ikke-bosatt 303043
1 - bosatt 529578
Total 832614
Send til kommandolinjen
Kjør

Spesifikasjonene for omkodingsfiler går frem av forklaringen i dialogboksen. Blant annet må filen være av typen semikolon-separert (.csv), og ha fire kolonner som innholder følgende:

- Fra-kode
- Fra-navn (label)
- Til-kode
- Til-navn (label) (valgfri)

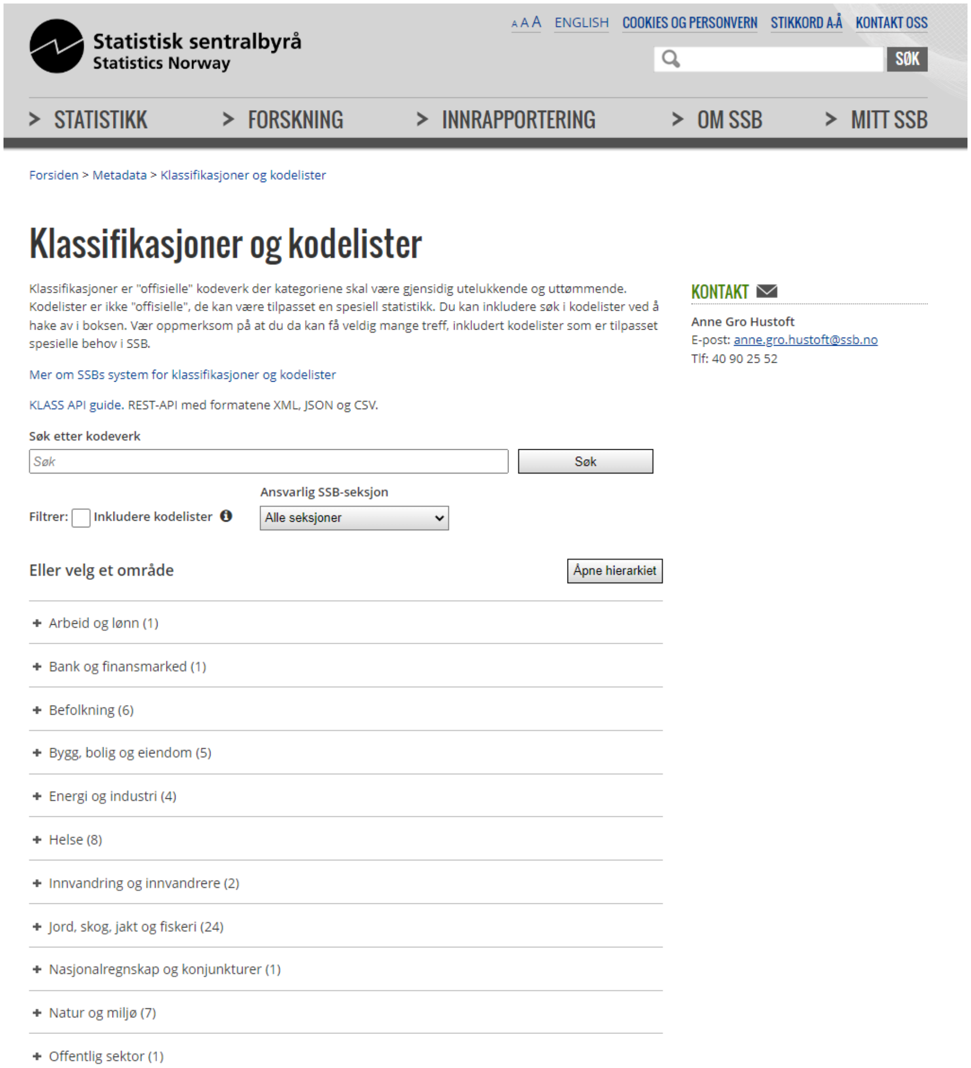
Funksjonaliteten er tilpasset SSB sine standard korrespondansefiler, men man kan også bruke andre filer evt. lage egne, så lenge de har riktig format og oppbygning.

Når man har lastet opp en omkodingsfil, vil microdata.no automatisk generere et `recode`-uttrykk nederst i ditt aktive skript, slik som dette (det eneste du behøver å gjøre etterpå er å bytte ut standard-variabelnavnet “variable” med navnet på variabelen du skal omkode):

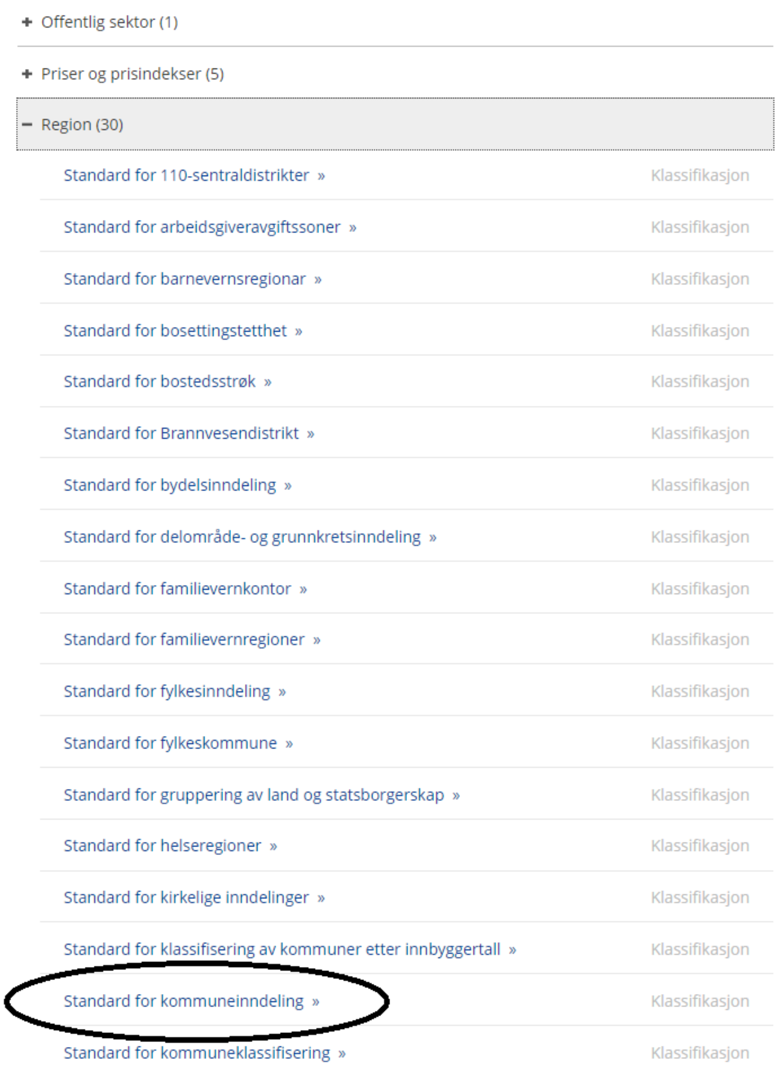
```
recode variable ('0301' '3020' '3024' '3027' '3029' '3030' = '01' 'Sentralitet: 1 (925-1000) -
høy') ('1103' '1108' '3002' '3003' '3004' '3005' '3019' '3021' '3022' '3025' '3031' '3032'
'3033' '3049' '3403' '3801' '3803' '4601' '5001' = '02' 'Sentralitet: 2 (870-924)') ('1106'
'1120' '1121' '1122' '1124' '1127' '1507' '1804' '3001' '3006' '3007' '3014' '3015' '3016'
'3017' '3018' '3023' '3026' '3028' '3034' '3035' '3036' '3038' '3047' '3048' '3053' '3054'
'3401' '3405' '3407' '3411' '3412' '3413' '3420' '3443' '3446' '3802' '3804' '3805' '3806'
'3807' '3811' '3813' '4202' '4203' '4204' '4215' '4627' '5031' '5035' '5401' = '03'
'Sentralitet: 3 (775-869)') ('1101' '1114' '1119' '1130' '1146' '1149' '1505' '1506' '1516'
'1517' '1520' '1528' '1531' '1532' '1577' '1806' '1824' '1833' '1841' '1870' '3011' '3013'
'3037' '3041' '3050' '3414' '3415' '3416' '3440' '3441' '3442' '3447' '3448' '3451' '3808'
'3812' '3814' '3816' '3817' '4201' '4205' '4206' '4207' '4213' '4214' '4216' '4219' '4223'
'4225' '4612' '4614' '4621' '4622' '4623' '4624' '4626' '4630' '4631' '4647' '5006' '5007'
'5028' '5029' '5036' '5037' '5038' '5053' '5059' '5402' '5403' '5406' = '04' 'Sentralitet: 4
(670-774)') ('1111' '1112' '1135' '1145' '1160' '1515' '1525' '1535' '1539' '1547' '1554' '1557'
'1560' '1563' '1566' '1579' '1813' '1820' '1840' '1860' '1865' '1866' '1868' '3012' '3039'
'3040' '3042' '3043' '3044' '3045' '3046' '3051' '3417' '3418' '3419' '3421' '3422' '3427'
'3428' '3430' '3435' '3436' '3437' '3438' '3439' '3449' '3450' '3452' '3453' '3815' '3818'
'3819' '3820' '3821' '4211' '4212' '4217' '4218' '4226' '4227' '4228' '4602' '4611' '4613'
'4615' '4617' '4618' '4625' '4628' '4632' '4640' '4643' '4649' '4650' '4651' '5021' '5022'
'5025' '5027' '5032' '5034' '5045' '5047' '5054' '5055' '5057' '5061' '5405' '5411' '5416'
'5418' '5419' '5421' '5428' '5437' '5444' = '05' 'Sentralitet: 5 (565-669)') ('1133' '1134'
'1144' '1151' '1511' '1514' '1573' '1576' '1578' '1811' '1812' '1815' '1816' '1818' '1822'
'1825' '1826' '1827' '1828' '1832' '1834' '1835' '1836' '1837' '1838' '1839' '1845' '1848'
'1851' '1853' '1856' '1857' '1859' '1867' '1871' '1874' '1875' '3052' '3423' '3424' '3425'
'3426' '3429' '3431' '3432' '3433' '3434' '3454' '3822' '3823' '3824' '3825' '4220' '4221'
'4222' '4224' '4616' '4619' '4620' '4629' '4633' '4634' '4635' '4636' '4637' '4638' '4639'
'4641' '4642' '4644' '4645' '4646' '4648' '5014' '5020' '5026' '5033' '5041' '5042' '5043'
'5044' '5046' '5049' '5052' '5056' '5058' '5060' '5404' '5412' '5413' '5414' '5415' '5417'
'5420' '5422' '5423' '5424' '5425' '5426' '5427' '5429' '5430' '5432' '5433' '5434' '5435'
'5436' '5438' '5439' '5440' '5441' '5442' '5443' = '06' 'Sentralitet: 6 (0-564) - lav')
```

<u>Oppskrift for å laste ned korrespondansefil fra SSB:</u>

Trinn 1: Gå til ssb.no/klass



Trinn 2: Velg emnet du ønsker å hente omkodingsfil fra, f.eks. “Standard for kommuneinndeling” som du finner under “Region”



Trinn 3: Velg fanen "Korrespondanser" og deretter den omkodingen du er interessert i, f.eks. "Sentralitet 2020"

Gjeldende versjon: (Gyldig fra og med januar 2020)

Kommuneinndeling 2020

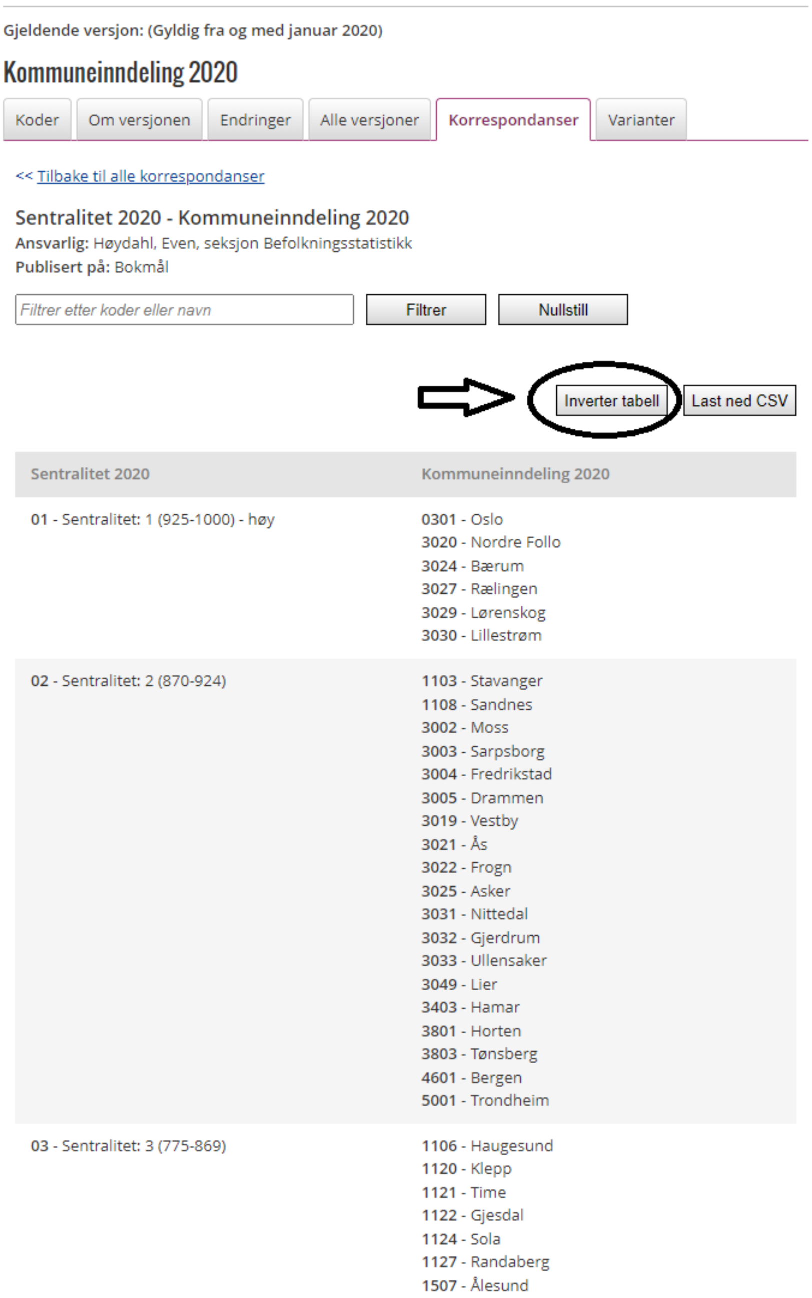
Koder | Om versjonen | Endringer | Alle versjoner | Korrespondanser | Varianter

Korrespondanser

Korrespondansetabellen viser sammenhengen mellom versjoner av to ulike kodeverk, f.eks. sammenhengen mellom Politidistrikt 2016 og Kommuneinndeling 2014 (hvilke kommuner tilhører hvilke politidistrikt). Dersom du ønsker å se forskjellen mellom to påfølgende versjoner av samme kodeliste, f.eks. mellom Kommuneinndeling 2014 og Kommuneinndeling 2013, finner du den under fanen «Endringer».

Korrespondanser fra	Nivå	Korrespondanser til	Nivå	Eier
Kommuneinndeling 2020	Alle	Bydelsinndeling 2020	Alle	Befolkningsstatistikk
Kommuneinndeling 2020	Alle	Delområde- og grunnkretsinndeling 2021	Alle	Befolkningsstatistikk
Kommuneinndeling 2020	Alle	Delområde- og grunnkretsinndeling 2020	Alle	Befolkningsstatistikk
Arbeidsgiveravgiftssoner 2014-07	Alle	Kommuneinndeling 2020	Alle	Næringslivets strukturer
Barnevernsregionar 2004	Alle	Kommuneinndeling 2020	Alle	Helse-, omsorg- og sosialstatistikk
Familievernregioner 2020	Alle	Kommuneinndeling 2020	Alle	Helse-, omsorg- og sosialstatistikk
Fylkesinndeling 2020	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk
Fylkeskommune 2020	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk
Helseregioner 2007	Alle	Kommuneinndeling 2020	Alle	Helse-, omsorg- og sosialstatistikk
KOSTRA - kommunegruppering 2020	Alle	Kommuneinndeling 2020	Alle	Offentlige finanser
Kommunale organisasjonsnumre 2020	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk
Kirkelig fellesråd 2021	Nivå 1	Kommuneinndeling 2020	Nivå 1	Offentlige finanser
Kirkelig fellesråd 2020	Nivå 1	Kommuneinndeling 2020	Alle	Offentlige finanser
Landsdelsinndeling 2020	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk
Politidistrikt 2016	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk
Reiselivsregioner 2020	Alle	Kommuneinndeling 2020	Alle	Næringslivets konjunkturer
Sentralitet 2020	Alle	Kommuneinndeling 2020	Alle	Befolkningsstatistikk

Trinn 4: Denne tabellen koder om fra sentralitetskode til kommunekode. Om du vil kode om motsatt vei, altså fra kommunekode til sentralitetskode, klikker du på “Inverter tabell”



Gjeldende versjon: (Gyldig fra og med januar 2020)

Kommuneinndeling 2020

Koder | Om versjonen | Endringer | Alle versjoner | Korrespondanser | Varianter

<< Tilbake til alle korrespondanser

Sentralitet 2020 - Kommuneinndeling 2020

Ansvarlig: Høydahl, Even, seksjon Befolkningsstatistikk

Publisert på: Bokmål

Filtrer etter koder eller navn | Filtrer | Nullstill

Inverter tabell | Last ned CSV

Sentralitet 2020	Kommuneinndeling 2020
01 - Sentralitet: 1 (925-1000) - høy	**0301** - Oslo **3020** - Nordre Follo **3024** - Bærum **3027** - Rælingen **3029** - Lørenskog **3030** - Lillestrøm
02 - Sentralitet: 2 (870-924)	**1103** - Stavanger **1108** - Sandnes **3002** - Moss **3003** - Sarpsborg **3004** - Fredrikstad **3005** - Drammen **3019** - Vestby **3021** - Ås **3022** - Frogn **3025** - Asker **3031** - Nittedal **3032** - Gjerdrum **3033** - Ullensaker **3049** - Lier **3403** - Hamar **3801** - Horten **3803** - Tønsberg **4601** - Bergen **5001** - Trondheim
03 - Sentralitet: 3 (775-869)	**1106** - Haugesund **1120** - Klepp **1121** - Time **1122** - Gjesdal **1124** - Sola **1127** - Randaberg **1507** - Ålesund

Trinn 5: Du har nå en tabell som koder om fra kommunekoder til sentralitetskode. Denne kan lastes ned ved å klikke på "Last ned CSV". Velg passende lagringsområde på din PC, og hent frem filen når du er i dialogboksen for opplasting av filer i skriptvinduet.

Gjeldende versjon: (Gyldig fra og med januar 2020)

Kommuneinndeling 2020

Koder | Om versjonen | Endringer | Alle versjoner | Korrespondanser | Varianter

<< Tilbake til alle korrespondanser

Sentralitet 2020 - Kommuneinndeling 2020

Ansvarlig: Høydahl, Even, seksjon Befolkningsstatistikk
Publisert på: Bokmål

Filtrer etter koder eller navn | Filtrer | Nullstill

Inverter tabell | Last ned CSV

Kommuneinndeling 2020	Sentralitet 2020
0301 - Oslo	**01** - Sentralitet: 1 (925-1000) - høy
1101 - Eigersund	**04** - Sentralitet: 4 (670-774)
1103 - Stavanger	**02** - Sentralitet: 2 (870-924)
1106 - Haugesund	**03** - Sentralitet: 3 (775-869)
1108 - Sandnes	**02** - Sentralitet: 2 (870-924)
1111 - Sokndal	**05** - Sentralitet: 5 (565-669)
1112 - Lund	**05** - Sentralitet: 5 (565-669)
1114 - Bjerkreim	**04** - Sentralitet: 4 (670-774)
1119 - Hå	**04** - Sentralitet: 4 (670-774)
1120 - Klepp	**03** - Sentralitet: 3 (775-869)
1121 - Time	**03** - Sentralitet: 3 (775-869)
1122 - Gjesdal	**03** - Sentralitet: 3 (775-869)
1124 - Sola	**03** - Sentralitet: 3 (775-869)
1127 - Randaberg	**03** - Sentralitet: 3 (775-869)
1130 - Strand	**04** - Sentralitet: 4 (670-774)
1133 - Hjelmeland	**06** - Sentralitet: 6 (0-564) - lav

3.3 Bruk av funksjoner

I tillegg til de vanlige matematiske operatorene

```
=, +,  -,  /,  *,  ( , )
```

har en gjennom microdata.no tilgang på et stort antall funksjoner som vil være til hjelp når en skal generere variabler. Et konkret eksempel er når en skal angi bosted på fylkesnivå. Siden bostedsinformasjon i utgangspunktet angis som kommunenummer (= tosifret fylkesnummer + tosifret tilleggsnummer som angir kommune) gjennom en firesifret alfanumerisk variabel, må en benytte funksjonen `substr()` for å hente ut de to første sifrene som angir fylke:

```
generate fylke = substr(bosted,1,2)
```

Sifrene 1 og 2 inni funksjonen angir hhv. startposisjon for verdien som skal leses inn og antall posisjoner som skal leses. Kommunen Bergen har verdien ‘1201’. Å trekke ut de 2 første sifrene vil generere fylkesverdien for Hordaland som er ‘12’.

Et annet typisk bruksområde for funksjonen `substr()` er når en skal hente ut utdanningsnivå på et grovere aggregeringsnivå enn den totale 6-sifrede kodingen . Det er vanlig å bruke en inndeling på 1-sifret eller 2-sifret nivå. Da er denne funksjonen et nyttig hjelpemiddel.

Andre sentrale funksjoner er `round()` og `int()` evt. `floor()`. Disse er nyttige om en skal gjøre om fra desimaltall til heltall eller trekke ut delverdier fra en større tallverdi. `round()` avrunder desimaltall på vanlig måte, mens `int()` og `floor()` avrunder nedover. Om en f.eks. ønsker å hente ut fødselsåret fra den numeriske variabelen `faarmnd` (år og måned på formen YYYYMM), kan dette gjøres på følgende måte:

```
generate faar = int( faarmnd / 100)
```

I denne operasjonen dividerer vi på 100 og beholder heltallet (samme som å avrunde nedover). Dette vil i praksis trekke ut de fire første sifrene fra en numerisk 6-sifret verdi. For verdien 201006 vil en altså hente ut verdien 2010. Dette kan brukes til å generere alder ut i fra fødselsdato:

```
generate alder = 2013 - int( faarmnd / 100)
```

Resultatet vil gi alder for samtlige individer i datasett målt per 2013. Om fødselsdato er oppgitt med 8 siffer (*YYYYMMDD*), må en justere formelen for å få samme resultat:

```
generate alder = 2013 - int( faarmnd / 10000)
```

I vedlegg B presenteres en fullstendig liste over tilgjengelige funksjoner. Merk at disse stiller krav til hvilke typer variabler de brukes på. F.eks. kan funksjonen `substr()` kun brukes på alfanumeriske variabler.

3.4 Generere aggregerte verdier over tid - collapse

I tillegg til å aggregere data til et høyere enhetsnivå[7], f.eks. fra personnivå til familienivå (eller kommunenivå), kan kommandoen `collapse` også benyttes om en ønsker å beregne verdier målt over et angitt tidsspenn. En foretar da i praksis en aggregering fra hendelsesnivå/forløpsnivå til personnivå. Eksempler kan være beregninger av varighet i en tilstand målt over et gitt tidsintervall, uthenting av tilstand/status i et gitt tidsintervall, uthenting av antall forekomster i gitte tilstander/statuser i et gitt tidsintervall, eller summering av verdier over et gitt tidsintervall.

Dette gjøres på hendelsesorganiserte datasett (se kapittel 2.3.2) gjennom følgende kommando:

```
collapse (<aggregeringstype>) <datasett>, by(<enhetsid>)
```

En angir altså hva slags type aggregering en vil foreta i parentesen bak `collapse,` og deretter navnet på et hendelsesorganisert datasett. Aggregeringstype kan være følgende:

- `max`	maksverdi
- `min`	minimumsverdi
- `mean`	gjennomsnittsverdi
- `median`	medianverdi
- `count`	antall verdier
- `sum`	sum av verdier
- `semean`	standardfeil av gjennomsnitt
- `sebinomial`	binominal standardfeil av gjennomsnitt
- `sd`	standardavvik
- `percent`	prosentandel gyldige verdier
- `iqr`	interkvartilbredde (avstand mellom 75. og 25. prosentil)

Opsjonen `by(`*`<enhetsid>`*`)` brukes til å angi enhetstypen en skal aggregere over. Dette vil som regel være individ, gitt ved enhetsid `PERSONID_1`.

[7] Se kapittel 2.8

Eksempel 1: Beregne antall ganger individene har skiftet sivilstand i løpet av 2000-2005

```
require no.ssb.fdb:12 as db
create-dataset sivstforløp
import-event db/SIVSTANDFDT_SIVSTAND 2000-01-01 to 2005-01-01 as
     sivstperiode
collapse (count) sivstperiode, by(PERSONID_1)
rename sivstperiode antsivstand
replace antsivstand = antsivstand - 1
tabulate antsivstand
```

Eksempel 2: Beregne antall ganger individene har skilt seg i løpet av 2000-2005

```
require no.ssb.fdb:12 as db
create-dataset sivstforløp
import-event db/SIVSTANDFDT_SIVSTAND 2000-01-01 to 2005-01-01 as
     sivstperiode
keep if sivstperiode == '4'
collapse (count) sivstperiode, by(PERSONID_1)
rename sivstperiode antgangerskilt
tabulate antgangerskilt
```

Merk at variabelen `sivstperiode` i utgangspunktet angir sivilstand (hver nye record representerer en endring i sivilstand). Gjennom trinnene i eksemplene blir imidlertid variabelen transformert fra å inneholde sivilstand på hendelsesnivå til etterpå å inneholde `count`-verdien målt over 2000-2005 på enhetsnivå (= individ). Etter at `collapse` er kjørt inneholder variabelen `sivstperiode` altså en verdi som angir hvor mange sivilstatuser hvert individ har hatt over den angitte perioden (eksempel 1) eller hvor mange ganger en har hatt sivilstatusen “skilt” (= antall ganger skilt) (eksempel 2).

NB! For å kunne jobbe videre med den aggregerte verdien generert gjennom `collapse`, må en koble datasettet sammen med de øvrige variabler som ligger i hoveddatasettet bygget opp gjennom `import`-prosedyren (se kapittel 2.3.1). Se kapittel 2.8 for hvordan dette gjøres.

3.5 Endre navn på variabler

Variabelnavn kan endres fritt. Det er hensiktsmessig at variabler har forståelige og intuitive navn. Dette gjøres enkelt gjennom følgende kommando:

```
rename <variabelnavn_gml> <variabelnavn_ny>
```

Siden variabler i microdata.no er sterkt knyttet til tid, kan det være en god regel å inkludere tidspunkt (årstall) i navnet. Eksempel:

```
rename sivstand sivstand00
```

3.6 Lage labler

Tabellkjøringer og annen statistisk output blir mer forståelige om en knytter tekst til de ulike kategoriske verdiene for en variabel. I microdata.no kan man definere et sett med “value labels” som kan knyttes til alle variabler med samme type verdikoding:

```
define-labels <labelsettnavn> <verdi1> <label1> <verdi2>
<label2> …. <verdin> <labeln>

assign-labels <variabel> <labelsettnavn>
```

Først angis altså labler for de ulike verdiene, og i neste trinn knyttes settet med labler til nærmere spesifiserte variabler.

Eksempel på en kategorisk (alfanumerisk) bostedsvariabel på fylkesnivå (variabelen `fylke`). Settet med labler (navngitt som “fylkerstring”) kan gjennom kommandoen `assign-labels` knyttes til så mange variabler en ønsker, gitt at de har samme type verdisett (en slipper altså å opprette samme settet med labler flere ganger):

```
define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03'
'Oslo' '04' 'Hedmark' '05' 'Oppland' '06' 'Buskerud' '07'
'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder'
'11' 'Rogaland' '12' 'Hordaland' '14' 'Sogn og Fjordane' '15'
'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring
```

Om en variabel har numeriske verdier, droppes fnutter rundt verdiene i define-labels-kommandoen. Lablene som lenkes mot verdiene trenger ikke ha fnutter rundt dersom de bare inneholder bokstaver. Men labler som inneholder spesialtegn som mellomrom, bindestrek, skråstrek, punktum etc, må ha fnutter rundt. Om en er usikker, kan en alltid bruke fnutter rundt labler uansett.

Det er også lov å bruke dobbelfnutter dersom det er ønskelig.

<u>NB! Kommategn må ikke brukes inni labler! Dette vil gi feilmelding ved kjøring (kommategnet er reservert til å brukes kun i forbindelse med opsjoner).</u>

3.7 Endre verdiformat fra alfanumerisk (tekst) til numerisk

Mange av variablene tilgjengelig i microdata.no har alfanumerisk verdiformat, dvs. at de betraktes som tekst. Gjennom kommandoen `destring` kan en konventere slike om til numeriske variabler. Som standard vil variabelen overskrives med det nye formatet:

```
destring <variabel/variabelliste> [, <opsjoner>]
```

Om enkelte verdier består av ikke-numeriske karakterer, f.eks. “,”, “.”, “-”, “kr”, vil konverteringen ikke gjennomføres med mindre en bruker opsjonene `force` eller `ignore()`.

Opsjonen `force` tvinger systemet til å konvertere til numerisk uansett, men verdier bestående av ikke-numeriske karakterer vil få manglende verdi: `sysmiss`

Gjennom opsjonen `ignore()` kan en definere hvilke karakterer/tegn som skal ignoreres under konverteringen. Dette kan være aktuelt dersom verdier inneholder bindestreker, kommategn, tusenskilletegn eller annet. Eksempelet under vil ignorere punktum, komma og bindestrek i verdier der dette finnes for variabelen *var1*:

```
destring var1, ignore('.,-')
```

Alfanumeriske verdier som benytter komma som desimalskilletegn (‘2,1’, ‘10000,00’ etc) kan konverteres direkte til numeriske verdier der en beholder desimalene. Den konverterte verdien vil da få punktum som desimaltegn. Dette gjøres gjennom opsjonen `dpcomma`.

For mer informasjon om bruk av `destring`, f.eks. oversikt over alle tilgjengelige opsjoner, bruk kommandoen `help destring`.

3.8 Eksempel

```
// Kobler til databank
require no.ssb.fdb:12 as db

// Henter variablene en trenger
create-dataset demografidata
import db/BEFOLKNING_KJOENN as kjonn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue
import db/BOSATTEFDT_BOSTED 2015-01-01 as bosted

// Beregner alder i 2015 ut i fra fødselsår
generate alder = 2015 - int( faarmnd / 100 )

// Lager en dummyvariabel som angir mann ut i fra kjønn
generate mann = 0
replace mann = 1 if kjonn == '1'

// Gruppere formue i 4 intervaller
generate formueint = 1
replace formueint = 2 if formue > 150000
replace formueint = 3 if formue > 250000
replace formueint = 4 if formue > 400000

// Angi formue i 1000 kr
generate formue1000 = formue / 1000

// Koder om fra kommune- til fylkesnivå
generate fylke = substr(bosted,1,2)

// Legger til verdilabler for å navngi fylker med navn (=> penere deskriptiv output)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring

tabulate fylke
```

4. Hvordan gjøre seg kjent med variabler

I microdata.no kan en bruke ulike teknikker for å utforske variabler og datasett. Den enkleste er bruk av tabeller (enveis- eller krysstabeller) eller oppsummeringsstatistikk (for metriske variabler). En kan også visualisere gjennom histogrammer, søylediagrammer, kakediagrammer og anonymiserte plotdiagrammer (hexbinplot) på en oversiktlig måte.

Analysesystemet microdata.no har følgende kommandoer tilgjengelig for produksjon av deskriptiv statistikk:

- `tabulate`
- `summarize`
- `boxplot`
- `hexbin`
- `piechart`
- `histogram`
- `barchart`
- `sankey`

Gjennom options kan en vise alternative fremstillinger av de samme fordelingene, og en kan utelate enheter fra tabellene/figurene gjennom if-betingelser.

4.1 Tabulate - frekvenstabeller

Kommandoen `tabulate` brukes til å lage frekvenstabeller, og er den vanligste statistikk-kommandoen når en skal gjøre seg kjent med variabelinnholdet, samt når en skal lage deskriptiv statistikk.

Kommandoen kan brukes på alle kategoriske variabler. Disse vil ofte være alfanumeriske, men det er fullt mulig å lage frekvenstabeller for numeriske variabler også, gitt at antallet verdier ikke blir for omfattende.

Standardfremvisningen for tabeller generert gjennom `tabulate` er frekvenstall (antall enheter), og disse kan være enveis, toveis og flerdimensjonale. Som standard vises rekke- og kolonneverdienes verdi-labler for variabler som har dette, og missingverdier utelates fra tabellgrunnlaget.

Gjennom bruk av opsjoner kan en kontrollere presentasjonen og overstyre standardfremvisningen:

- Vise prosentandeler i stedet for frekvenstall
- Vise kolonne- og rekkeverdier uten verdi-labler
- Vise tabeller der missingkategorier tas med i tabellgrunnlaget
- Lage såkalte volumtabeller som viser oppsummerende verdier (gjennomsnitt, sum m.m.) for valgfrie variabler innen hver celle
- Gjennomføre en chi-test (tester for avvik fra en helt tilfeldig bivariat fordeling) gjennom en `chi2`-opsjon

Som for de fleste kommandoer i microdata.no kan en i forbindelse med `tabulate` benytte filtre i form av IF-betingelser for å styre hvilke enheter som skal inngå i den aktuelle tabellen. En trenger altså ikke nødvendigvis trimme utvalget i datasettet i forkant av statistiske kjøringer.

Syntax:

```
tabulate <variabel/variabelliste> [, <opsjoner>]
```

For mer informasjon om denne kommandoen, bruk kommandoen `help tabulate`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

Tips:

Tabeller generert gjennom kommandoen `tabulate` kan eksporteres over i andre programmer som Excel, Word, Google Sheets m.m. Dette gjøres gjennom å klikke på et "copy"-ikon som dukker opp når en holder musepekeren over tabellen. Deretter bruker en tastekombinasjonen *<Ctrl>* + *<C>* og limer så inn i ønsket dokument. Dette kan også gjøres på annen type output, som f.eks. `regress`.

4.1.1 Enveis frekvenstabeller

Eksempel på frekvenstabell for variablene “bostedsfylke per 2000-01-01” og kjønn:

»tabulate fylke00

fylke00		
	Østfold	144599
	Akershus	292087
	Oslo	310657
	Hedmark	108743
	Oppland	109887
	Buskerud	143930
	Vestfold	124566
	Telemark	95748
	Aust-Agder	59546
	Vest-Agder	90565
	Rogaland	223820
	Hordaland	258569
	Sogn og Fjordane	66239
	Møre og Romsdal	146564
	Sør-Trøndelag	156169
	Nord-Trøndelag	74196
	Nordland	140258
	Troms	91962
	Finnmark	45540
	Uoppgitt	6
Sum		*2683675*

»tabulate kjonn

kjonn		
	Mann	1424545
	Kvinne	1274038
Sum		*2698574*

4.1.2 Flerdimensjonale frekvenstabeller

Flerdimensjonale frekvenstabeller er tabeller med fordelinger over 2 eller flere variabler. Disse inneholder såkalte celleverdier, rekke- og kolonnesummer, og totalverdi (nederst til høyre).

Eksempel på frekvenstabell med fordeling over kjønn og registerstatus:

```
»tabulate kjonn regstat
```

kjonn		regstat: bosatt	utvandret	død	uregistrert person	Sum
kjonn	Mann	1414045	3157	7	17	*1417207*
kjonn	Kvinne	1266295	2538	8	6	*1268840*
Sum		*2680342*	*5695*	*11*	*15*	*2686050*

Eksempel på tredimensjonal frekvenstabell med fordeling over kjønn, registerstatus og bostedsfylke (merk at hele tabellen ikke vises i illustrasjonen under):

```
»tabulate kjonn regstat fylke00
```

kjonn	fylke00	regstat: bosatt	utvandret	uregistrert person	død	Sum
Mann	Østfold	76494	19	--	--	*76517*
Mann	Akershus	151246	59	--	--	*151300*
Mann	Oslo	159035	209	11	--	*159249*
Mann	Hedmark	57504	7	--	--	*57517*
Mann	Oppland	58036	6	--	--	*58050*
Mann	Buskerud	75605	25	--	--	*75625*
Mann	Vestfold	65492	16	--	--	*65513*
Mann	Telemark	50891	6	--	--	*50890*
Mann	Aust-Agder	31863	14	--	--	*31874*
Mann	Vest-Agder	48504	15	--	--	*48514*
Mann	Rogaland	119190	25	--	--	*119216*
Mann	Hordaland	136460	42	--	--	*136512*
Mann	Sogn og Fjordane	35231	10	--	--	*35231*
Mann	Møre og Romsdal	78937	18	--	--	*78947*
Mann	Sør-Trøndelag	82371	16	--	--	*82394*
Mann	Nord-Trøndelag	39680	7	--	--	*39689*
Mann	Nordland	74471	19	--	--	*74488*
Mann	Troms	48745	20	--	--	*48769*
Mann	Finnmark	24157	11	--	--	*24173*
	Østfold	67909	16	--	--	*67927*
	Akershus	140466	35	--	--	*140494*

4.1.3 Frekvenstabeller og prosentuering

Kommandoen `tabulate` kan, i tillegg til å vise frekvenser (antall enheter) for kombinasjoner av verdier for de aktuelle variabler, brukes til å vise prosentandeler. Dette styres gjennom opsjoner:

- `rowpct` rekkeprosent (andel av rekketotalen)
- `colpct` kolonneprosent (andel av kolonnetotalen)
- `cellpct` celleprosent (andel av totalen)
- `freq` frekvensverdi (standardvisning, brukes kun i kombinasjon med prosentuering)

Flere opsjoner kan kombineres i samme kommando, og en kan f.eks. vise både frekvenser og rekkeprosenter i en og samme tabell (se eksempel nr 4 nedenfor).

Eksempler:

```
»tabulate sivstand00 sivstand05, rowpct
```

		sivstand05										
		Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	Sum
sivstand00	0 ukjent	64.71	29.41	9.8	9.8	--	--	--	--	--	--	100
	Ugift	91.72	7.73	0.17	0.29	0	0.02	0.06	0	0	0	100
	Gift	0	89.95	3.38	2.49	0	4.17	0	0	--	--	100
	Enke/enkemann	--	0.93	0.02	0.03	0	99.03	--	--	--	--	100
	Skilt	--	10.8	88.49	0.56	0	0.11	0.05	0	--	--	100
	Separert	0.01	16.36	52.87	28.52	0.01	2.18	0.04	0.01	--	--	100
	Registrert partner	--	0.53	--	--	--	--	76.66	7.96	13.53	1.92	100
	Separert partner	--	--	--	--	--	--	18.18	20.45	56.82	--	100
	Skilt partner	--	5.68	--	--	--	--	13.64	7.95	64.77	--	100
	Gjenlevende partner	--	--	--	--	--	--	--	--	--	123.08	100
	Sum	45.61	38.91	7.6	1.52	0	6.29	0.06	0.01	0.01	0	--

```
»tabulate sivstand00 sivstand05, colpct
```

		sivstand05										
		Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	Sum
sivstand00	0 ukjent	0	0	0	0.01	--	--	--	--	--	--	0
	Ugift	100	9.88	1.13	9.59	72.73	0.16	46.35	32.79	14.55	10	49.73
	Gift	0	87.7	16.88	62.31	31.82	25.17	1.22	2.05	--	--	37.93
	Enke/enkemann	--	0.11	0.01	0.08	22.73	74.09	--	--	--	--	4.7
	Skilt	--	1.73	72.46	2.29	22.73	0.11	4.73	4.1	--	--	6.22
	Separert	0	0.58	9.52	25.71	22.73	0.47	0.91	2.46	--	--	1.37
	Registrert partner	--	0	--	--	--	--	45.6	49.18	53.97	58	0.04
	Separert partner	--	--	--	--	--	--	0.63	7.38	13.23	--	0
	Skilt partner	--	0	--	--	--	--	0.47	2.87	15.08	--	0
	Gjenlevende partner	--	--	--	--	--	--	--	--	--	32	0
	Sum	100	100	100	100	100	100	100	100	100	100	--

```
»tabulate sivstand00 sivstand05, cellpct
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	*Sum*
0 ukjent	0	0	0	0	--	--	--	--	--	--	*0*
Ugift	45.61	3.84	0.09	0.15	0	0.01	0.03	0	0	0	*49.73*
Gift	0	34.12	1.28	0.95	0	1.58	0	0	--	--	*37.93*
Enke/enkemann	--	0.04	0	0	0	4.66	--	--	--	--	*4.7*
Skilt	--	0.67	5.51	0.03	0	0.01	0	0	--	--	*6.22*
Separert	0	0.22	0.72	0.39	0	0.03	0	0	--	--	*1.37*
Registrert partner	--	0	--	--	--	--	0.03	0	0	0	*0.04*
Separert partner	--	--	--	--	--	--	0	0	0	--	*0*
Skilt partner	--	0	--	--	--	--	0	0	0	--	*0*
Gjenlevende partner	--	--	--	--	--	--	--	--	--	0	*0*
Sum	*45.61*	*38.91*	*7.6*	*1.52*	*0*	*6.29*	*0.06*	*0.01*	*0.01*	*0*	--

```
»tabulate sivstand00 sivstand05, rowpct freq
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	*Sum*
0 ukjent	33 64.71	15 29.41	5 9.8	5 9.8	--	--	--	--	--	--	*51* *100*
Ugift	1915459 91.72	161418 7.73	3612 0.17	6112 0.29	16 0	432 0.02	1175 0.06	80 0	55 0	5 0	*2088366* *100*
Gift	41 0	1432952 89.95	53856 3.38	39697 2.49	7 0	66463 4.17	31 0	5 0	--	--	*1593035* *100*
Enke/enkemann	--	1830 0.93	34 0.02	54 0.03	5 0	195650 99.03	--	--	--	--	*197576* *100*
Skilt	--	28222 10.8	231185 88.49	1457 0.56	5 0	280 0.11	120 0.05	10 0	--	--	*261270* *100*
Separert	5 0.01	9396 16.36	30361 52.87	16379 28.52	5 0.01	1250 2.18	23 0.04	6 0.01	--	--	*57425* *100*
Registrert partner	--	8 0.53	--	--	--	--	1156 76.66	120 7.96	204 13.53	29 1.92	*1508* *100*
Separert partner	--	--	--	--	--	--	16 18.18	18 20.45	50 56.82	--	*88* *100*
Skilt partner	--	5 5.68	--	--	--	--	12 13.64	7 7.95	57 64.77	--	*88* *100*
Gjenlevende partner	--	--	--	--	--	--	--	--	--	16 123.08	*13* *100*
Sum	*1915545* *45.61*	*1633855* *38.91*	*319053* *7.6*	*63707* *1.52*	*22* *0*	*264057* *6.29*	*2535* *0.06*	*244* *0.01*	*378* *0.01*	*50* *0*	*4199434* --

4.1.4 Frekvenstabeller og kategori-labler

Om en kun ønsker å vise kolonne- og rekkeverdiene, og ikke verdi-lablene, kan opsjonen `nolabels` benyttes.

Eksempel:

```
»tabulate kjonn regstat fylke00, nolabels
```

kjonn	fylke00	regstat 1	3	9	5	Sum
1	01	76494	19	--	--	76517
	02	151246	59	--	--	151300
	03	159035	209	11	--	159249
	04	57504	7	--	--	57517
	05	58036	6	--	--	58050
	06	75605	25	--	--	75625
	07	65492	16	--	--	65513
	08	50891	6	--	--	50890
	09	31863	14	--	--	31874
	10	48504	15	--	--	48514
	11	119190	25	--	--	119216
	12	136460	42	--	--	136512
	14	35231	10	--	--	35231
	15	78937	18	--	--	78947
	16	82371	16	--	--	82394
	17	39680	7	--	--	39689
	18	74471	19	--	--	74488
	19	48745	20	--	--	48769
	20	24157	11	--	--	24173
	01	67909	16	--	--	67927
	02	140466	35	--	--	140494

4.1.5 Frekvenstabeller og missingverdier

Som standard brukes ikke missingverdier i tabellgrunnlaget når kommandoen `tabulate` kjøres. Disse holdes altså utenfor tallberegningene med mindre en benytter opsjonen `missing`.

Eksempel:

fylke00	bosatt	utvandret	død	SYSMISS	uregistrert person	Sum
Østfold	144411	34	5	162	--	144597
Akershus	291708	90	--	294	--	292087
Oslo	309693	347	--	601	8	310657
Hedmark	108642	17	--	78	--	108750
Oppland	109810	5	--	71	5	109883
Buskerud	143772	35	--	132	--	143931
Vestfold	124417	33	5	125	7	124573
Telemark	95667	5	--	65	--	95748
Aust-Agder	59486	16	--	49	--	59547
Vest-Agder	90478	14	--	84	--	90569
Rogaland	223568	35	--	215	--	223829
Hordaland	258272	63	--	227	--	258569
Sogn og Fjordane	66187	12	--	39	5	66234
Møre og Romsdal	146447	26	--	89	7	146566
Sør-Trøndelag	156024	32	--	114	--	156177
Nord-Trøndelag	74134	9	--	44	--	74191
Nordland	140114	23	--	114	--	140262
Troms	91835	25	--	99	--	91962
Finnmark	45483	14	--	39	--	45549
Uoppgitt	--	5	--	6	--	6
SYSMISS	204	4851	--	10451	--	15496
Sum	2680340	5693	5	13109	15	2699164

4.1.6 Frekvenstabeller og filtrering

Om en ønsker å lage tabeller for delpopulasjoner, kan en filtrere gjennom bruk av IF-betingelser. En trenger altså ikke justere på datasettet i forkant.

Eksempler:

```
»tabulate fylke00 regstat if regstat == '1'
```

		regstat	
		bosatt	Sum
fylke00	Østfold	144408	144408
	Akershus	291704	291704
	Oslo	309690	309690
	Hedmark	108645	108645
	Oppland	109805	109805
	Buskerud	143766	143766
	Vestfold	124413	124413
	Telemark	95670	95670
	Aust-Agder	59486	59486
	Vest-Agder	90473	90473
	Rogaland	223572	223572
	Hordaland	258279	258279
	Sogn og Fjordane	66183	66183
	Møre og Romsdal	146440	146440
	Sør-Trøndelag	156028	156028
	Nord-Trøndelag	74128	74128
	Nordland	140113	140113
	Troms	91842	91842
	Finnmark	45483	45483
Sum		2680141	2680141

```
»tabulate fylke00 regstat if alder > 30
```

fylke00	bosatt	utvandret	død	uregistrert person	Sum
Østfold	100971	11	5	--	100993
Akershus	209788	67	--	--	209858
Oslo	211918	218	--	13	212144
Hedmark	78091	6	--	--	78100
Oppland	78183	8	--	--	78188
Buskerud	101441	16	--	--	101464
Vestfold	87173	16	--	5	87185
Telemark	66740	--	--	--	66741
Aust-Agder	40247	7	--	--	40258
Vest-Agder	60803	12	--	--	60815
Rogaland	149250	32	--	--	149279
Hordaland	177392	41	--	--	177436
Sogn og Fjordane	45405	--	--	--	45404
Møre og Romsdal	100535	18	--	7	100551
Sør-Trøndelag	109421	26	--	--	109448
Nord-Trøndelag	52323	5	--	--	52327
Nordland	97681	18	--	--	97691
Troms	63562	18	--	--	63582
Finnmark	31089	11	--	--	31095
Sum	*1862018*	*550*	*9*	*12*	*1862583*

4.1.7 Volumtabeller

Kommandoen `tabulate` kan i tillegg til å vise frekvenser og prosentandeler brukes til å lage såkalte volumtabeller. I hver celle vil det da i stedet vises oppsummerende statistikk for en valgfri variabel. En kan fritt velge hvilken variabel som skal oppsummeres gjennom følgende opsjon:

```
summarize(<variabel>)
```

Som standard vises gjennomsnittsverdi som statistikk, men en kan overstyre dette ved å benytte følgende tilleggsopsjoner i `tabulate`-kommandoen:

- `mean` Gjennomsnitt (standard)
- `sum` Sum
- `std` Standardavvik
- `p25` 25%-kvartil
- `p50` 50%-kvartil (=medianverdi)
- `p75` 75%-kvartil
- `gini` Gini-koeffisientverdi
- `iqr` Interkvartilavstand (avstand mellom 75. og 25. kvartil)

Eksempel der en viser standardverdi (gjennomsnitt) for inntekt fordelt på kjønn:

```
tabulate kjønn, summarize(inntekt)
```

Eksempel der en viser medianverdi i stedet for gjennomsnitt:

```
tabulate kjønn, summarize(inntekt) p50
```

Eksempel der en viser gjennomsnittsinntekt fordelt på kjønn og sivilstand:

```
tabulate sivilstand kjønn, summarize(inntekt)
```

Eksempel på volumtabell der en viser gjennomsnittsformue fordelt på bostedsfylke og kjønn:

```
»tabulate fylke00 kjonn, summarize(formue)
```

fylke00	kjonn: Mann	kjonn: Kvinne	Sum
Østfold	420698.87	192280.33	*313535.11*
Akershus	482632.84	258851.53	*374659.58*
Oslo	407740.27	253787.41	*332438.7*
Hedmark	448040.46	207501.65	*335125.52*
Oppland	461985.03	203399.93	*340732.13*
Buskerud	451406.11	221951.55	*342941.33*
Vestfold	440535.63	210095.86	*331709.72*
Telemark	390275.19	181083.63	*292646.24*
Aust-Agder	396850.1	183568.96	*298338.32*
Vest-Agder	432110.14	192591.27	*321157.44*
Rogaland	423818.48	185711.51	*312884.07*
Hordaland	383759.34	189964.32	*292278.38*
Sogn og Fjordane	486762.24	203607.58	*354812.34*
Møre og Romsdal	435580.27	189451.31	*322286.72*
Sør-Trøndelag	370140.51	180124.65	*280522.87*
Nord-Trøndelag	399271.57	162695.47	*289981.45*
Nordland	350952.27	162837.57	*262965.02*
Troms	335615.35	170122.5	*257822.14*
Finnmark	295861.55	165703.72	*234849.97*
Sum	*416819.53*	*205087.38*	*316849.45*

4.2 Summarize og boxplot - statistikk for metriske variabler

Kommandoene `summarize` og `boxplot` brukes til å vise oppsummerende statistikk for metriske/kontinuerlige variabler. I likhet med andre statistikker i microdata.no, kan en lage statistikk også for delpopulasjoner via IF-betingelser (trenger ikke justere på datasettet i forkant).

Nedenfor vises eksempler for variablene inntekt og formue målt i hhv. 2019 og 2018, der populasjonen er alle bosatte i alderen 16-66 år.

Kommandoen `summarize` viser nøkkelstatistikk for de spesifiserte numeriske variablene:

- Gjennomsnitt
- Standardavvik
- Antall enheter med gyldig verdi
- Første prosentilverdi (øvre grenseverdi)
- Indre kvartilverdier (50% = medianverdi)
- Siste prosentilverdi (nedre grenseverdi)

Det er også mulig å vise ginikoeffisient-verdier samt interkvartilverdier (avstanden mellom 75. og 25. prosentil) ved å bruke hhv. opsjonene `gini` og `iqr`.

Kommandoen `boxplot` viser en grafisk fremstilling gjennom et standard boxplot med boks for de to midterste kvartilene, gjennomsnitt (samt minimums- og maksimumsverdi.

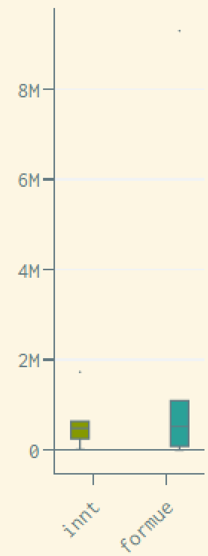
```
demografidata» summarize innt formue
```

Variabel	Gj.snitt	Std.avvik	Antall	1%	25%	50%	75%	99%
innt	485031.3743	333780.7368	2958923	1253	241256	473966	644347.75	1729989
formue	935371.5972	1434013.7402	3389824	64	81067	528833	1094222	9289672

```
demografidata» summarize formue if alder > 50
```

Variabel	Gj.snitt	Std.avvik	Antall	1%	25%	50%	75%	99%
formue	1533890.3026	2227840.8478	1023086	264	401163.5	878830	1731050.5	15032181

```
demografidata» boxplot innt formue
```

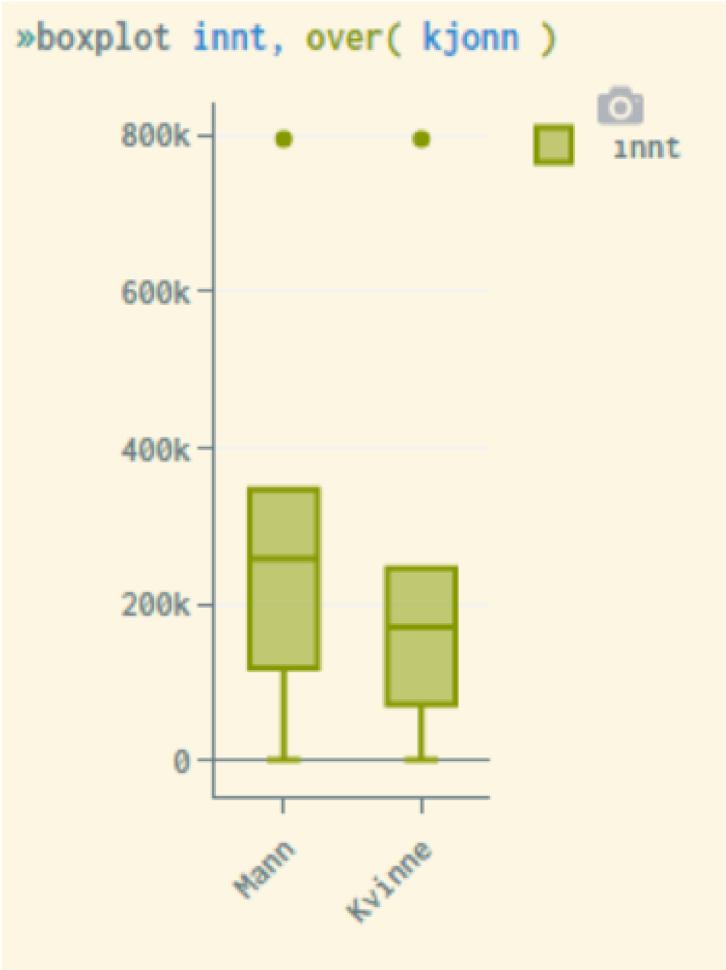


Om en holder musepekeren over de ulike områdene i boxplot-figuren, vil en kunne se hvilke verdier de ulike punktene representerer.

Kommandoen `boxplot` gir mulighet til å vise separate tall for gitte kategorier representert ved en annen kategorisk variabel:

`boxplot <variabel1>, over(variabel2)`

Eksempel på boxplot for inntekt per 2000-01-01 fordelt på kjønn:



For mer informasjon om disse kommandoene, bruk kommandoene `help summarize` eller `help boxplot`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres. F.eks. kan opsjonen `gini` brukes til å vise gini-koeffisientverdier i tillegg til standard `summarize`-resultat.

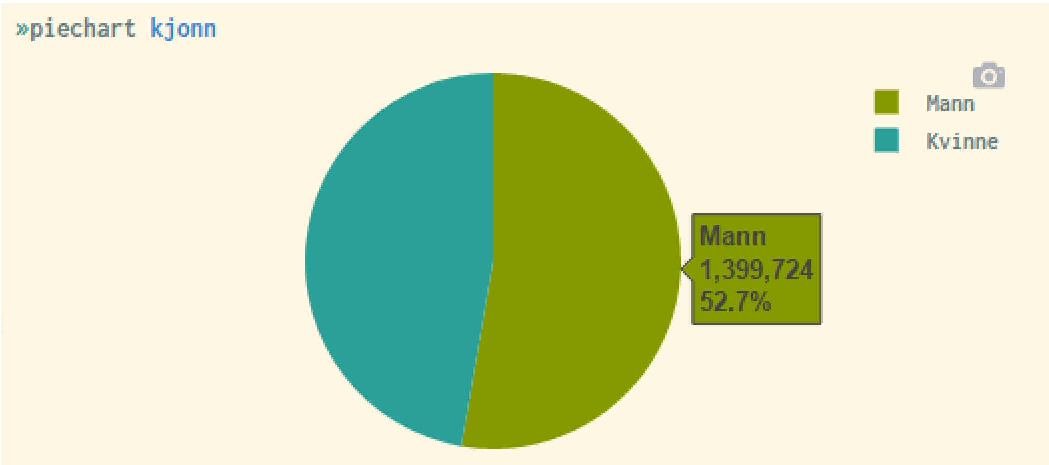
4.3 Piechart - kakediagram

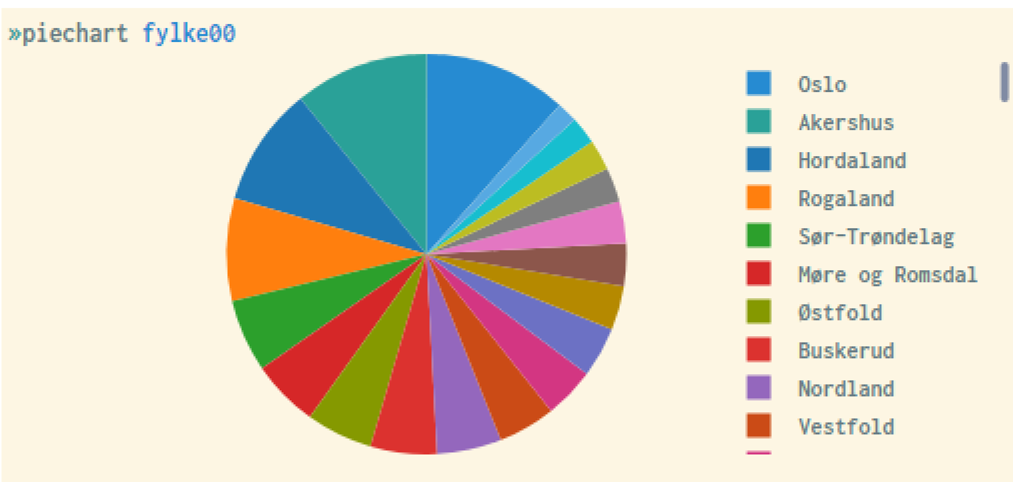
Gjennom følgende kommando kan en lage kakediagram for kategoriske variabler:

```
piechart <variabel>
```

For mer informasjon om denne kommandoen, bruk kommandoen `help piechart`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

Om en holder musepekeren over de ulike feltene i figuren, vil en kunne se tilhørende verdier.





4.4 Histogram - grafisk fremstilling av frekvensfordelinger

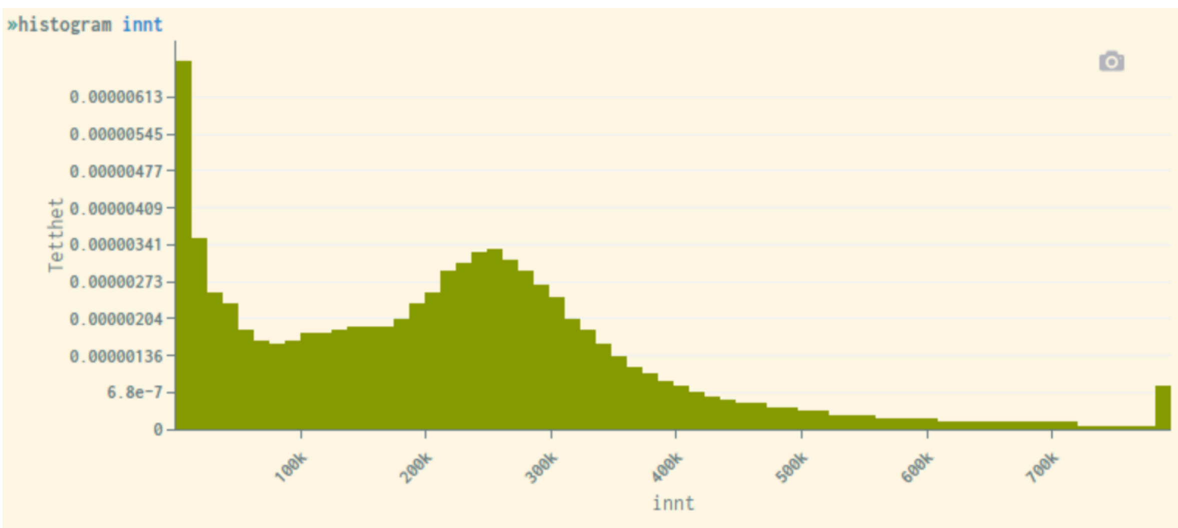
Histogrammer er grafiske fremstillinger av univariate fordelinger for kontinuerlige variabler (f.eks. inntekt). Hver søyle representerer frekvensverdien for et gitt forhåndsbestemt intervall for den aktuell variabelen. Gjennom opsjonene `bin()` og `width()` kan en overstyre dette og selv bestemme hhv. antall søyler og søyleintervallenes bredde. Nedenfor illustreres dette gjennom eksempler.

Standardfremvisningen viser tetthet som frekvensverdi. Også dette kan overstyres gjennom opsjoner, slik at måleenheten på y-aksen i stedet viser faktisk frekvens (antall), andel eller prosentandel. Følgende opsjoner kan benyttes til dette: `freq, fraction, percent`

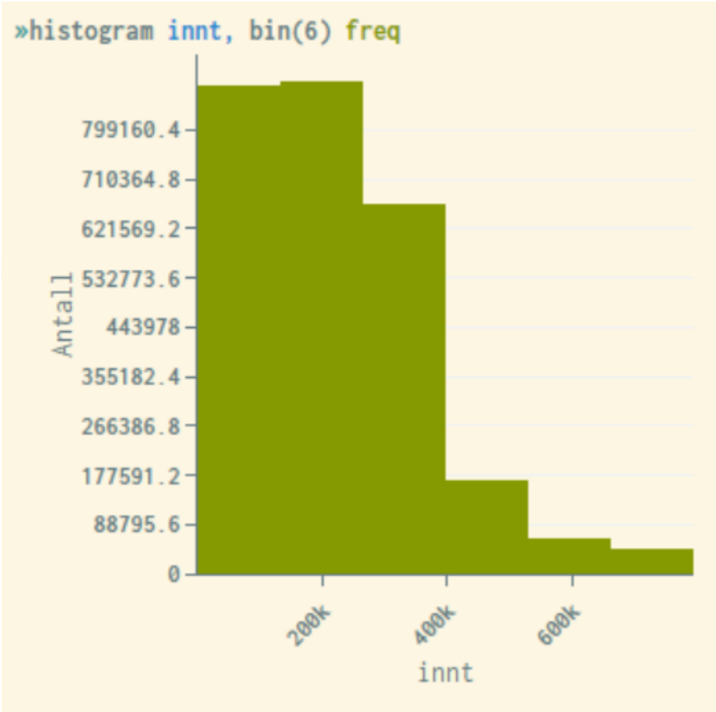
Personer med svært høye eller svært lave inntekter kan lett identifiseres om verdi-intervallet blir for smalt, noe som er problematisk med tanke på personvernet. Derfor foretar systemet en topp-/bunnkoding der de 1% høyeste og 1% laveste verdiene erstattes av grenseverdien til hhv. den siste og første prosentilen. Derfor vil alltid første og siste søyle være mye høyere enn nabosøylene, som illustrert i eksemplene nedenfor. Denne topp-/bunnkodingen omtales i detalj i Vedlegg C.

Om en holder musepekeren over de ulike søylene i figuren, vil en få opp intervallet samt frekvensverdi for den aktuelle søylen.

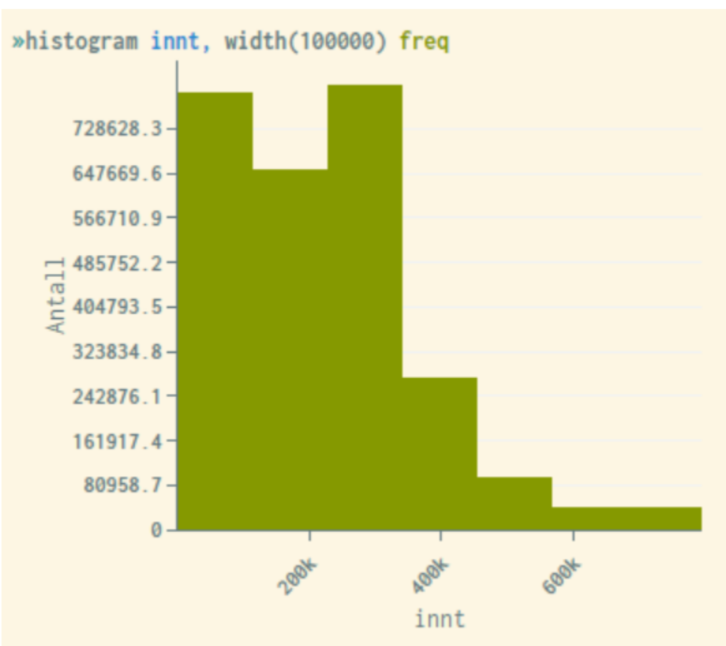
Eksempel:



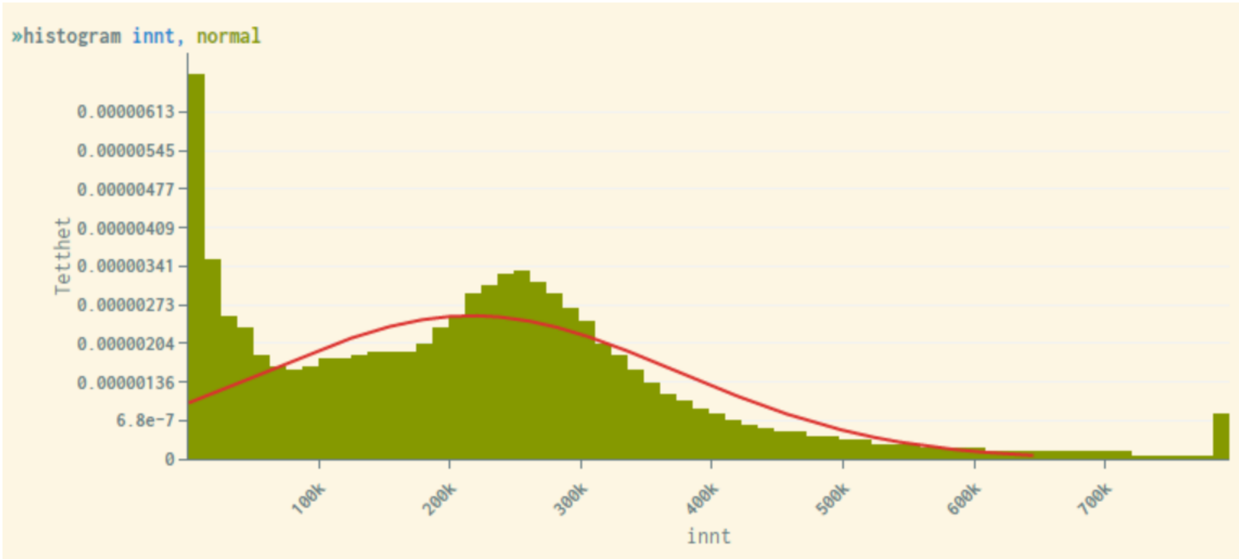
Histogram over inntekt fordelt på 6 søyler og frekvenstall på y-aksen (hver søyle har samme intervallbredde for inntekt):



Histogram over inntekt der hver søyle får intervallbredde for inntekt lik 100’000:

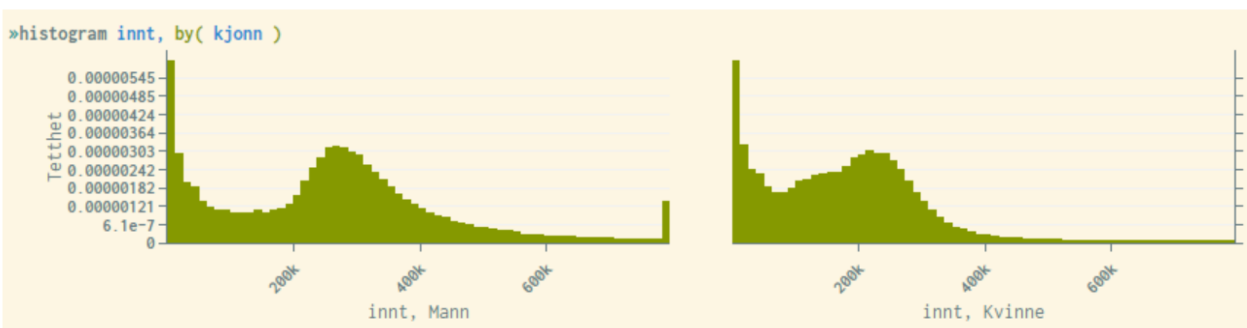


Gjennom opsjonen `normal`, kan en legge en normalfordelt kurve over søylene i figuren. Dette er til hjelp for å se på grad av avvik fra en normalfordeling:



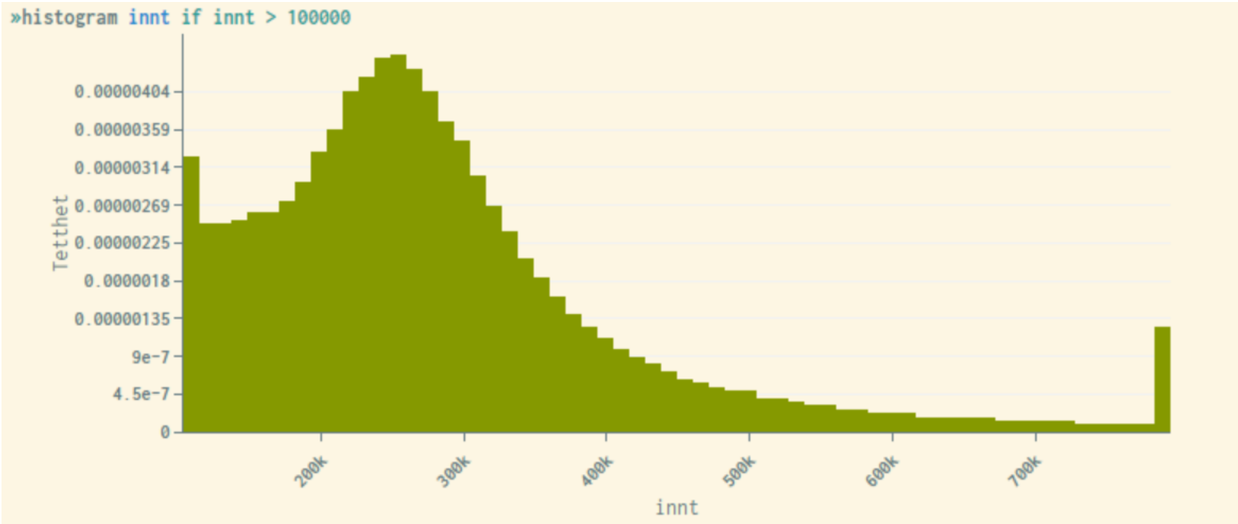
Histogrammer kan vises over fordelinger for en annen variabel som må være kategorisk, f.eks. Kjønn. Dette gjøres gjennom opsjonen `by( <variabel> )`.

Eksempel:



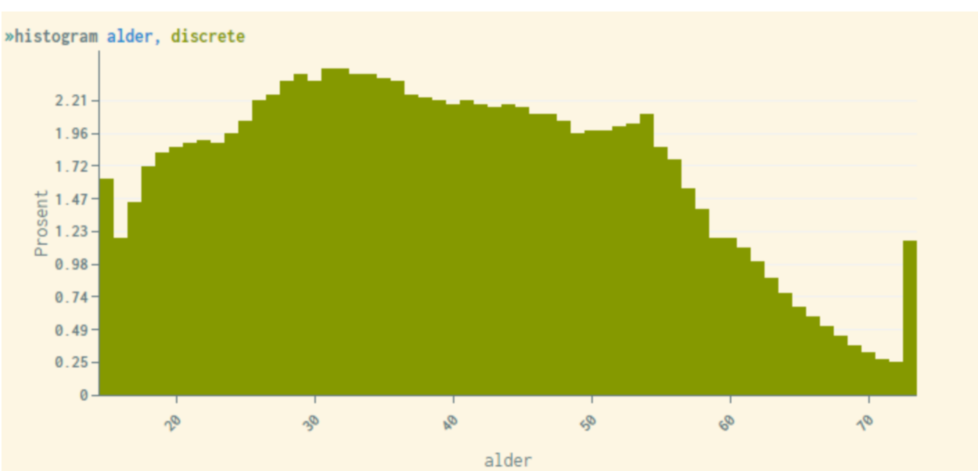
Som for andre statistiske fremstillinger i microdata.no, kan en gjøre en filtrering gjennom IF-betingelser, der en kun viser histogram for en delpopulasjon.

Eksempel der en viser histogram kun for personene med inntekt over 100 000:



Som nevnt vil histogram som standard dele inn i et forhåndsbestemt antall søyler/intervaller. Gjennom opsjonen `discrete` kan man overstyre dette og vise en søyle for hver enkelt verdi. Dette er ikke hensiktsmessig for metriske variabler av økonomisk art (blir svært mange søyler), men for numeriske variabler med et begrenset antall verdier anbefales det å bruke denne typen fremstilling. Eksempler på slike variabler kan være alder, prosentandeler, eller beløp som er ferdig avrundet (til nærmeste 10 000 eller 100 000).

Eksempel på bruk av opsjonen `discrete` for variabelen "alder" (en ser også her at systemet sørger for at første og siste søyle er mye høyere enn nabosøyler pga. topp-/bunn-kodingen, siden personer med svært høy/lav alder er relativt lette å identifisere):



Histogrammer som kombinerer `bin()` og `discrete` vil returnere et tomt diagram evt. en feilmelding siden disse to opsjonene ikke er kompatible sammen.

For mer informasjon om denne kommandoen, bruk kommandoen `help histogram`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

4.5 Barchart - søylediagram

Kommandoen `barchart` brukes til å lage vanlige søyledigrammer. En angir aktuell variabel/variabler samt statistikken som søylene skal måle. Gjennom opsjonen `over()`, kan en fordele søylene over en eller flere andre variabler som må være kategoriske (f.eks. kjønn).

Som for andre grafiske fremstillinger i microdata.no, kan en gjennom å holde musepekeren over de ulike felt i figuren få opp tilhørende verdier.

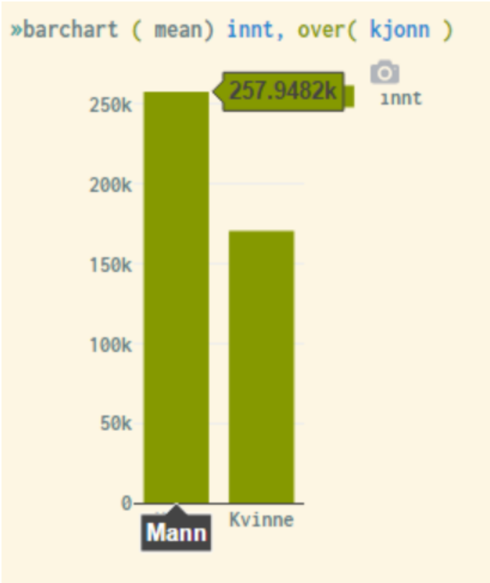
Syntax:

```
barchart(<statistikkmål>) <variabelliste> [, over(<variabelliste>)]
```

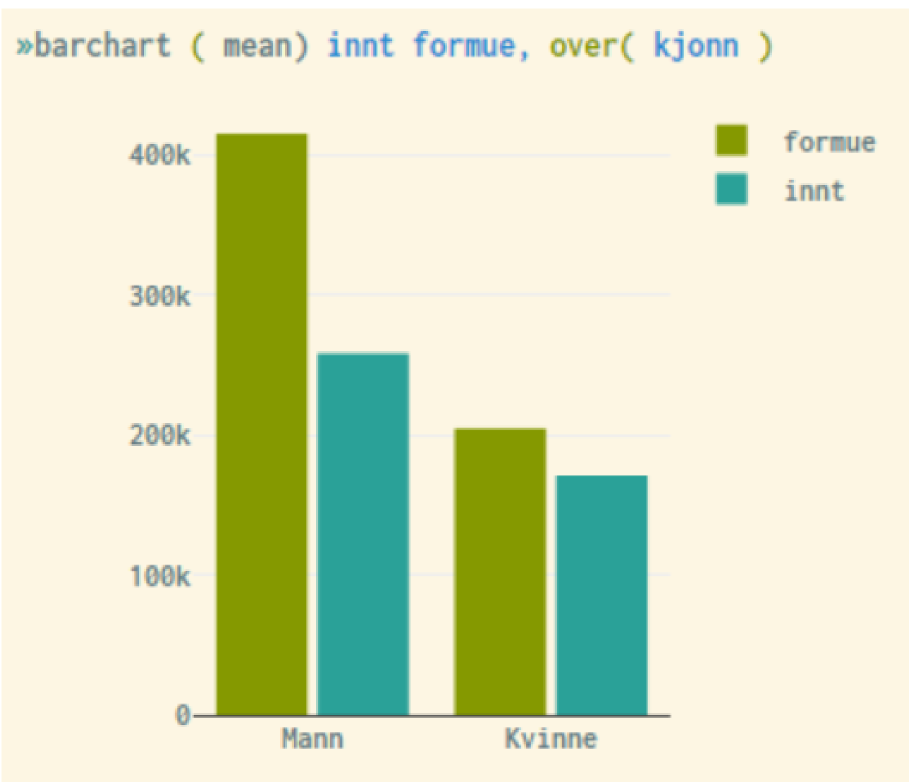
For å lage stablede søylediagrammer, kan en bruke opsjonen `stack`.

For mer informasjon om denne kommandoen, bruk kommandoen `help barchart`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

Eksempel på søylediagram over gjennomsnittsinntekt fordelt på kjønn:



Eksempel på søylediagram over gjennomsnittsinntekt og gjennomsnittsformue fordelt på kjønn:



4.6 Hexbin - anonymiserende plotdiagram

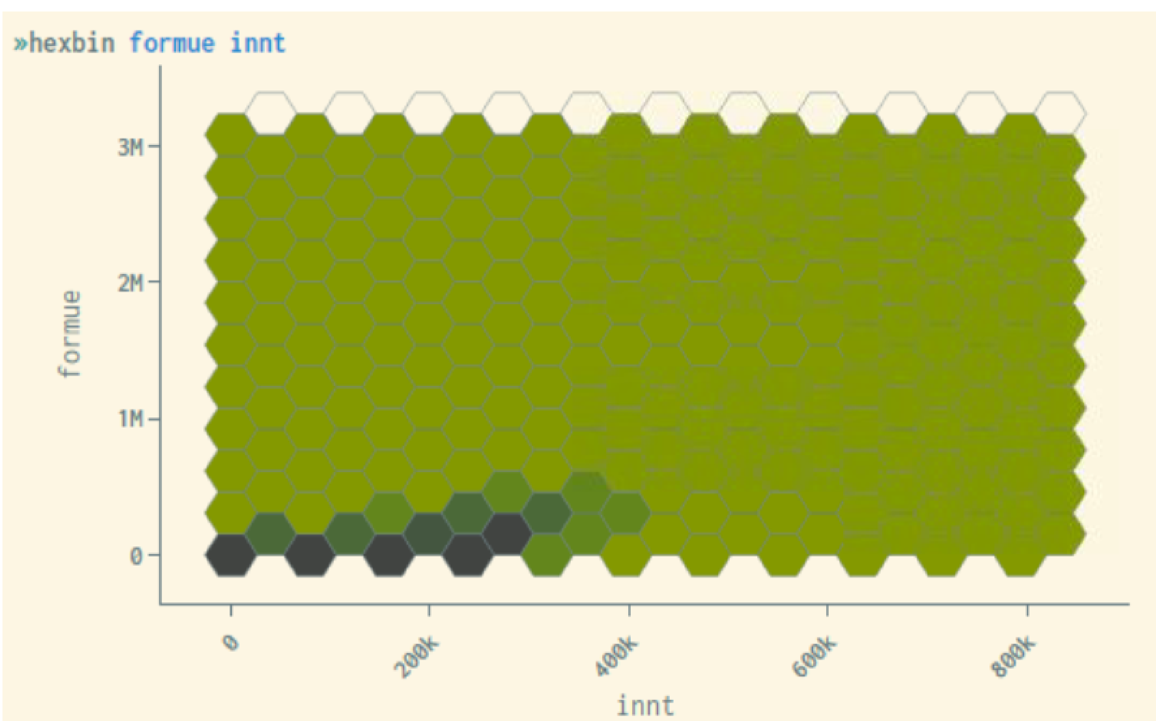
Hexbin-diagram er i praksis et anonymisert plotdigram der det to-dimensjonale arealet er delt inn i et gitt antall hexagonfelt. Fargen på disse hexagonene representerer tettheten av observasjoner/punkter i det aktuelle arealet. Jo mørkere farge, dess flere punkter befinner seg der. En får da en grafisk oversikt over fordelingen av enheter (individer) mellom to kontinuerlige variabler. Hexbindiagrammer egner seg ikke for kategoriske variabler.

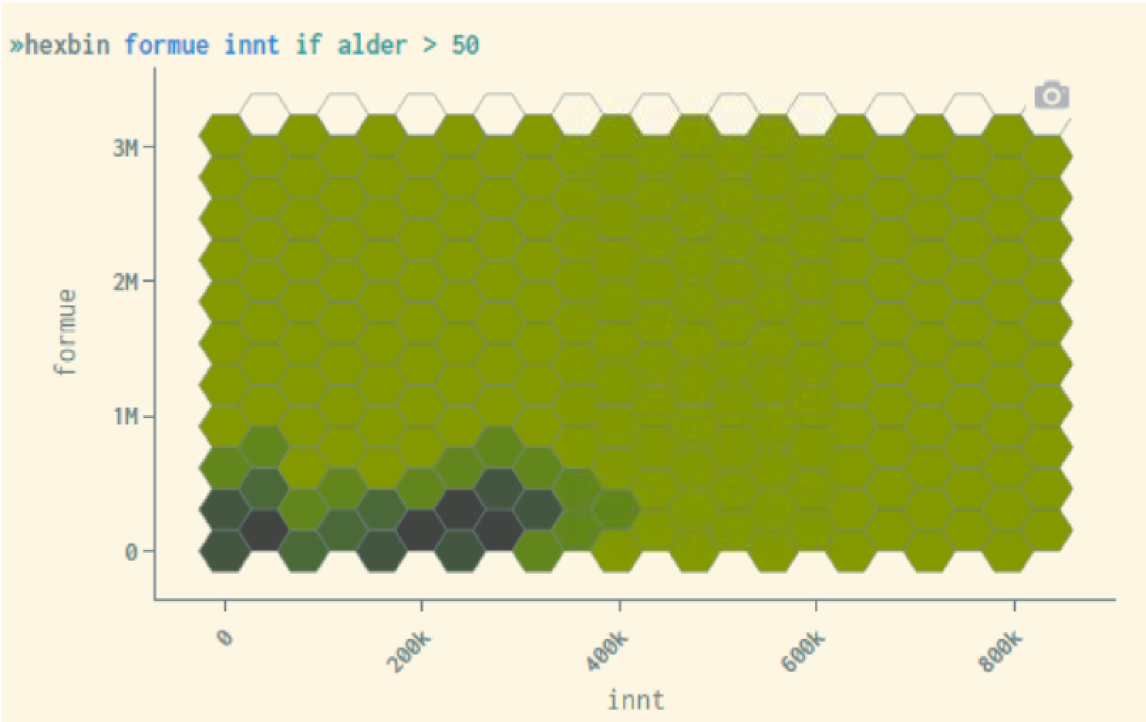
En kan også her holde musepekeren over de ulike felt i figuren - da får en opp de aktuelle underliggende verdier.

Også hexbin-diagram kan filtreres gjennom IF-opsjoner for å vise figur for delpopulasjoner. En kan dessuten justere på antallet hexagoner samt antallet intervaller gjennom opsjoner.

For mer informasjon om denne kommandoen, bruk kommandoen `help hexbin`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

Eksempler:





4.7 Sankey - overgangsdiagram

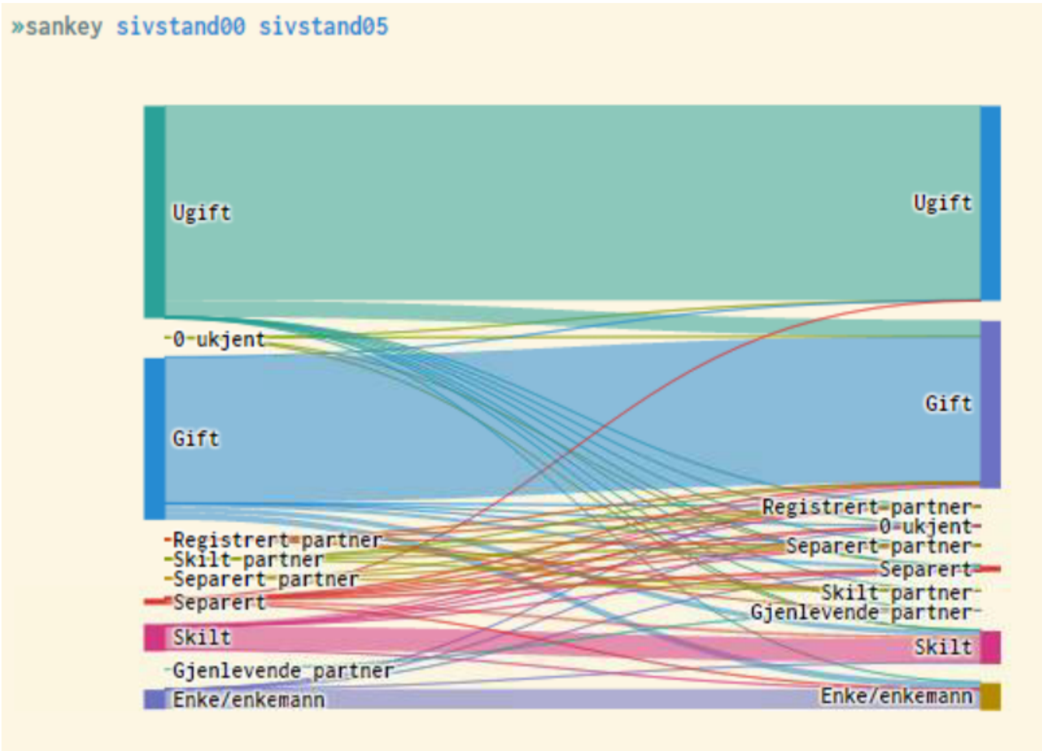
Sankey-diagram er en måte å visualisere overganger mellom statuser. I microdata.no kan dette brukes til å få oversikt over enheters (individers) bevegelser mellom to tidspunkter, enten for samme variabel eller for ulike variabler. En kan se på bevegelser mellom ulike typer tilstander (f.eks. arbeidssøkerstatus -> jobbstatus), eller endringer i fordelinger for samme variabel over tid (f.eks. bosted00 -> bosted05 eller sivilstand00 -> sivilstand05).

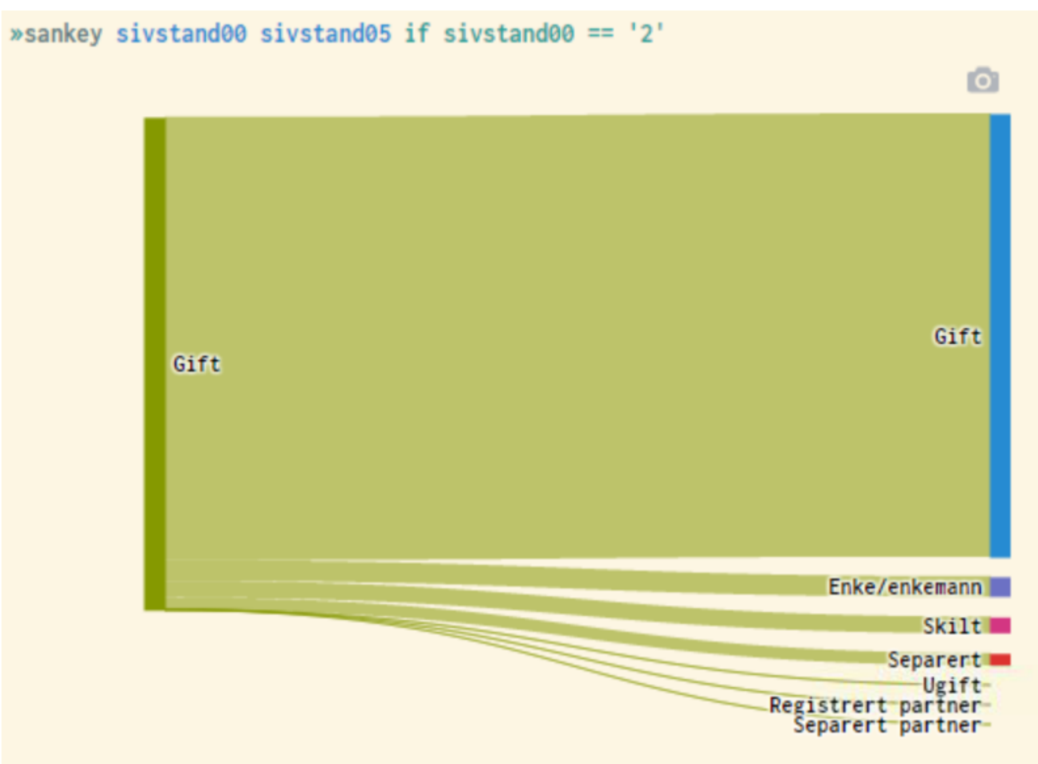
Overgangsvisualiseringen forutsetter at en benytter to kategoriske tverrsnittsvariabler. Antallet kategorier bør ikke være for stort, da diagrammet fort kan bli uleselig. Dette kan løses gjennom å kode om til færre kategorier eller ved å bruke et IF-filter som styrer hvilke overganger en vil se på.

Ved å holde musepekeren over et overgangsfelt vil en få frem antallet enheter som omfattes.

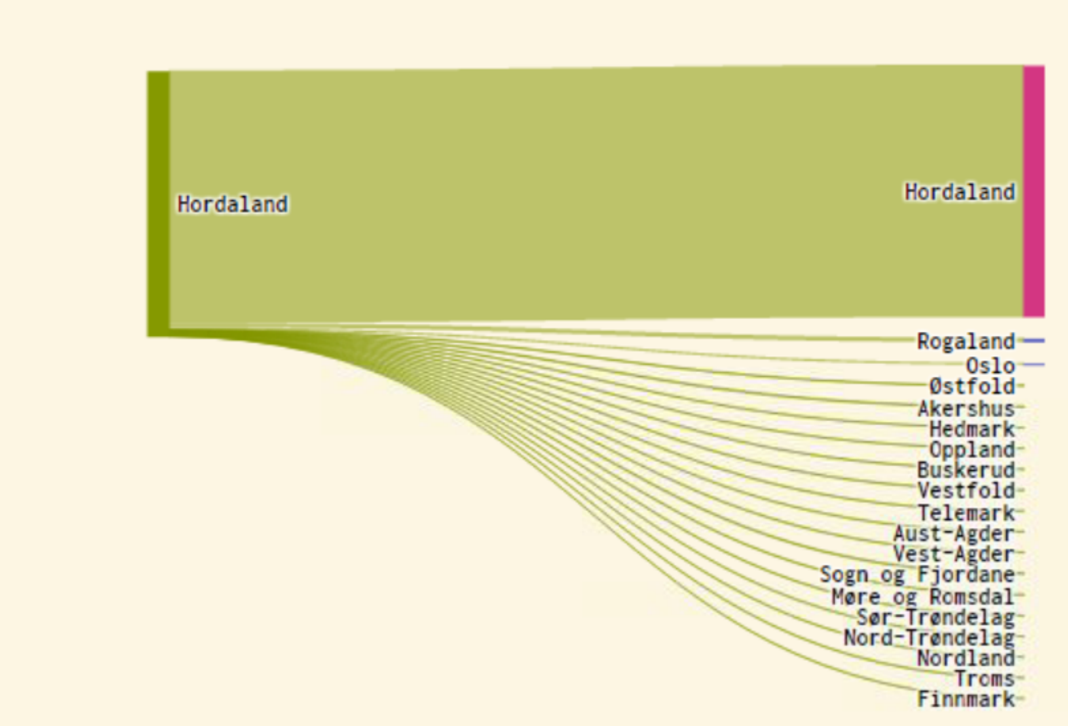
For mer informasjon om denne kommandoen, bruk kommandoen `help sankey`. Dette vil vise syntaxeksempler og en fullstendig liste over tilgjengelige opsjoner som kan brukes til å tilpasse utseende til statistikken som genereres.

Eksempler:

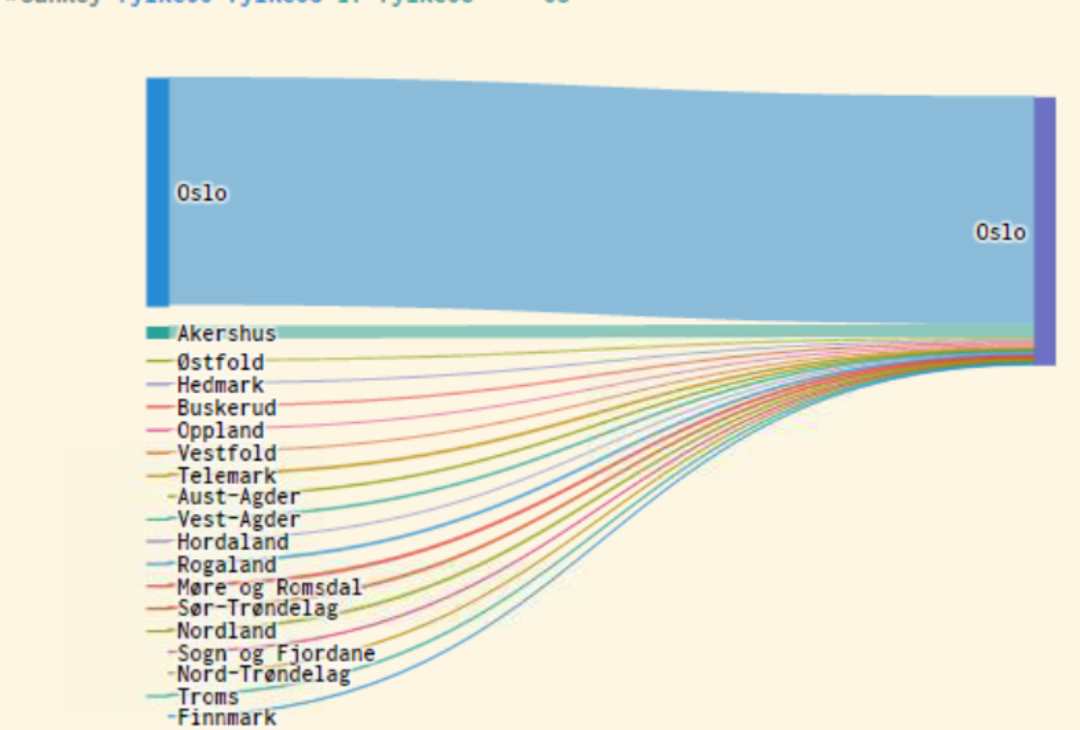

»sankey sivstand00 sivstand05
Ugift
0-ukjent
Gift
Registrert partner
Skilt partner
Separert partner
Separert
Skilt
Gjenlevende partner
Enke/enkemann
Ugift
Gift
Registrert partner
0-ukjent
Separert partner
Separert
Skilt partner
Gjenlevende partner
Skilt
Enke/enkemann

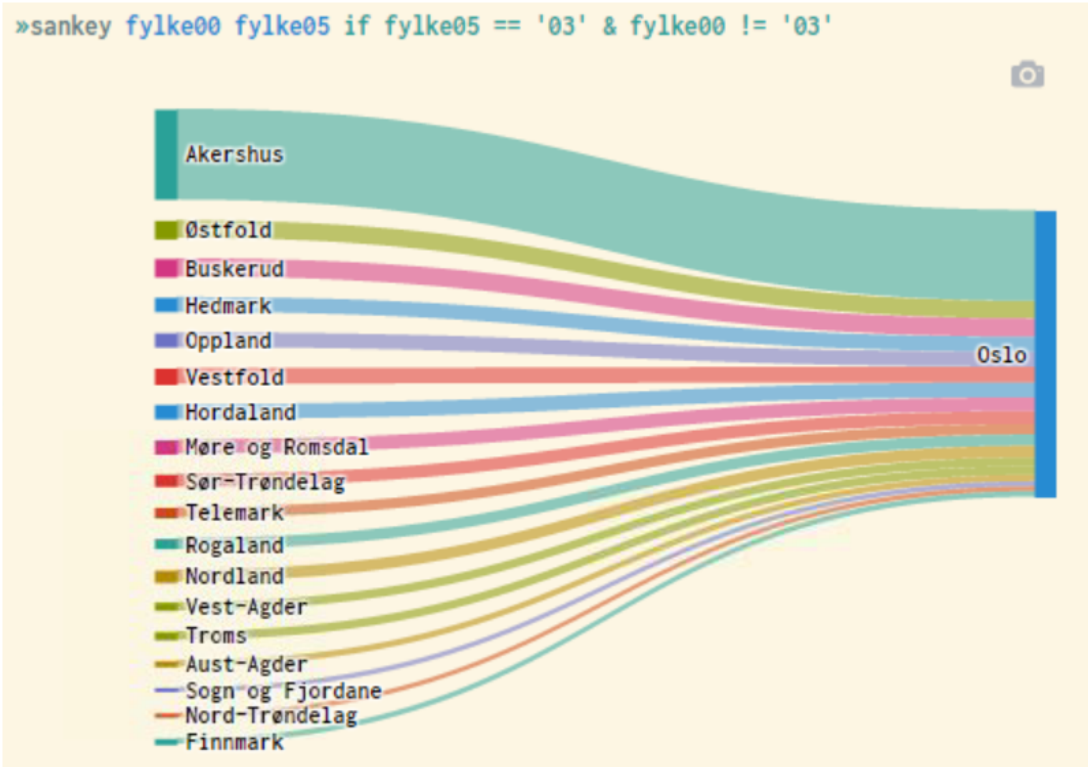

»sankey sivstand00 sivstand05 if sivstand00 == '2'
Gift
Gift
Enke/enkemann
Skilt
Separert
Ugift
Registrert partner
Separert partner

```
»sankey fylke00 fylke05 if fylke00 == '12'
```




```
»sankey fylke00 fylke05 if fylke05 == '03'
```





4.8 Eksempler

Syntaxene nedenfor kan brukes til å gjenskape eksemplene på deskriptiv statistikk som presenteres i kapittel 4. Disse vil være tilgjengelig som ferdige kjørbare skript i microdata.no.

4.8.1 Tabulate

```
require no.ssb.fdb:12 as db

create-dataset demografidata
Import db/INNTEKT_WYRKINNT 2015-01-01 as innt
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/SIVSTANDFDT_SIVSTAND 2015-01-01 as sivstand15
import db/SIVSTANDFDT_SIVSTAND 2019-01-01 as sivstand19
import db/BOSATTEFDT_BOSTED 2015-01-01 as bosted
import db/BEFOLKNING_STATUSKODE 2015-01-01 as regstat
```

```
// Koder om fra kommune- til fylkesnivå
generate fylke = substr(bosted,1, 2)


// Genererer deskriptiv statistikk

// Frekvenstabeller - enveis og toveis
// Den beste måten å gjøre seg kjent med diskrete variabler er gjennom frekvenstabeller. Disse viser antallet enheter for hver kategori, og gir dessuten overblikk over hvilke kategorier som benyttes for den aktuelle variabelen. En kan se på enkeltvariabler gjennom enveistabeller, men også kombinere to eller flere i en og samme tabell (krysstabeller). Dette gir innblikk i hvordan frekvensene fordeler seg, kontrollert for verdier på andre variabler

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06' 'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12' 'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag' '18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring

tabulate fylke
tabulate kjønn
tabulate kjønn regstat
tabulate kjønn fylke

// Krysstabell med kun kategoriverdier (ikke labler)
tabulate kjønn regstat fylke, nolabels

// Krysstabell der missingverdier tas med
tabulate fylke regstat, missing

// Krysstabell der vi bare viser resultatet for personer over 30 år
generate alder = 2015 - int(faarmnd/100)
tabulate fylke regstat if alder > 30

// Prosenttabeller
tabulate sivstand15 sivstand19, rowpct
tabulate sivstand15 sivstand19, colpct
tabulate sivstand15 sivstand19, cellpct
tabulate sivstand15 sivstand19, rowpct freq

// tabulate-kommandoen kan også brukes til å lage volumtabeller vha. summarize-option. Dette viser statistikk som gjennomsnitt m.m. for en gitt variabel fordelt på kategorier spesifisert gjennom de valgte tabulate-variable

tabulate fylke kjønn, summarize(formue)
```

4.8.2 Summarize og boxplot

```
// Nøkkelstatistikk for metriske/kontinuerlige variabler
// Kommandoen summarize brukes for å vise nøkkelinformasjon om metriske/kontinuerlige variabler. Verdier som vises er gjennomsnitt, kvartiler m.m. Kommandoen boxplot viser de samme tallene grafisk gjennom standard boxplotfremstilling

require no.ssb.fdb:12 as db

create-dataset demografidata
import db/INNTEKT_WYRKINNT 2015-01-01 as innt
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/BOSATTEFDT_BOSTED 2015-01-01 as bosted

// Koder om fra kommune- til fylkesnivå
generate fylke = substr(bosted,1,2)

// Lager alder per 2015
generate alder = 2015 - int(faarmnd/100)

summarize innt formue
summarize formue if alder > 50
summarize formue if bosted == '0301'

boxplot innt formue
boxplot innt, over(kjønn)
```

4.8.3 Histogram og barchart

```
// Histogram og barchart

require no.ssb.fdb:12 as db

create-dataset demografidata
import db/INNTEKT_WYRKINNT 2015-01-01 as innt
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue
```

```
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd

// Lager alder per 2015
generate alder = 2015 - int(faarmnd/100)

// Histogram (frekvensfordelinger)
// Dette er en måte å vise frekvensfordelinger for metriske/kontinuerlige variabler på en grafisk måte
der verdiene grupperes i passende intervaller og tilegnes søyler som viser graden av forekomst.
Søylearealene i diagrammet summerer seg til 1 som default, men en kan overstyre dette gjennom
options. Gjennom options kan en dessuten selv velge inndelingen av av verdier (hvor mange søyler en
ønsker), legge på en normalfordelingskurve som referanse m.m.

histogram innt
histogram innt, freq
histogram innt, fraction
histogram innt, percent

histogram innt, normal
histogram innt, bin(6) freq
histogram innt, width(100000) freq

histogram innt, by(kjønn)
histogram innt if innt > 100000


// Ved bruk av discrete-option kan en også lage histogrammer for diskrete variabler. Da vil hver
kategori representeres av respektive søyler

histogram alder, discrete


// Søylediagram
// Slike diagrammer er fine til å fremstille statistikk for kontinuerlige/metriske variabler på en
oversiktlig måte. En kan kombinere flere variabler og bryte ned tallene på kategoriske egenskaper
(kjønn, utdanningsnivå etc)

barchart (mean) innt, over(kjønn)
barchart (mean) innt formue, over(kjønn)
```

4.8.4 Piechart og hexbin-plot

```
require no.ssb.fdb:12 as db

create-dataset demografidata
import db/INNTEKT_WYRKINNT 2015-01-01 as innt
import db/INNTEKT_BRUTTOFORM 2015-01-01 as formue
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/BOSATTEFDT_BOSTED 2015-01-01 as bosted

// Koder om fra kommune- til fylkesnivå
generate fylke = substr(bosted,1,2)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring


// Genererer alder per 2015
generate alder = 2015 - int(faarmnd/100)

// Kakediagram
// Dette er en fin måte å vise den prosentvise fordelingen for diskrete variabler på en grafisk måte

drop if alder < 16

piechart kjønn
piechart fylke


// Hexbinplot
// Dette er en anonymiserende måte å fremstille plotdiagrammer (egner seg best for
kontinuerlige/metriske variabler), der tettheten i plottene fargelegges på en systematisk måte for å
avdekke mønstre i fordelingen mellom to variabler

hexbin formue innt
hexbin formue innt if alder > 50
```

4.8.5 Sankey-diagram

```
// Overgangsdiagram (Sankey)

require no.ssb.fdb:12 as db

create-dataset demografidata
import db/SIVSTANDFDT_SIVSTAND 2010-01-01 as sivstand10
import db/SIVSTANDFDT_SIVSTAND 2015-01-01 as sivstand15
import db/BOSATTEFDT_BOSTED 2010-01-01 as bosted10
import db/BOSATTEFDT_BOSTED 2015-01-01 as bosted15

// Koder om fra kommune- til fylkesnivå
generate fylke10 = substr(bosted10,1,2)
generate fylke15 = substr(bosted15,1,2)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke10 fylkerstring
assign-labels fylke15 fylkerstring

sankey fylke10 fylke15 if fylke10 == '12'
sankey fylke10 fylke15 if fylke15 == '03'
sankey fylke10 fylke15 if fylke15 == '03' & fylke10 != '03'

sankey sivstand10 sivstand15
sankey sivstand10 sivstand15 if sivstand10 == '2'
```

5. Avansert analyse

I tillegg til de deskriptive mulighetene som finnes i microdata.no, kan en også foreta avanserte analyser i form av regresjonsanalyser m.m. Foreløpig er følgende analysemetoder tilgjengelige i microdata.no:

- `correlate`
- `anova`
- `normaltest`
- `regress`
- `ivregress`
- `oaxaca`
- `logit / probit`
- `mlogit`
- `regress-panel`
- `predict`-kommandoer for uthenting av prediksjons- og residualverdier m.m.

Mer analysefunksjonalitet vil bli lagt til systemet fortløpende, i tråd med tilbakemeldinger fra brukere. I prinsippet kan alt av analysefunksjonalitet som finnes i Stata også implementeres i microdata.no.

5.1 Correlate - korrelasjon

Kommandoen `correlate` brukes til å analysere statistisk sammenheng mellom variabler, altså korrelasjon. Det rapporteres verdier mellom -1 og 1 i en korrelasjonsmatrise for de spesifiserte variablene, der minus- og pluss-verdier impliserer hhv. negativ og positiv sammenheng. Verdien 0 indikerer ingen sammenheng. Jo nærmere +/- 1 en kommer, jo sterkere korrelasjon.

Syntax:

```
correlate <variabelliste> [if <betingelse>] [, <opsjoner>]
```

Dersom en ikke oppgir variabelliste, vises en korrelasjonsmatrise for alle variablene i datasettet.

En kan gjennom opsjoner vise alternative mål:

- `covariance` Viser kovarians i stedet for korrelasjonskoeffisient
- `pairwise` Parvis fremvisning
- `obs` Viser antall observasjoner som ligger bak hver korrelasjonskoeffisient

- `sig` Viser signifikansverdien for hver korrelasjonskoeffisient

Eksempler:

```
»correlate alder formue
```

	formue
alder	0.3291

```
»correlate alder formue, obs
```

		formue
alder	corr	0.3291
	obs	3172121

```
»correlate alder formue, sig
```

		formue
alder	corr	0.3291
	sig	0

5.2 Anova

Anova-tester kan ses på som en forenklet regresjonsanalyse, der en undersøker om gjennomsnittsverdien til en avhengig kontinuerlig variabel er forskjellig i to eller flere uavhengige grupper gitt ved en annen kategorisk variabel. Ett eksempel er å teste om gjennomsnittslønnen er forskjellig for personer med hhv. lav, middels og høy utdannelse (benytter en uavhengig variabel der utdanningsnivå deles inn i tre grupper).

En Anova-test kan sjekke om det eksisterer signifikante forskjeller mellom minst to av gruppene (gitt ved den uavhengige variabelen), men gir ikke svar på hvilke(n) gruppe(r) dette gjelder. Til dette må en foreta en regresjonsanalyse (se kapittel 5.4).

Syntax:

```
    anova <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Om en tester for kun 2 variabler, dvs. én avhengig og én uavhengig variabel, foretar man en enveis Anova-test. En kan også teste en avhengig variabel opp mot to andre kategoriske uavhengige variabler. Dette kalles toveis Anova-test.

Eksempel:

```
»anova innt05 mann
    Antall Obs: 2489943          Root MSE:
            R²: 0.0746      1.905535e+5
                               Justert R²: 0.0746
```

Kilde	Del. SS	Frihetsgrad	MS	F	Sanns > F
Residual	9.04113e+16	2.489941e+6	3.631062e+10	-	-
mann	7.291912e+15	1	7.291912e+15	2.0082e+5	0

5.3 Normaltest

Normaltest-kommandoen kjører et utvalg tester for normalfordeling for valgte variabler, eller hele datasett dersom ingen variabler er oppgitt. For hver test er måltall og p-verdi oppgitt. Testene som kjøres er skewness, kurtosis, s-k (ikke justert), Jarque-Bera og Shapiro-Wilk.

Syntax:

```
normaltest <variabelliste> [if <betingelse>]
```

Eksempel der en tester for normalfordeling for variabelen `innt19` (yrkesinntekt målt i 2019). P-verdier lavere enn 0.05 betyr at fordelingen ikke er normalfordelt, og vice versa:

```
demografidata» normaltest innt19
```

		Test	P
innt19	Skjevhet	2300.9636637	0
	Kurtose	1295.6776859	0
	Shapiro-Wilk	0.6435465	0
	Jarque-Bera	1.0114248e+12	0
	Normaltest	6973214.4474433	0

5.4 Regress - ordinær lineær regresjonsanalyse

Kommandoen `regress` brukes til å utføre en ordinær lineær regresjonsanalyse (OLS) der den avhengige variabelen er en kontinuerlig/metrisk variabel. Et typisk eksempel er inntekt.

Syntax:

```
   regress <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Den avhengige variabelen må angis først, etterfulgt av forklaringsvariablene. Opsjoner kan benyttes for ulike formål, som f.eks. robust- eller cluster-estimering, jfr. underkapitlene nedenfor. I likhet med andre statistiske kommandoer, kan også regresjonskommandoer kombineres med en if-betingelse for å kjøre regresjoner på utvalgte grupper. For full oversikt over muligheter, bruk kommandoen `help regress`.

Kort fortalt går modellen ut på å estimere (mulige) marginaleffekter av et sett med uavhengige variabler (forklaringsvariabler) på den avhengige variabelen (responsvariabelen). Marginaleffekt er et mål på hvor mye den avhengige variabelen estimeres til å øke i verdi dersom en uavhengig variabel øker med én måleenhet.

Det viktigste å se på når en skal tolke resultatet av en regresjonskjøring er forklaringskraften til:

a) Modellen som helhet
b) Hver enkelt variabel

Dette gjøres ved å studere de respektive signifikansverdiene "*Justert* R^2" og "$P > |t|$".

Nedenfor vises et eksempel på resultatet av en regresjonskjøring i microdata.no. Tallene i den nederste delen knyttes til de ulike variablene, mens tallene øverst viser til analysemodellen som helhet.

```
»regress innt05 mann gift alder formuehøy
```

Kilde	SS	df	MS
Modell	1.074715e+16	4	2.686788e+15
Residual	8.695606e+16	2.489938e+6	3.492298e+10
Total	9.770321e+16	2.489942e+6	3.923915e+10

Antall Obs: 2489943
F(4, 2489938): 7693
R²: 0.1099
Justert R²: 0.1099
Root MSE: 1.868769e+5

innt05	Coef.	Std. Avvik	t	P>|t|	[% Konf.	Intervall]
alder	-1157.7	10.741	-107.78	0	-1178.8	-1136.7
formuehøy	88753.	387.74	228.89	0	87993.	89513.
gift	55131.	276.38	199.47	0	54590.	55673.
mann	99052.	242.13	409.07	0	98577.	99526.
Konst	2.455021e+5	393.99	623.1	0	2.447299e+5	2.462744e+5

Justert R^2 gir et samlemål for hvor mye av den observerte variansen i den avhengige variabelen som kan forklares av summen av de uavhengige variablene. Skalaen går fra 0 til 1, der nærmest mulig 1 er det ideelle. I praksis vil en aldri nå verdien 1 ved analyser av sosioøkonomiske individdata pga. tilfeldig støy og uobserverte årsakssammenhenger, og typiske verdier vil gjerne ligge i intervallet 0 - 0.5.

R^2 vil alltid øke i verdi for hver ekstra uavhengige variabel som legges til. Dette betyr ikke nødvendigvis at modellen blir bedre, spesielt om variablene en legger til *ikke* er signifikante. *Justert R^2* tar hensyn til dette og vil kun øke i verdi dersom de ekstra variablene er signifikante.

Dersom *justert R^2* får *lavere* verdi ved å tilføye en ekstra uavhengig variabel, er dette en indikasjon på den valgte variabelen kan ha en relativt høy grad av korrelasjon med noen av de andre uavhengige variablene, dvs. multikollinearitet. Dette er absolutt noe en bør unngå.

$P > |t|$ eller *p-verdiene* (i kolonne 4 i den nedre hovedtabellen) angir sannsynligheten for at *t*-verdien fremkommer som et resultat av ren tilfeldighet. For å kunne si at en variabel er signifikant, må den tilhørende *p*-verdien være lavere enn 0.05 ved et 5%-signifikansnivå. Verdier nærmest mulig 0 er ideelt.

Verdien *t* (kolonne 3) er kort fortalt et standardisert mål for koeffisientverdien (=marginaleffekten), jfr. verdiene i *Coef.*-kolonnen (kolonne 1), der en ved et 5%-signifikansnivå får grenseverdier på +/- 1.96. Verdier som overstiger 1.96 med positivt eller negativt fortegn vil altså regnes som signifikante på et 5%-nivå (5%-nivå er den grenseverdien som er vanlig å operere med).

En kan også se på 95%-konfidensintervallet presentert i de to kolonnene lengst til høyre i hovedtabellen. Dersom intervallet inkluderer verdien 0, kan en utelukke at den aktuelle koeffisienten viser en signifikant sammenheng mellom den tilhørende uavhengige variabelen og responsvariabelen.

Koeffisientverdiene i kolonne 1 er kun relevante for signifikante variabler, og viser marginaleffekten på responsvariabelen av en enhets økning i verdien på den tilhørende uavhengige variabelen.

I eksempelet ovenfor ser en at alle variablene er signifikante med god margin (høye t-verdier). Alder har negativ effekt på inntekt, mens de øvrige variabler har positiv effekt. *"Konst"* peker på konstantleddet, dvs. startverdien på responsvariabelen når alle uavhengige variabler har verdien 0, og har ikke noen stor betydning rent tolkningsmessig.

5.4.1 Faktorvariabler

Faktorvariabler kan brukes til å automatisere omkoding av flerkategorivariabler slik at de kan brukes i et regresjonsuttrykk. I praksis vil hver kategori minus referansekategorien representeres ved separate dummyvariabler, der en tolkningsmessig måler effekten av de enkelte kategorier sammenliknet med referansekategorien. En bruker da prefikset `i.` foran variabelnavnet i det aktuelle regresjons-uttrykket. Den laveste verdien vil som standard benyttes som referanseverdi.

Faktorvariabler kan også brukes til å estimere effekten av kombinasjoner av verdier for utvalgte kategoriske variabler (i tillegg til effekten hver enkelt forklaringsvariabel har hver for seg). Rasjonalet bak er at enkelte egenskaper har ulik effekt på den avhengige variabelen når en ser på ulike grupper. F.eks. kan effekten av utdanning på fremtidig inntekt være systematisk forskjellig for menn versus kvinner. Om en har slike antakelser, kan faktorvariabler komme til nytte.

En angir faktorvariabler og kombinasjoner av disse i regresjonsuttrykk på samme måte som i Stata. Prefikset `i.` brukes altså til å angi at en variabel er kategorisk, mens symbolet `#` brukes til å angi at alle kategorier bortsett fra referansegruppene skal kombineres og estimeres gjennom respektive koeffisientestimat. Ved bruk av `##` angir en at også hver enkelt kategori hver for seg skal estimeres og inngå i regresjonsanalysen.

Eksempel:
Lineær regresjonsanalyse med `innt19` (yrkesinntekt i 2019) som den avhengige variabelen. De uavhengige variablene er `mann, utdanningsnivå`, og alle undergrupper av de to variablene kombinert med hverandre, bortsett fra referansegruppen:

```
regress innt19 i.mann i.utdanningsnivå utdanningsnivå#mann
```

Resultat:

```
demografidata» regress innt19 i.mann i.utdanningsnivå utdanningsnivå#mann
```

Kilde	SS	df	MS
Modell	89493449031967680	17	5.2643205e+15
Residual	473374297090160100	2.849433e+6	1.661293e+11
Total	562867746122127740	2.84945e+6	1.9753558e+11

Antall Obs:	2849451
F(17, 2849433):	31688.092777
R²:	0.15899
Justert R²:	0.15899
Root MSE:	4.0758962e+5

innt19	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
mann 1	1.3303472e+5	13515.2	9.84328	0	1.0654525e+5	1.5952419e+5
utdanningsnivå 1 × mann 1	-62208.5	15641.7	-3.97709	0.00007	-92865.8	-31551.3
utdanningsnivå 2 × mann 1	-12423	13556.1	-0.91641	0.35945	-38992.6	14146.6
utdanningsnivå 3 × mann 1	16534.7	13702	1.20673	0.22753	-10320.8	43390.2
utdanningsnivå 4 × mann 1	54116.1	13543.4	3.99574	0.00006	27571.4	80660.7
utdanningsnivå 5 × mann 1	1.3340997e+5	13806.3	9.66294	0	1.0635002e+5	1.6046991e+5
utdanningsnivå 6 × mann 1	57306.9	13548.4	4.22979	0.00002	30752.5	83861.3
utdanningsnivå 7 × mann 1	1.3155888e+5	13598.2	9.67468	0	1.0490677e+5	1.5821099e+5
utdanningsnivå 8 × mann 1	85630.4	14292.6	5.99122	0	57617.3	1.1364354e+5
utdanningsnivå 1	12109.6	11206	1.08063	0.27985	-9853.81	34073.1
utdanningsnivå 2	-7298	9571.06	-0.7625	0.44575	-26056.9	11460.9
utdanningsnivå 3	1.1586814e+5	9669.74	11.9825	0	96915.7	1.348205e+5
utdanningsnivå 4	1.1891478e+5	9561.66	12.4366	0	1.0017425e+5	1.3765532e+5
utdanningsnivå 5	2.0559287e+5	9812.71	20.9516	0	1.8636029e+5	2.2482545e+5
utdanningsnivå 6	2.6033553e+5	9556.87	27.2406	0	2.4160439e+5	2.7906666e+5
utdanningsnivå 7	4.405264e+5	9596.47	45.905	0	4.2171765e+5	4.5933516e+5
utdanningsnivå 8	5.124071e+5	10152.7	50.4696	0	4.9250803e+5	5.3230618e+5
Konst	2.3898019e+5	9538.33	25.0546	0	2.2028538e+5	2.5767499e+5

Dette alternativet gir samme resultat:

```
regress innt19 utdanningsnivå##mann
```

Prefikset `c.` kan benyttes til å signalisere at en variabel skal regnes som en kontinuerlig variabel (ikke-kategorisk). Dette kan være aktuelt å bruke i de tilfeller hvor en variabel kan tolkes som kontinuerlig, f.eks. utdanningsnivå. Følgende uttrykk kjører en liknende regresjon som over, men der utdanningsnivå anses som en kontinuerlig variabel:

```
regress innt19 i.mann c.utdanningsnivå utdanningsnivå#mann
```

Resultat:

```
demografidata» regress innt19 i.mann c.utdanningsnivå utdanningsnivå#mann
```

Kilde	SS	df	MS
Modell	8.9493449e+16	17	5.2643205e+15
Residual	4.733743e+17	2.849433e+6	1.661293e+11
Total	5.6286775e+17	2.84945e+6	1.9753558e+11

Antall Obs: 2849451
F(17, 2849433): 31688.092777
R^2: 0.15899
Justert R^2: 0.15899
Root MSE: 4.0758962e+5

innt19	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
utdanningsnivå	58638.7	1194.24	49.101	0	56298	60979.4
utdanningsnivå 1 × mann 0	-46529.1	10526.9	-4.41999	0.00001	-67161.5	-25896.6
utdanningsnivå 2 × mann 0	-1.2457553e+5	8078.72	-15.4201	0	-1.4040955e+5	-1.0874151e+5
utdanningsnivå 3 × mann 0	-60048.1	7627.63	-7.87244	0	-74998	-45098.2
utdanningsnivå 4 × mann 0	-1.1564027e+5	7081.39	-16.3301	0	-1.2951956e+5	-1.0176098e+5
utdanningsnivå 5 × mann 0	-87600.9	7156.52	-12.2407	0	-1.0162748e+5	-73574.4
utdanningsnivå 6 × mann 0	-91497	6820.51	-13.4149	0	-1.0486502e+5	-78129
utdanningsnivå 7 × mann 0	30055	7042.62	4.26759	0.00002	16251.7	43858.3
utdanningsnivå 8 × mann 0	43297	7902.05	5.4792	0	27809.2	58784.7
mann 1	1.3303472e+5	13515.2	9.84328	0	1.0654525e+5	1.5952419e+5
utdanningsnivå 1 × mann 1	-1.0873769e+5	10209.8	-10.6503	0	-1.2874857e+5	-88726.8
utdanningsnivå 2 × mann 1	-1.3699855e+5	8098.19	-16.9171	0	-1.5287072e+5	-1.2112638e+5
utdanningsnivå 3 × mann 1	-43513.4	7651.56	-5.68686	0	-58510.2	-28516.6
utdanningsnivå 4 × mann 1	-61524.1	7087.67	-8.68044	0	-75415.7	-47632.5
utdanningsnivå 5 × mann 1	45809	6997.09	6.54686	0	32094.9	59523
utdanningsnivå 6 × mann 1	-34190.1	6829.01	-5.00659	0	-47574.7	-20805.4
utdanningsnivå 7 × mann 1	1.6161394e+5	7032.37	22.9814	0	1.4783073e+5	1.7539715e+5
utdanningsnivå 8 × mann 1	1.2892746e+5	7770.6	16.5916	0	1.1369734e+5	1.4415758e+5
Konst	2.3898019e+5	9538.33	25.0546	0	2.2028538e+5	2.5767499e+5

5.4.2 Modelldiagnostikk

Det er mulig å foreta tester av regresjonsmodellen opp mot dataene en analyserer, for på denne måten å sjekke om den valgte modellen trenger å modereres. Dette gjøres ved å angi opsjoner for hvilke test-parametre en vil vise frem resultatet av. Regresjonsresultatet vil da også vise parameterverdiene under hovedtabellen.

Følgende opsjoner kan benyttes for modelltesting:

- `ov`: Ramseys RESET test for utelatte variabler. Viser en samlet F-verdi med tilhørende P-verdi
- `vif`: Variance inflation factor test for multikollinearitet. Viser variance inflation factor (VIF)-verdier for de uavhengige variablene i en tabell
- `het_bp`: Breusch-Pagan test for heteroskedastisitet. Viser en samlet chikvadrat-verdi med tilhørende P-verdi
- `het_iid`: Studentisert Breusch-Pagan test for heteroskedastisitet. Viser en samlet chikvadrat-verdi med tilhørende P-verdi (en nyere versjon av BP-testen som er mer robust siden den ikke forutsetter at residualene er normalfordelt)
- `het_fstat`: F-statistikk fra Breusch-Pagan test for heteroskedastisitet. Viser en samlet F-verdi med tilhørende P-verdi. Antallet frihetsgrader baserer seg ikke på antall variabler i regresjonsmodellen, men stammer fra en ekstra OLS-modell som sammenligner residualer og predikerte verdier. Derfor kun én frihetsgrad

Eksempel der en tester for utelatte variabler, multikollinearitet og heteroskedastisitet:

```
regress innt19 mann gift alder formuehøy, ov vif het_bp
```

Resultat (henter ut alle test-parametrene):

```
demografidata» regress innt19 mann gift alder formuehøy, ov vif het_bp het_iid het_fstat
```

Kilde	SS	df	MS
Modell	88819204460328320	4	22204801115082080
Residual	491419147639550460	2.958921e+6	1.6608052e+11
Total	580238352099878800	2.958925e+6	1.9609769e+11

Antall Obs:	2958926
F(4, 2958921):	133699.007529
R²:	0.15307
Justert R²:	0.15307
Root MSE:	4.0752978e+5

innt19	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
mann	1.3352883e+5	476.411	280.28	0	1.3259508e+5	1.3446258e+5
gift	87311.8	541.892	161.124	0	86249.7	88373.9
alder	4631.46	19.9379	232.293	0	4592.39	4670.54
formuehøy	2.7244037e+5	658.837	413.516	0	2.7114907e+5	2.7373167e+5
Konst	1.5790516e+5	787.586	200.492	0	1.5636152e+5	1.594488e+5

Breusch-Pagan

chi2(1): 2154648.283657
Prob > chi2: 0

Breusch-Pagan, studentisert

chi2(1): 1105.523026
Prob > chi2: 0

Breusch-Pagan, f-test

F(1, 2958924): 1105.935482
Prob > F: 0

Ramseys RESET test

F(3, 2958918): 4677.063546
Prob > F: 0

Variance inflation factor

	VIF	1/VIF
mann	1.009167	0.990917
gift	1.242664	0.804723
alder	1.346809	0.742496
formuehøy	1.114218	0.89749
Gj.snitt	1.178215	-

Det finnes også andre metoder i microdata.no for å teste regresjonsmodeller:

- Kommandoen `correlate` gir parvise mål på korrelasjon mellom enkeltvariabler i en tabellmatrise. Dette kan avdekke multikollinearitet. I kapittel 5.1 gjennomgås denne kommandoen.

- Kommandoen `regress-predict` gir mulighet for å studere bl.a. residual- og prediksjonsverdier. Dette kan kombineres med grafiske verktøy som `histogram` for å vise residualfordelinger og for å sjekke for f.eks. normalfordeling. Kommandoen `hexbin` kan dessuten vise et anonymisert 2-veis plot. Se kapittel 5.4.4 for en gjennomgang av denne kommandoen.

5.4.3 Cluster- og robust-estimering

Opsjonene `robust` og `cluster()` brukes hver for seg til å spesifisere om en ønsker hhv. robust- eller cluster-estimering, og vil som resultat presentere regresjonsestimater med justerte standardavvik for de estimerte koeffisienter. Også tilhørende t-, z- og p-verdier påvirkes. Øvrige verdier påvirkes ikke sammenliknet med standard estimering.

Merk at `robust` og `cluster` ikke kan benyttes i kombinasjon (`cluster` impliserer robust estimering).

Robust estimering kan brukes der det er mistanke om problematiske “outliers” eller heteroskedastisitet.

Cluster-estimering brukes når en mistenker at det er systematiske avhengigheter innen grupper av observasjoner, f.eks. innen skoler eller kommuner. Gruppene spesifiseres gjennom en variabel (cluster-variabel) som inngår i parentesen til cluster-opsjonen, f.eks. `cluster(skole)` eller `cluster(kommune)`. Følgende forutsetninger gjelder, hvis ikke vil systemet gi en feilmelding:

- Antallet grupper må være av en viss størrelse
- Clustervariabelen må være numerisk
- Clustervariabelen kan ikke inngå som variabel i regresjonsuttrykket.

Eksempler:
```
regress inntekt mann gift høy_utdanning, robust
regress inntekt mann gift høy_utdanning, cluster(kommune)
```

Robust- og cluster-opsjoner kan benyttes også på øvrige regresjonstyper.

5.4.4 Prediksjonsverdier og residualverdier

Alle regresjonsvarianter som finnes i microdata.no har tilknyttede kommandoer som genererer blant annet residual- og prediksjonsverdier. Dette er verdier som kan brukes til å analysere dataspredningen og for testing av regresjonsmodeller. Prediksjonsverdier kan dessuten brukes som input til videre analyser.

Kommandoene har samme navn som tilhørende regresjonskommando pluss "`-predict`"

Syntax:

```
    regress-predict <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Variablene angis på samme måte som for den tilhørende regresjons-modellen som kjøres med kommandoen `regress`.

Følgende verdier kan hentes ut: Prediksjonsverdier, residualer og "Cook's distance"

En bestemmer selv hvilke verdier en vil generere gjennom bruk av opsjoner. Resultatet av kjøringene er et sett med variabler som inneholder de ulike verdiene. Som standard genereres førstnevnte verditype, men det anbefales likevel å spesifisere dette gjennom opsjoner ettersom en da også kan bestemme navn på de genererte variablene inni en parentes som vist i syntax-eksempelet nedenfor. Om en kjører flere "predict-kommandoer", må en lage nye navn for de automatisk genererte variablene.

Syntax-eksempel:

```
    regress-predict lønn alder mann formue, residuals(res)
    predicted(pred) cooksd(cook)
```

De automatisk genererte variablene kan brukes som input til videre analyser eller til å vises grafisk. Aktuelle grafiske kommandoer er `hexbin` og `histogram`. Ved å kjøre `histogram` på residualvariabelen, kan en sjekke hvorvidt residualene er normalfordelte. Hexbin-kommandoen kan dessuten brukes til lage anonymiserte spredningsplott der en kombinerer to sett med verdier.

For mer detaljer anbefales det å bruke kommandoen `help regress-predict`

5.4.5 Grafisk visning av koeffisientestimater

Ved bruk av mange variabler eller faktorledd i en regresjonsmodell kan det være krevende å gå gjennom lister med koeffisientestimater. I slike tilfeller kan det være nyttig å vise estimatene grafisk gjennom et standard koeffisientplot: `coefplot`

Kommandoen `coefplot` tar hele regresjonsmodeller som argument, inkludert opsjoner. Den grafiske visningen vil derfor reflektere de ulike opsjonene man har valgt for en regresjonsmodell. F.eks. dersom man bruker opsjonen `level()` for å justere på signifikansnivået (standard er 5%), så vil man se at konfidensintervallene vil justere seg deretter.

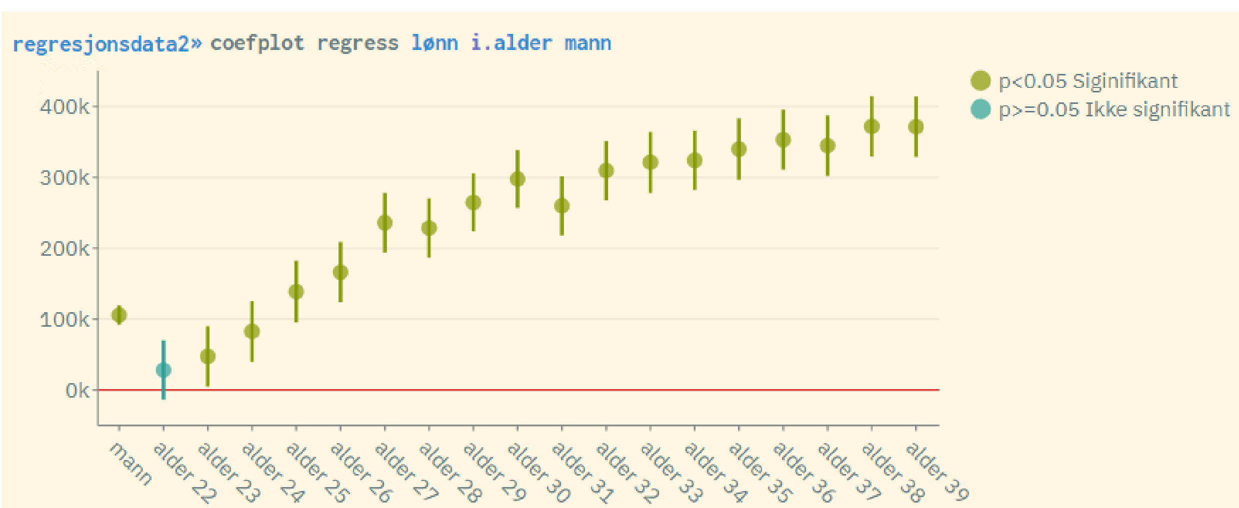
Eksempler på kommandosyntax:

```
coefplot regress lønn alder mann formue, standardize

coefplot regress lønn i.alder mann høy_utd høy_formue if
inrange(alder, 20, 40)
```

Koeffisientplot viser de estimerte koeffisientverdiene med tilhørende konfidensintervall, som resulteres av kjøring av den spesifikke regresjonsmodellen. Verdier/konfidensintervaller fargelegges basert på om estimatet er signifikant eller ikke, slik at dette blir lettere å se.

Eksempel:



Siden `coefplot` kun rapporterer koeffisientestimatene og tilhørende konfidensintervall, anbefales det å kjøre regresjonskommandoen i kombinasjon med koeffisientplot. Da får man hentet ut alle tallene man trenger. F.eks.:

```
regress lønn i.alder mann høy_utd oslo
coefplot regress lønn i.alder mann høy_utd oslo
```

5.5 IV-regress - lineær regresjonsanalyse med instrumentvariabler

Om en mistenker avhengighet mellom uavhengige variabler i en lineær regresjonsmodell, kan en bruke kommandoen `ivregress` til å sette opp et uttrykk der en spesifiserer hvilke variabler dette gjelder.

Syntax:

```
ivregress <variabel> <variabelliste> (<variabelliste> =
<variabelliste>) [if <betingelse>] [, <opsjoner>]
```

Den avhengige variabelen må angis først, etterfulgt av forklaringsvariablene og instrument-uttrykket som angis inni parentes. Opsjoner kan benyttes for ulike formål, som f.eks. robust- eller cluster-estimering, jfr. underkapitlene nedenfor. I likhet med andre statistiske kommandoer, kan også regresjonskommandoer kombineres med en if-betingelse for å kjøre regresjoner på utvalgte grupper. For full oversikt over muligheter, bruk kommandoen `help ivregress`.

Eksempel der en mistenker at formue henger sammen med alder og bosted (= Oslo):

```
ivregress lønn mann (formuehøy = alder oslo)
```

Resultatet av kjøringen er standard regresjonsresultat der instrumentvariabel og instrumenter listes opp under tabellen. I praksis behandles alle uavhengige variabler som instrumenter, bortsett fra variabelen som defineres som instrumentvariabel.

Eksempel der `formuehøy` er instrumentvariabel og `alder` er instrument (`mann` inngår også som instrument selv om variabelen ikke er spesifisert i parentesuttrykket):

```
ivtest» ivregress lønn mann (formuehøy = alder)
```

Kilde	SS	df
Modell	-2.4773801e+17	3
Residual	6.793658e+17	2.631922e+6
Total	4.3162778e+17	2.631925e+6

Antall Obs:	2631925
chi2(2):	208368.33996
Prob > chi2:	0
R²:	-0.57396
Justert R²:	-0.57396
Root MSE:	5.0806009e+5

lønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
mann	69580.9	651.824	106.748	0	68303.4	70858.5
formuehøy	1.2590452e+6	3167.96	397.43	0	1.2528361e+6	1.2652543e+6
Konst	2.6450893e+5	599.623	441.124	0	2.6333369e+5	2.6568417e+5

Instrumenterte: formuehøy
Instrumenter: mann alder

5.5.1 Faktorvariabler

Se kapittel 5.4.1 for informasjon om hvordan faktorvariabler kan brukes. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.5.2 Modelldiagnostikk

Diverse diagnostikk knyttet til instrumentvariabel-modellering er under utvikling og testing, og vil snart bli tilgjengelig. Dette inkluderer standard testestimatorer for endogenitet, korrelasjon og overidentifisering. Inntil da kan en teste modelleringen på følgende måter:

- Kjøre regresjon med og uten instrumentering, og sammenligne resultatet: `regress` vs. `ivregress`
- Bruke kommandoen `correlate` til å sjekke for korrelasjon mellom utvalgte variabler
- Studere residualer m.m. vha. kommandoen `ivregress-predict`

5.5.3 Cluster- og robust-estimering

Se kapittel 5.4.3 for informasjon om hvordan benytte cluster- eller robust-estimering. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.5.4 Prediksjonsverdier og residualverdier

Alle regresjonsvarianter som finnes i microdata.no har tilknyttede kommandoer som genererer blant annet residual- og prediksjonsverdier. Dette er verdier som kan brukes til å analysere dataspredningen og for testing av regresjonsmodeller. Prediksjonsverdier kan dessuten brukes som input til videre analyser.

Kommandoene har samme navn som tilhørende regresjonskommando pluss "`-predict`"

Syntax:

```
ivregress-predict <variabel> <variabelliste> (<variabelliste> =
<variabelliste>) [if <betingelse>] [, <opsjoner>]
```

Variablene angis på samme måte som for den tilhørende regresjons-modellen som kjøres med kommandoen `ivregress`.

Følgende verdier kan hentes ut: Prediksjonsverdier og residualer

En bestemmer selv hvilke verdier en vil generere gjennom bruk av opsjoner. Resultatet av kjøringene er et sett med variabler som inneholder de ulike verdiene. Som standard genereres førstnevnte verditype, men det anbefales likevel å spesifisere dette gjennom opsjoner ettersom en da også kan bestemme navn på de genererte variablene inni en parentes som vist i syntax-eksempelet nedenfor. Om en kjører flere "predict-kommandoer", må en lage nye navn for de automatisk genererte variablene.

Syntax-eksempel:

```
ivregress-predict lønn mann (formue = alder), residuals(res3)
predicted(pred3)
```

De automatisk genererte variablene kan brukes som input til videre analyser eller til å vises grafisk. Aktuelle grafiske kommandoer er `hexbin` og `histogram`. Ved å kjøre `histogram` på residualvariabelen, kan en sjekke hvorvidt residualene er normalfordelte. Hexbin-kommandoen kan dessuten brukes til lage anonymiserte spredningsplott der en kombinerer to sett med verdier.

For mer detaljer anbefales det å bruke kommandoen `help ivregress-predict`

5.5.5 Grafisk visning av koeffisientestimater

Koeffisientestimater kan også vises grafisk. Dette gjør det mer oversiktlig dersom modellen inneholder mange forklaringsvariabler eller faktorledd.

Eksempel på syntax:

```
coefplot ivregress lønn mann (formue = alder)
```

For mer informasjon om kommandoen `coefplot`, se kapittel 5.4.5.

5.6 Oaxaca - ordinær lineær regresjon med dekomponering av gruppespesifikke effekter

Kommandoen `oaxaca` er et verktøy til å måle om det er systematiske forskjeller mellom to grupper, f.eks. menn og kvinner, og forskjellene blir videre dekomponert i en forklart og en uforklart komponent.

Kommandoen utfører en Blinder-Oaxaca dekomponering[8] som brukes til å forklare forskjeller i den avhengige variabelens gjennomsnittverdi for to grupper. Forskjellen/differansen dekomponeres til to komponenter: Forklart differanse («between group») og uforklart effekt (koeffisienteffekt). I likhet med kommandoen `regress`, brukes kontinuerlige avhengige variabler som f.eks. lønn. Forskjellen er at man spesifiserer de to gruppene gjennom by-variabelen når man bruker `oaxaca`.

By-variabelen som brukes til å gruppere må være kategorisk, men kan ha både numerisk og alfanumerisk verdiformat. Verdien som rangeres først (numerisk eller alfabetisk) knyttes til gruppe 1. Om variabelen inneholder mer enn to verdier, brukes de to verdiene som rangeres først, mens de andre holdes utenfor analysen.

Standard-løsningen som brukes er «three-fold», og man får ut hovedtallene:

- Differansen i gjennomsnittsverdi for den avhengige variabelen målt for hver av de to gruppene: mean(gruppe1) – mean(gruppe2)
- Dekomponert differanse: Forklart, uforklart og samtidig effekt
- Antall enheter tilhørende de to respektive gruppene, samt hvilke verdikoder som benyttes

Ved å bruke opsjonen `pool`, vil systemet bruke en såkalt «two-fold pooled» tilnærming der dekomponeringen bruker det samlede gjennomsnittet som referanseverdi (samtidig effekt rapporteres ikke ved denne tilnærmingen).

Det mest vanlige bruksområdet er å analysere systematiske forskjeller i økonomiske variabler som lønn, og sammenlikne menn mot kvinner. Men også andre typer grupperinger kan benyttes.

Eksempel på bruk av `oaxaca`:

---

[8] Metoden baserer seg på prinsippene beskrevet i Ben Janns Stata Journal-artikkel (2008): https://www.stata-journal.com/sjpdf.html?articlenum=st0151. Python-implementasjonen som benyttes i microdata.no beskrives her: https://github.com/statsmodels/statsmodels/blob/main/statsmodels/stats/oaxaca.py.

```
demografidata» summarize inntø5 if kjonn == '1'
```

Variabel	Gj.snitt	Std.avvik	Antall	1%	25%	50%	75%	99%
innt05	344049.472	233042.9399	1307221	-400	199916	332449	439539.25	1298943

```
demografidata» summarize inntø5 if kjonn == '2'
```

Variabel	Gj.snitt	Std.avvik	Antall	1%	25%	50%	75%	99%
innt05	230558.5825	143493.2302	1182715	1277	123842	236013	317681	711650

```
demografidata» oaxaca inntø5 gift alder formuehøy by kjonn
```

Forskjell i mean difference (Første gruppe - Andre gruppe): 122404.32107998701

Modell	kjonn	Antall observasjoner
Første gruppe	1 - Mann	1307226
Andre gruppe	2 - Kvinne	1182717

Gruppeforskjell	Effekt	Standardfeil
Forklart	10205.5734	381.211
Uforklart	105704.0872	340.0603
Samtidig effekt	6494.6605	546.3228

```
demografidata» oaxaca inntø5 gift alder formuehøy by kjonn, pool
```

Forskjell i mean difference (Første gruppe - Andre gruppe): 122404.32107998701

Modell	kjonn	Antall observasjoner
Første gruppe	1 - Mann	1307226
Andre gruppe	2 - Kvinne	1182717

Gruppeforskjell	Effekt	Standardfeil
Forklart	16047.2441	276.6971
Uforklart	106357.077	440.6838

Merk at differansen i gjennomsnittsverdi (mean difference) som rapporteres av `oaxaca` avviker litt fra differansen man finner gjennom å bruke kommandoen `summarize` på den avhengige variabelen for hver av de to gruppene. Årsaken er at deskriptiv statistikk som genereres gjennom kommandoer som `summarize` er gjenstand for winsorisering (høyre- og venstresensur). Regresjonsresultater fra kommandoer som `oaxaca` blir derimot ikke winsorisert, og viser den korrekte differansen.

For mer om winsorisering og andre personvern-mekanismer, se vedlegg C.

5.7 Logit og probit - logistisk regresjonsanalyse

Logistisk regresjonsanalyse brukes til å beregne sannsynligheten for «suksess» (én tilstand foran en annen) eller for å havne i en av flere mulige tilstander.

Syntax:

```
logit <variabel> <variabelliste> [if <betingelse>] [, <opsjoner>]
probit <variabel> <variabelliste> [if <betingelse>] [, <opsjoner>]
```

Den avhengige variabelen må angis først, etterfulgt av forklaringsvariablene. Opsjoner kan benyttes for ulike formål, som f.eks. robust- eller cluster-estimering, jfr. underkapitlene nedenfor. I likhet med andre statistiske kommandoer, kan også regresjonskommandoer kombineres med en if-betingelse for å kjøre regresjoner på utvalgte grupper. For full oversikt over muligheter, bruk kommandoen `help logit` eller `help probit`.

Kommandoene `logit` og `probit` brukes til å utføre en logistisk analyse der den avhengige variabelen er en kategorisk variabel med 2 mulige utfall (dummyvariabel). Eksempler kan være jobb/ikke jobb, alderspensjonist/ikke pensjonert etc. Logit-modeller antar at sannsynligheten for «suksess» følger en logaritmisk (log) fordeling, mens probit-varianten antar en normalfordeling. De to fordelingene er tilnærmet like, og resultatene vil derfor bli tilnærmet like. Logit er imidlertid den mest brukte modellen, og det er den vi fokuserer på i eksemplene nedenfor.

Resultatet av kommandoen `logit` gir en tabell med standardverdier som koeffisienter, standardavvik, z-verdier, p-verdier og konfidensintervall. Tallene inni hovedtabellen knyttes til de ulike variablene, mens tallene øverst viser til analysemodellen som helhet (gir pekepinn på modellens kvalitet/forklaringskraft).

Eksempel:

```
»logit høyinnt mann gift alder formuehøy
   Antall iter: 8              LR chi2(5): 7.6908e+5
     Log sans:                 Prob > chi2: 0
 -1.532013e+6                   Pseudo R2: 0.2006
    Antall obs: 7241907
```

høyinnt	Coef.	Std. Avvik	z	P>|z|	[% Konf.	Intervall]
alder	-0.0454	0.0001	-418.84	0	-0.0456	-0.0452
formuehøy	1.3824	0.0043	315.11	0	1.3738	1.391
gift	1.8289	0.0036	506.76	0	1.8219	1.836
mann	1.2097	0.0036	329.65	0	1.2025	1.2169
Konst	-2.1892	0.0048	-449.09	0	-2.1988	-2.1796

I eksempelet over er den avhengige variabelen `høyinnt` kodet på følgende måte:

```
generate høyinnt = 0
replace høyinnt = 1 if innt05 > 400000
```

I likhet med ordinær lineær regresjonsanalyse (se kapittel 5.4) er det noen tall som er viktigere å se på enn andre. P-verdien *Prob > chi2* angir hvor god modellen er, dvs. den angir hvor stor forklaringskraft summen av alle variablene i modellen har. Jo nærmere 0 dess bedre, og verdier bør være under 0.05.

Pseudo R2 er en variant av *justert* R^2 (rapporteres ved ordinære lineære regresjonsanalyser) som angir hvor stor andel av variansen i responsvariabelen som blir forklart av de uavhengige variablene (skala fra 0 til 1 der en søker høyest mulig verdi). Dette samlemålet må imidlertid tolkes med stor grad av varsomhet ettersom den i mange tilfeller angir en verdi som enten er kunstig høy eller lav. *Prob > chi2* er derfor å anbefale for logistiske regresjonsmodeller.

Variablenes p-verdi $P > |z|$ tilsvarer $P > |t|$ i ordinær lineær regresjonsanalyse. Grenseverdien er også her 0.05 om en opererer med et signifikansnivå på 5% (noe de fleste bruker). Rapporterte verdier under dette gjør at en kan konkludere med at tilhørende variabel er signifikant på et 5% nivå.

Det er det samme om en ser på *z*-verdi eller tilhørende p-verdi. Verdien *z* er en standardisert versjon av coeffisientverdien, som har forventning lik 0 og der verdier som overstiger +/- 1.96 impliserer at den aktuelle variabelen som koeffisienten tilhører har en signifikant påvirkning på sannsynligheten for «suksess». Positive verdier angir positiv effekt, og vice versa.

Konfidensintervallet gitt ved de to kolonnene lengst til høyre kan tolkes på samme måte som for ordinær lineær regresjonsanalyse, dvs. om det inkluderer verdien 0 tyder dette på null signifikans.

En ser av eksempelet ovenfor at alle forklaringsvariablene er signifikante med god margin (har høye z-verdier). “Alder” har en negativ effekt på sannsynligheten for å havne i en høyinntektsgruppe, mens de andre variablene har en tilsvarende positiv effekt. Videre ser en at modellens P-verdi er lik 0, noe som viser at vi har en god forklaringsmodell.

5.7.1 Faktorvariabler

Se kapittel 5.4.1 for informasjon om hvordan faktorvariabler kan brukes. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.7.2 Marginaleffekter

Opsjonen `mfx()` brukes til å angi at marginaleffekter skal estimeres i tillegg til de vanlige logistiske koeffisientene. Dette foretrekkes av mange siden marginaleffekter er lettere å tolke enn standard-estimatene.

Det er mulig å velge mellom fire forskjellige typer marginaleffekter:

- `dydx`: marginaleffekt = d(y) / d(x)
- `eyex`: elastisitetsverdi = d(ln(y)) / d(ln(x))
- `dyex`: semielastisitet = d(y) / d(ln(x))
- `eydx`: semielastisitet = d(ln(y)) / d(x)

Ved å kombinere opsjonene `mfx()` og `mfx_at()`, kan man overstyre standardmålet. Følgende varianter er tilgjengelig:

- `mfx_at(overall)` (snittverdien av marginaleffektene målt over alle verdier av x) (standardmål dersom opsjonen utelates)
- `mfx_at(mean)` (marginaleffekt målt ved snittet av x)
- `mfx_at(median)` (marginaleffekt målt ved median av x)
- `mfx_at(zero)` (marginaleffekt målt ved 0-verdien for x)

Opsjonen `mfx_at()` brukes vanligvis i kombinasjon med `mfx()`, for eksempel:

```
logit høyinnt mann gift alder formuehøy, mfx(dydx) mfx_at(mean)
```

Men man kan også bare bruke `mfx_at()`. Da benyttes standardvarianten `mfx(dydx)`.

Følgende alternative regresjonsuttrykk vil presentere de samme marginaleffekt-verdiene:

```
logit høyinnt mann gift alder formuehøy, mfx(dydx) mfx_at(overall)
logit høyinnt mann gift alder formuehøy, mfx_at(overall)
logit høyinnt mann gift alder formuehøy, mfx(dydx)
```

Eksempel der man kjører en logit-regresjon og estimerer gjennomsnittlige marginaleffekter (vanligst å bruke). Vanlige logistiske estimater listes først, deretter marginaleffektene:

```
demografidata» logit høyinnt mann gift alder innvandrer ettbarn flerebarn høyutd oslo ledig formuehøy, mfx(dydx)
```

Antall iter: 8 LR chi2(11): 33066.7
Log sans: -85882.3 Prob > chi2: 0
Antall obs: 667309 Pseudo R2: 0.16143

høyinnt	Coef.	Std.feil	z	P>|z|	[95% Konf.	intervall]
mann	0.98596	0.01477	66.7287	0	0.957	1.01492
gift	0.57347	0.01664	34.4628	0	0.54086	0.60609
alder	0.02396	0.00063	37.6186	0	0.02271	0.02521
innvandrer	-0.42407	0.01834	-23.1177	0	-0.46003	-0.38812
ettbarn	0.41667	0.01973	21.1096	0	0.37798	0.45535
flerebarn	0.64685	0.01867	34.6332	0	0.61025	0.68346
høyutd	1.54034	0.01902	80.9701	0	1.50306	1.57763
oslo	0.14832	0.01858	7.97929	0	0.11189	0.18476
ledig	0.24728	0.02347	10.5358	0	0.20128	0.29328
formuehøy	1.42484	0.01573	90.5629	0	1.394	1.45567
Konst	-5.65992	0.02941	-192.447	0	-5.71756	-5.60227

dydx av høyinnt

	Margin	Std.feil	z	P>|z|	[% Konf.	intervall]
mann	0.03052	0.00047	64.2524	0	0.02959	0.03145
gift	0.01775	0.00052	34.1367	0	0.01673	0.01877
alder	0.00074	0.00002	37.0107	0	0.0007	0.00078
innvandrer	-0.01313	0.00057	-23.0486	0	-0.01424	-0.01201
ettbarn	0.0129	0.00061	21.0301	0	0.01169	0.0141
flerebarn	0.02002	0.00058	34.2954	0	0.01888	0.02117
høyutd	0.04769	0.00061	78.0678	0	0.04649	0.04888
oslo	0.00459	0.00057	7.97561	0	0.00346	0.00572
ledig	0.00765	0.00072	10.5239	0	0.00623	0.00908
formuehøy	0.04411	0.00051	84.995	0	0.04309	0.04513

5.7.3 Cluster- og robust-estimering

Se kapittel 5.4.3 for informasjon om hvordan benytte cluster- eller robust-estimering. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.7.4 Prediksjonsverdier og residualverdier

Alle regresjonsvarianter som finnes i microdata.no har tilknyttede kommandoer som genererer blant annet residual- og prediksjonsverdier. Dette er verdier som kan brukes til å analysere dataspredningen og for testing av regresjonsmodeller. Prediksjonsverdier kan dessuten brukes som input til videre analyser.

Kommandoene har samme navn som tilhørende regresjonskommando pluss "`-predict`"

Syntax:

```
logit-predict <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]

probit-predict <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Variablene angis på samme måte som for den tilhørende regresjons-modellen som kjøres med kommandoen `logit` evt. `probit`.

Følgende verdier kan hentes ut:

- `logit-predict`: Sannsynlighetsverdier, prediksjonsverdier og residualer
- `probit-predict`: Sannsynlighetsverdier og prediksjonsverdier

En bestemmer selv hvilke verdier en vil generere gjennom bruk av opsjoner. Resultatet av kjøringene er et sett med variabler som inneholder de ulike verdiene. Som standard genereres førstnevnte verditype i listen over, men det anbefales likevel å spesifisere dette gjennom opsjoner ettersom en da også kan bestemme navn på de genererte variablene inni en parentes som vist i syntax-eksempelet nedenfor. Om en kjører flere “predictkommandoer”, må en lage nye navn for de automatisk genererte variablene.

Syntax-eksempel:

```
logit-predict høylønn alder mann formue, residuals(res4)
predicted(pred4) probabilities(prob4)
```

De automatisk genererte variablene kan brukes som input til videre analyser eller til å vises grafisk. Aktuelle grafiske kommandoer er `hexbin` og `histogram`. Ved å kjøre `histogram` på residualvariabelen, kan en sjekke hvorvidt residualene er normalfordelte. Hexbin-kommandoen kan dessuten brukes til lage anonymiserte spredningsplott der en kombinerer to sett med verdier.

For mer detaljer anbefales det å bruke kommandoen `help logit-predict` eller `help probit-predict`

5.7.5 Grafisk visning av koeffisientestimater

Koeffisientestimater kan også vises grafisk. Dette gjør det mer oversiktlig dersom modellen inneholder mange forklaringsvariabler eller faktorledd.

Eksempel på syntax:

```
coefplot logit høylønn i.alder mann høy_formue
```

For mer informasjon om kommandoen `coefplot`, se kapittel 5.4.5.

5.8 Mlogit - multinomisk logistisk regresjonsanalyse

En kan benytte logistiske modeller med flere enn to mulige utfall for responsvariabelen, dvs. multinomiske logit-modeller, gjennom følgende kommando:

```
mlogit <variabel> <variabelliste> [if <betingelse>] [, <opsjoner>]
```

Den avhengige variabelen må angis først, etterfulgt av forklaringsvariablene. Opsjoner kan benyttes for ulike formål, som f.eks. robust- eller cluster-estimering, jfr. underkapitlene nedenfor. I likhet med andre statistiske kommandoer, kan også regresjonskommandoer kombineres med en if-betingelse for å kjøre regresjoner på utvalgte grupper. For full oversikt over muligheter, bruk kommandoen `help mlogit`.

I det rapporterte resultatet vil hovedtabellen bli mer omfattende enn for vanlige (binomiske) logistiske modeller. En får et sett med koeffisienter, standardfeil, z-verdier etc for hvert mulige utfall minus referanseutfallet. Ved f.eks. 3 utfall får en altså 2 sett med verdier. Ved tolkninger av disse så sammenlikner en med sannsynligheten for å havne i referanseutfallet. Betydningen av de ulike rapporterte målene i tabellen gjennomgås i kapittel 5.7.

Eksempel:

```
»mlogit inntgr mann gift alder formuehøy
  Antall iter: 8             LR chi2(10): 2.027e+6
     Log sans:               Prob > chi2: 0
 -3.9626159e+6                 Pseudo R2: 0.2036
   Antall obs: 7241907
```

inntgr		Coef.	Std. Avvik	z	P>|z|	[% Konf.	Intervall]
2	alder	-0.0561	0	-712.67	0	-0.056	-0.056
	formuehøy	0.7157	0.005	156.58	0	0.707	0.725
	gift	1.8572	0.003	674.68	0	1.852	1.863
	mann	-0.2324	0.002	-101.07	0	-0.237	-0.228
	Konst	0.4212	0.003	134.66	0	0.415	0.427
3	alder	-0.0586	0	-495.28	0	-0.059	-0.058
	formuehøy	1.5778	0.005	334.31	0	1.569	1.587
	gift	2.3359	0.004	601.74	0	2.328	2.343
	mann	1.1127	0.004	298.24	0	1.105	1.12
	Konst	-1.4676	0.005	-290.81	0	-1.478	-1.458

I eksempelet over er den avhengige variabelen `inntgr` kodet på følgende måte:

```
generate inntgr = 1
replace inntgr = 2 if innt05 > 200000
replace inntgr = 3 if innt05 > 400000
```

5.8.1 Faktorvariabler

Se kapittel 5.4.1 for informasjon om hvordan faktorvariabler kan brukes. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.8.2 Marginaleffekter

Se kapittel 5.7.2 for informasjon om hvordan marginaleffekter estimeres. Fremgangsmåten er den samme som for binære logistiske modeller.

5.8.3 Cluster- og robust-estimering

Se kapittel 5.4.3 for informasjon om hvordan benytte cluster- eller robust-estimering. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.8.4 Prediksjonsverdier og residualverdier

Alle regresjonsvarianter som finnes i microdata.no har tilknyttede kommandoer som genererer blant annet residual- og prediksjonsverdier. Dette er verdier som kan brukes til å analysere dataspredningen og for testing av regresjonsmodeller. Prediksjonsverdier kan dessuten brukes som input til videre analyser.

Kommandoene har samme navn som tilhørende regresjonskommando pluss "`-predict`"

Syntax:

```
mlogit-predict <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Variablene angis på samme måte som for den tilhørende regresjons-modellen som kjøres med kommandoen `mlogit`.

Følgende verdier kan hentes ut: Sannsynlighetsverdier og prediksjonsverdier

En bestemmer selv hvilke verdier en vil generere gjennom bruk av opsjoner. Resultatet av kjøringene er et sett med variabler som inneholder de ulike verdiene. Som standard genereres førstnevnte verditype, men det anbefales likevel å spesifisere dette gjennom opsjoner ettersom en da også kan bestemme navn på de genererte variablene inni en parentes som vist i syntax-eksempelet nedenfor. Om en kjører flere “predictkommandoer”, må en lage nye navn for de automatisk genererte variablene.

Syntax-eksempel:

```
mlogit-predict lønnkat alder mann høyformue, predicted(pred6)
probabilities(prob6)
```

De automatisk genererte variablene kan brukes som input til videre analyser eller til å vises grafisk. Aktuelle grafiske kommandoer er `hexbin` og `histogram`. Ved å kjøre `histogram` på residualvariabelen, kan en sjekke hvorvidt residualene er normalfordelte. Hexbin-kommandoen kan dessuten brukes til lage anonymiserte spredningsplott der en kombinerer to sett med verdier.

For mer detaljer anbefales det å bruke kommandoen `help mlogit-predict`

5.8.5 Grafisk visning av koeffisientestimater

Koeffisientestimater kan også vises grafisk. Dette gjør det mer oversiktlig dersom modellen inneholder mange forklaringsvariabler eller faktorledd.

Eksempel på syntax:

```
coefplot mlogit lønnkat alder mann høy_formue
```

For mer informasjon om kommandoen `coefplot`, se kapittel 5.4.5.

5.9 Regress-panel - paneldata-analyse

Paneldata er datasett der hver enhet har oppgitt verdier for samtlige variabler målt over et gitt antall måletidspunkt. Dette har den fordelen at en kan ta med tidskomponenten i analyser, og at en får mye større datagrunnlag og gjerne analyser av en bedre kvalitet.

Syntax:

```
regress-panel <variabel> <variabelliste> [if <betingelse>] [,
<opsjoner>]
```

Den avhengige variabelen må angis først, etterfulgt av forklaringsvariablene. Opsjoner kan benyttes for ulike formål, som f.eks. robust- eller cluster-estimering, jfr. underkapitlene nedenfor. I likhet med andre statistiske kommandoer, kan også regresjonskommandoer kombineres med en if-betingelse for å kjøre regresjoner på utvalgte grupper. For full oversikt over muligheter, bruk kommandoen `help regress-panel`.

Se kapittel 2.4 for hvordan en oppretter datasett for paneldata-analyse. Der finner en også et skript-eksempel.

Det finnes et stort batteri av paneldataanalyser som kan tas i bruk, avhengig av hvilke antakelser som gjøres om de ulike variablenes variasjon over tid. Vanlige varianter som brukes er “fixed effect”- og “random effect”-analyser.

I eksempelet nedenfor brukes årslønn (årlig lønnsinntekt) som avhengig variabel, og dummyvariabler for hhv. sivilstatus=gift og bosted=oslo brukes som forklaringsvariabler. I tillegg er 5 måletidspunkter benyttet: 31/12 i årene 2011-2015. Populasjon = alle personer som fullførte et masterstudium i løpet av høstsemesteret 2010.

Eksempel 1: Panel-regresjon med fixed effects

```
paneldata2» regress-panel årslønn gift oslo, fe
```

Antall Obs:	20225	R^2 i:	0.03406
Antall grupper:	4247	R^2 mellom:	-0.00656
Min obs/grp:	1	R^2 total:	0.00487
Snitt obs/grp:	4.76218	Corr(u_i, Xb):	-0.11256
Maks obs/grp:	5		
F(2,15976):	281.69067574	Sigma u:	206600.87339816123
Prob > F:	1.11022e-16	Sigma e:	126287.16304855184
		Rho:	0.72799

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	1.0535662e+5	4609.16	22.858	0	1.0506759e+5	1.0564565e+5
oslo	36284.5	4911.99	7.38693	1.57651e-13	35976.5	36592.5
Konst	4.6886852e+5	2598.49	180.438	0	4.6870557e+5	4.6903147e+5

Eksempel 2: Panel-regresjon med random effects (samme datasett som eksempel 1)

```
paneldata2» regress-panel årslønn gift oslo, re
```

Antall Obs:	20225	R² i:	0.0333
Antall grupper:	4247	R² mellom:	0.00315
Min obs/grp:	1	R² total:	0.00896
Snitt obs/grp:	4.76218		
Maks obs/grp:	5	Sigma u:	196036.29557757667
F(2,20222):	96.71945325	Sigma e:	126287.16304855184
Prob > F:	1.11022e-16	Rho:	0.70671

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	91013.1	3913.31	23.2573	0	90767.7	91258.5
oslo	27222.5	4026.04	6.76161	1.40187e-11	26970	27475
Konst	4.6809574e+5	3756.31	124.615	0	4.6786019e+5	4.6833129e+5

I tillegg til regresjonsanalyser, er det mulig å gjøre seg kjent med paneldatasett gjennom ulike deskriptive verktøy:

- `tabulate-panel` tilsvarer kommandoen `tabulate` for vanlige datasett, jfr. kapittel 4.1, men viser verdier for alle måletidspunkt. Opsjoner for prosentuering kan brukes i likhet med `tabulate`. Spesifiseres flere variabler, vises det flerdimensjonale krysstabeller for de aktuelle variabler

- `summarize-panel` tilsvarer kommandoen `summarize` for vanlige datasett, jfr. kapittel 4.2, men viser verdier for alle måletidspunkt. Verdier vises vertikalt og ikke horisontalt, og en må holde musepeker over tallene for å vise deres betydning

- `transitions-panel` viser to-veis frekvens/sannsynlighet for overganger mellom alle kombinasjoner av kategoriske verdier over tid (overgangssannsynligheter), for en gitt variabel. Forspalten representerer utgangsverdiene, mens tabellhodet representerer overgangsverdien. Spesifiseres flere variabler, vises toveis overgangstabeller for hver variabel i respektive tabeller. Overganger representeres som standard gjennom frekvenser og prosenter (rekkevis). Overganger enten fra eller til manglende verdi (sysmiss) holdes utenfor tabuleringen.

Eksempel 3: Tabulate-panel for hhv. gift og oslo (samme datasett som eksempel 1 og 2)

```
paneldata2» tabulate-panel gift
```

		date@panel					
		2011-12-31	2012-12-31	2013-12-31	2014-12-31	2015-12-31	*Total*
gift	0	3517	3398	3237	3047	2908	*16110*
	1	1083	1208	1374	1556	1691	*6900*
Total		*4601*	*4604*	*4606*	*4598*	*4600*	*23005*

```
paneldata2» tabulate-panel oslo
```

		date@panel					
		2011-12-31	2012-12-31	2013-12-31	2014-12-31	2015-12-31	*Total*
oslo	0	3052	2993	2977	3010	3053	*15073*
	1	1551	1611	1623	1597	1550	*7936*
Total		*4601*	*4604*	*4606*	*4598*	*4600*	*23005*

Eksempel 4: Summarize-panel for den avhengige variabelen årslønn (samme datasett)

```
paneldata2» summarize-panel årslønn
```

date@panel	2011-12-31	408852.72 196058.35 4047 6066 1103085 303401 413573 497759.75
	2012-12-31	484347.39 198684.67 4065 13348 1202311 400655 474860 569977.25
	2013-12-31	527803.61 213332.71 4041 22857 1304797 431857.25 513947 617650.5
	2014-12-31	567728.92 235573.16 4043 21912 1462289 452964.25 546906 665830.75
	2015-12-31	596611.39 247937.55 4026 15000 1572028 474567.5 571551 701034.5
Total		*516957.13* *228922.05* *20227* *6066* *1572028* *406669* *501847* *620650*

Eksempel 5: Transitions-panel (overgangsrater mellom kombinasjoner av kategoriske verdier) for variablene oslo og gift (samme datasett)

```
paneldata2» transitions-panel oslo gift
```

oslo

	0	1	*Total*
0	11567 96.21	455 3.784	*12022* *100*
1	460 7.203	5926 92.79	*6386* *100*
Total	*12027* *65.33*	*6381* *34.66*	*18408* *100*

gift

	0	1	*Total*
0	12474 94.48	728 5.514	*13202* *100*
1	120 2.305	5086 97.69	*5206* *100*
Total	*12594* *68.41*	*5814* *31.58*	*18408* *100*

Kommentar til tabell i eksempel 5:
I 96.21% av tilfellene vil personer som ikke er bosatt i Oslo ha samme tilstand året etter (neste måling). Resten, 3.78%, vil flytte til Oslo. Blant dem i populasjonen som bor i Oslo i et gitt tidspunkt, vil 7.2% flytte ut av Oslo mens 92.8% blir værende året etter (neste måling).

Samme prinsipp gjelder for variabelen *gift*: Her ser vi at 5.5% endrer status fra ikke-gift til gift fra ett år til et annet (neste måling) over den totale måleperioden. 2.3% endrer status fra gift til ikke-gift.

5.9.1 Faktorvariabler

Se kapittel 5.4.1 for informasjon om hvordan faktorvariabler kan brukes. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.9.2 Modelldiagnostikk

Det er mulig å foreta modelltesting for å sjekke om en skal benytte fixed eller random effects-estimering i forbindelse med panelregresjoner. Dette gjøres ved å bruke kommandoen `hausman`.

Syntax og input følger samme logikk som tilhørende regresjonskommando (`regress-panel`): Avhengig variabel brukes som første input, og deretter listes de uavhengige.

Eksempel:

```
regress-panel lønn alder høyutdanning formue oslo
hausman lønn alder høyutdanning formue oslo
```

Resultatet av hausman-kjøringen er standard regresjonsresultat for hhv. fixed og random effect-estimering. I tillegg vises også differansen mellom koeffisientene ved de alternative estimeringer, samt et samlemål som indikerer hvilken variant som er best å benytte for det gjeldende datasett: P-verdi basert på kjikvadratdiagnostikk.

Eksempel der hausman-testen brukes til å sammenlikne estimater basert på fixed vs. random effects. Paneldataanalysen som skal testes estimerer effekten av å være hhv. gift og bosatt i Oslo på årslønn målt over årene 2016-2019:

```
paneldata» hausman årslønn gift oslo
```

Fixed effects

Antall Obs:	22454	R² i:	0.02933
Antall grupper:	6711	R² mellom:	0.03491
Min obs/grp:	0	R² total:	0.03388
Snitt obs/grp:	3.34585	Corr(u_i, Xb):	-0.00938
Maks obs/grp:	4		
F(2,16494):	249.245423	Sigma u:	239972.879944
Prob > F:	0	Sigma e:	146261.179896
		Rho:	0.72914

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	1.0026258e+5	6440.21	15.5682	0	87639	1.1288609e+5
oslo	79669.2	4926.32	16.1721	0	70013.1	89325.3
Konst	4.8489385e+5	2403.56	201.739	0	4.8018261e+5	4.896051e+5

Random effects

Antall Obs:	22454	R² i:	0.02851
Antall grupper:	6711	R² mellom:	0.04397
Min obs/grp:	0	R² total:	0.03712
Snitt obs/grp:	3.34585		
Maks obs/grp:	4	Sigma u:	225627.095599
F(2,22451):	97.17749	Sigma e:	146261.179896
Prob > F:	0	Rho:	0.70411

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	1.1688691e+5	5071.49	23.0478	0	1.0694642e+5	1.268274e+5
oslo	66497.3	4063.61	16.3641	0	58532.4	74462.3
Konst	4.74476e+5	3574.12	132.753	0	4.6747047e+5	4.8148153e+5

Hausman

	b_0	b_1	$b_0 - b_1$	Std.feil
gift	1.0026258e+5	1.1688691e+5	-16624.3	3969.41
oslo	79669.2	66497.3	13171.8	2784.91

chi2(2): 36.813255
Prob > chi2: 0

P-verdier < 0.05 indikerer at det er systematiske forskjeller i koeffisientestimatene og at fixed effect-modellering passer dataene best. P-verdier over denne grensen indikerer det motsatte (at random effect-modellering bør brukes).

For mer detaljer anbefales det å bruke help-kommandoen: `help hausman`

Kommandoen `regress-panel-predict` kan også benyttes som hjelpeverktøy for modelldiagnostikk, jfr. kapittel 5.9.4.

5.9.3 Cluster- og robust-estimering

Se kapittel 5.4.3 for informasjon om hvordan benytte cluster- eller robust-estimering. Fremgangsmåten er den samme som for ordinær lineær regresjon.

5.9.4 Prediksjonsverdier og residualverdier

Alle regresjonsvarianter som finnes i microdata.no har tilknyttede kommandoer som genererer blant annet residual- og prediksjonsverdier. Dette er verdier som kan brukes til å analysere dataspredningen og for testing av regresjonsmodeller. Prediksjonsverdier kan dessuten brukes som input til videre analyser.

Kommandoene har samme navn som tilhørende regresjonskommando pluss “`-predict`”

Syntax:

```
regress-panel-predict <variabel> <variabelliste> [if
<betingelse>] [, <opsjoner>]
```

Variablene angis på samme måte som for den tilhørende regresjons-modellen som kjøres med kommandoen `regress-panel`.

Følgende verdier kan hentes ut: Prediksjonsverdier, residualer og enhetseffekter

En bestemmer selv hvilke verdier en vil generere gjennom bruk av opsjoner. Resultatet av kjøringene er et sett med variabler som inneholder de ulike verdiene. Som standard genereres førstnevnte verditype, men det anbefales likevel å spesifisere dette gjennom opsjoner ettersom en da også kan bestemme navn på de genererte variablene inni en parentes som vist i syntax-eksempelet nedenfor. Om en kjører flere “predictkommandoer”, må en lage nye navn for de automatisk genererte variablene.

Syntax-eksempel:

```
regress-panel-predict lønn mann alder formue, predicted(ppred1)
residuals(pres1) effects(peff1)
```

De automatisk genererte variablene kan brukes som input til videre analyser eller til å vises grafisk. Aktuelle grafiske kommandoer er `hexbin` og `histogram`. Ved å kjøre `histogram` på residualvariabelen, kan en sjekke hvorvidt residualene er normalfordelte. Hexbin-kommandoen kan dessuten brukes til lage anonymiserte spredningsplott der en kombinerer to sett med verdier.

For mer detaljer anbefales det å bruke kommandoen `help regress-panel-predict`

5.9.5 Grafisk visning av koeffisientestimater

Koeffisientestimater kan også vises grafisk. Dette gjør det mer oversiktlig dersom modellen inneholder mange forklaringsvariabler eller faktorledd.

Eksempel på syntax:

```
coefplot regress-panel lønn alder mann høy_formue, re
```

For mer informasjon om kommandoen `coefplot`, se kapittel 5.4.5.

5.10 Eksempel

```
textblock
Foreta regresjonsanalyser og hente ut prediksjons- og residualverdier
-----------------------------------------------

Dette eksempelet viser hvordan man benytter de ulike regresjonskommandoene, inkludert uthenting
av prediksjons- og residualverdier. Særlig histogram er en veldig nyttig kommando som kan brukes til å
studere visuelt i hvilken grad residualene er normalfordelte. Men en kan i prinsippet benytte seg av
alle tilgjengelige og relevante kommandoer for videre analyser.
endblock

require no.ssb.fdb:12 as db

create-dataset regresjonsdata
import db/INNTEKT_WLONN 2019-12-31 as lønn
import db/INNTEKT_BER_BRFORM 2019-12-31 as formue
import db/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import db/BEFOLKNING_KJOENN as kjønn
import db/BEFOLKNING_STATUSKODE 2020-01-01 as bosattstatus

keep if bosattstatus == '1'
```

```
generate alder = 2019 - int(faarmnd/100)

generate mann = 0
replace mann = 1 if kjønn == '1'

//regress
regress lønn alder mann formue
regress-predict lønn alder mann formue
histogram predicted
hexbin predicted lønn
regress-predict lønn alder mann formue, residuals(res) predicted(pred) cooksd(cook)
regress-predict lønn alder mann, residuals(res2) predicted(pred2) cooksd(cook2)
histogram pred
histogram res
histogram cook
histogram res2

//ivregress
ivregress lønn mann (formue = alder)
ivregress-predict lønn mann (formue = alder), residuals(res3) predicted(pred3)
histogram pred3
histogram res3

//logit
summarize lønn formue
histogram lønn
histogram formue
generate høylønn = 0
replace høylønn = 1 if lønn > 800000
generate høyformue = 0
replace høyformue = 1 if formue > 4000000

logit høylønn alder mann høyformue
logit-predict høylønn alder mann høyformue, residuals(res4) predicted(pred4) probabilities(prob4)
histogram pred4
histogram res4
histogram prob4


//probit
probit høylønn alder mann høyformue
```

```
probit-predict høylønn alder mann høyformue, predicted(pred5) probabilities(prob5)
histogram pred5
histogram prob5

//mlogit
generate lønnkat = 0
replace lønnkat = 1 if lønn > 0
replace lønnkat = 2 if lønn > 800000

mlogit lønnkat alder mann høyformue
mlogit-predict lønnkat alder mann høyformue, predicted(pred6) probabilities(prob6)
summarize pred6_1
histogram pred6_2
histogram prob6_1
histogram prob6_2


//regress-panel
sample 0.05 54321

clone-units regresjonsdata paneldata
use paneldata
import-panel db/INNTEKT_WLONN db/BEFOLKNING_FOEDSELS_AAR_MND db/BEFOLKNING_KJOENN
db/INNTEKT_BER_BRFORM 2017-12-31 2018-12-31 2019-12-31
rename INNTEKT_WLONN lønn
rename INNTEKT_BER_BRFORM formue
generate alder = 2019 - int(BEFOLKNING_FOEDSELS_AAR_MND/100)
generate mann = 0
replace mann = 1 if BEFOLKNING_KJOENN == '1'

regress-panel lønn mann alder formue
regress-panel lønn mann alder formue, re
regress-panel-predict lønn mann alder formue, predicted(ppred1) residuals(pres1) effects(peff1)
regress-panel-predict lønn mann alder formue, re predicted(ppred2) residuals(pres2) effects(peff2)
histogram ppred1
histogram pres1
histogram peff1
histogram ppred2
histogram pres2
histogram peff2
hausman lønn mann alder formue
```

Vedlegg A: Oversikt over kommandoer

Kommando	Formål	Type funksjonalitet
`clear`	Tøm kommandoøkten	Støttekommando
`edit`	Lagre en kommandoøkt som et skript, og overskriv eksisterende/aktive skript	Support command
`help`	Hjelp	Støttekommando
`help-function`	Hjelp - funksjoner	Støttekommando
`history`	Kommandohistorikk	Støttekommando
`load`	Laste inn et skript som en kommandolinjeøkt	Støttekommando
`save`	Lagre en kommandoøkt som et skript	Støttekommando
`variables`	Vise metadata for registervariable	Støttekommando
`clone-dataset`	Duplisere et datasett	Datasett-kommando
`clone-units`	Duplisere en datapopulasjon	Datasett-kommando
`create-dataset`	Lage nytt, tomt, navngitt datasett	Datasett-kommando
`delete-dataset`	Slette datasett	Datasett-kommando
`rename-dataset`	Endring av datasettnavn	Datasett-kommando
`reshape-from-panel`	Restrukturer datasett fra long-/panel-format til wide-format	Datasett-kommando
`reshape-to-panel`	Restrukturer datasett fra wide-format til long-/ panel-format	Datasett-kommando
`require`	Koble til databank	Datasett-kommando
`use`	Bruk et navngitt datasett	Datasett-kommando
`assign-labels`	Sette labler på variabler og kategorier	Tilretteleggingskommando
`clone-variables`	Duplisere variabler	Tilretteleggingskommando
`collapse`	Aggregering av datasett	Tilretteleggingskommando
`define-labels`	Definere lister av value/labels	Tilretteleggingskommando
`destring`	Konvertere fra alfanumeriske til numeriske verdier	Tilretteleggingskommando
`drop`	Slette variabler eller records (drop if)	Tilretteleggingskommando
`drop-labels`	Fjerne kodelister	Tilretteleggingskommando
`generate`	Lage ny variabel på basis av uttrykk	Tilretteleggingskommando

`import`	Import av variabel til datasett	Tilretteleggingskommando
`import-event`	Import av forløpsvariabel (tidsrom) til datasett (må være tomt)	Tilretteleggingskommando
`import-panel`	Import av paneldata til datasett (må være tomt)	Tilretteleggingskommando
`keep`	Behold variabler eller records (slett resten)	Tilretteleggingskommando
`list-labels`	List ut egendefinerte value/label-lister	Tilretteleggingskommando
`merge`	Koble variabler fra et datasett inn i et annet. Koblingsnøkkel er enhetsidentifikator i kildedatasettet. Dette kan overstyres vha. en “on”-opsjon der en selv bestemmer koblingsnøkkel.	Tilretteleggingskommando
`recode`	Omkoding av numeriske variabler	Tilretteleggingskommando
`rename`	Omdøping av variabel	Tilretteleggingskommando
`replace`	Endre innhold i eksisterende variabler	Tilretteleggingskommando
`sample`	Lage et tilfeldig utvalg av totalpopulasjon	Tilretteleggingskommando
`split`	Dele opp strengvariabler i deler	Tilretteleggingskommando
`anova`	Anova/ancova variansanalyse	Analysekommando
`barchart`	Søylediagram for kategoriske variabler	Analysekommando
`boxplot`	Boksplott for numeriske variabler	Analysekommando
`ci`	Konfidensintervall og standardfeil	Analysekommando
`coefplot`	Koeffisientplot for grafisk visning av koeffisientestimater	Analysekommando
`correlate`	Korrelasjonsmatrise	Analysekommando
`hausman`	Hausmantest av panelregresjonsmodeller	Analysekommando
`hexbin`	Anonymisert scatterplot	Analysekommando
`histogram`	Histogram	Analysekommando
`ivregress`	Lineær regresjon med instrumentvariabler	Analysekommando
`ivregress-predict`	Generere variabler basert på resultater fra en lineær regresjon med instrumentvariabler	Analysekommando
`logit`	Logistisk regresjonsanalyse: Logit	Analysekommando
`logit-predict`	Generere variabler basert på resultater fra en logit-regresjon	Analysekommando
`mlogit`	Multinomisk logistisk regresjonanalyse	Analysekommando
`mlogit-predict`	Generere variabler basert på resultater fra en mlogit-regresjon	Analysekommando
`normaltest`	Et utvalg normalfordelingstester for angitte variabler	Analysekommando

`oaxaca`	Oaxaca dekomponering av gruppespesifikke effekter	Analysekommando
`piechart`	Kakediagram	Analysekommando
`probit`	Logistisk regresjonsanalyse: Probit	Analysekommando
`probit-predict`	Generere variabler basert på resultater fra en probit-regresjon	Analysekommando
`regress`	Lineær regresjon	Analysekommando
`regress-predict`	Generere variabler basert på resultater fra en ordinær lineær regresjon	Analysekommando
`regress-panel`	Lineær regresjon for paneldata	Analysekommando
`regress-panel-predict`	Generere variabler basert på resultater fra en lineær regresjon for paneldata	Analysekommando
`sankey`	Sankey-diagram (overgangsdiagram)	Analysekommando
`summarize`	Oppsummerende statistikk (numeriske variabler)	Analysekommando
`summarize-panel`	Oppsummerende statistikk for paneldata	Analysekommando
`tabulate`	Frekvens- og volumtabeller (kategoriske variabler)	Analysekommando
`tabulate-panel`	Frekvenstabeller for paneldata	Analysekommando
`transitions-panel`	Overgangssannsynligheter for paneldata	Analysekommando

Vedlegg B: Oversikt over funksjoner

Matematiske funksjoner

- ln(arg1)
  - Beskrivelse: Den naturlige logaritmen av arg1 (den inverse av exp(arg1))
  - arg1: Positive verdier
  - Output: Verdier mellom -744 og 709

- log10(arg1)
  - Beskrivelse: Base 10-logaritmen av arg1
  - arg1: Positive verdier
  - Output: Verdier mellom −323 og 308

- exp(arg1)
  - Beskrivelse: Eksponentialfunksjonen e^{arg1}(den inverse av ln(arg1))
  - arg1: Verdier mellom −8e+307 og 709
  - Output: Verdier >= 0

- sqrt(arg1)
  - Beskrivelse: Kvadratroten av arg1
  - arg1: Verdier >= 0
  - Output: Verdier >= 0

- abs(arg1)
  - Beskrivelse: Absoluttverdien av arg1 (dvs. fjerner negative fortegn)
  - arg1: Positive eller negative verdier
  - Output: Verdier >= 0

- sin(arg1)
  - Beskrivelse: Sinusverdien av arg1
  - arg1: Positive eller negative verdier
  - Output: Verdier mellom -1 og 1

- cos(arg1)
  - Beskrivelse: Returnerer cosinusverdien av arg1
  - arg1: Positive eller negative verdier
  - Output: Verdier mellom -1 og 1

- tan(arg1)
  - Beskrivelse: Returnerer tangensverdien av arg1
  - arg1: Positive eller negative verdier
  - Output: Positive eller negative verdier eller missing

- asin(arg1)
  - Beskrivelse: Returnerer radianverdien av arcsinusen til arg1
  - arg1: Verdier mellom -1 og 1
  - Output: Verdier mellom $-\pi/2$ og $\pi/2$

- acos(arg1)
  - Beskrivelse: Returnerer radianverdien av arccosinusen til arg1
  - arg1: Verdier mellom -1 og 1
  - Output: Verdier mellom 0 og π

- atan(arg1)
  - Beskrivelse: Returnerer radianverdien av arctangens til arg1
  - arg1: Positive eller negative verdier
  - Output: Verdier mellom $-\pi/2$ to $\pi/2$

- ceil(arg1)
  - Beskrivelse: Heltallsavrunding oppover
  - arg1: Positive eller negative verdier
  - Output: Positive eller negative heltallsverdier
  - Eksempler: `ceil(5.2) = 6`
    `ceil(-5.2) = -6`

- floor(arg1)
  - Beskrivelse: Heltallsavrunding nedover. Tilsvarer funksjonen int(arg1)
  - arg1: Positive eller negative verdier
  - Output: Positive eller negative heltallsverdier
  - Eksempler: `floor(5.8) = 5`
    `floor(-5.8) = -5`

- int(arg1)
  - Beskrivelse: Heltallsverdien av arg1 (dvs. dropper desimaltall). Tilsvarer funksjonen floor(arg1)
  - arg1: Positive eller negative verdier
  - Output: Positive eller negative heltallsverdier
  - Eksempler: `int(5.8) = 5`
    `int(-5.8) = -5`

- logit(arg1)
  - Beskrivelse: Logverdien av oddsratioen til arg1 (= ln {arg1/(1-arg1)})
  - arg1: Verdier mellom 0 og 1 (ikke inkludert)
  - Output: Positive eller negative verdier eller missing

- lnfactorial(arg1)
  - Beskrivelse: Den naturlige logaritmen av n-faktor (= ln(n!))
  - n: Heltallsverdier >= 0
  - Output: Verdier >= 0

- comb(n,k)
  - Beskrivelse: Kombinatorisk funksjonsverdi (= n!/{k!(n-k)!})
  - n: Heltallsverdier >= 1
  - k: Heltallsverdier mellom 0 og n
  - Output: Verdier >= 0 eller missing

- round(arg1,arg2)
  - Beskrivelse: Avrunder til nærmeste heltall dersom arg2 utelates eller settes lik 1. arg2 bestemmer hvilket nivå en skal avrunde på
  - arg1: Positive eller negative verdier
  - arg2: Positive eller negative verdier (default = 1 dersom arg2 droppes)
  - Output: Positive eller negative verdier
  - Eksempler:
    ```
    round(5.2) = 5
    round(5.8) = 6
    round(5.8,1) = 6
    round(5.8,5) = 5
    round(5.8,10) = 10
    round(5.8621,0.01) = 5.86
    ```

- quantile(arg1, arg2)
  - Beskrivelse: Returnerer verdi basert på rangeringen av en kontinuerlig verdi over en valgt inndeling med like mange verdier i hver gruppe. Mulige inndelinger: 2-100. Om 100 brukes som argument, returneres verdiene 0-99 basert på hvilket prosentil en verdi befinner seg i. Brukes verdien 10, grupperes verdier i desiler (0-9)
  - arg1: Variabel med kontinuerlige verdier
  - arg2: Heltallsverdier mellom 2 og 100
  - Output: Heltallsverdier mellom 0 og 99
  - Eksempler: generate innt_p100 = quantile(inntekt,100) (lager prosentiler)
    generate innt_p10 = quantile(inntekt,10) (lager desiler)
    generate innt_p4 = quantile(inntekt,4) (lager kvartiler)

Rekke-beregninger (der 2 eller flere variabler inngår)

- rowmin(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut minimumsverdien av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
  - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
  - Output: Numeriske verdier eller missing
  - Eksempler: generate minverdi = rowmin(1,2,3,4)
    generate min_innt = rowmin(innt18,innt19,innt20)

- rowmax(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut maksverdien av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
  - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
  - Output: Numeriske verdier eller missing
  - Eksempler: generate maxverdi = rowmax(1,2,3,4)
    generate max_innt = rowmax(innt18,innt19,innt20)

- rowmean(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut gjennomsnittsverdien av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
  - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
  - Output: Numeriske verdier eller missing
  - Eksempler: generate snittverdi = rowmean(1,2,3,4)
    generate snitt_innt = rowmean(innt18,innt19,innt20)

- rowmedian(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut medianverdien av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
  - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
  - Output: Numeriske verdier eller missing
  - Eksempler: generate medianverdi = rowmedian(1,2,3,4)
    generate median_innt = rowmedian(innt18,innt19,innt20)

- rowstd(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut standardavviket av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
  - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
  - Output: Numeriske verdier eller missing
  - Eksempler: generate stdverdi = rowstd(1,2,3,4)
    generate std_innt = rowstd(innt18,innt19,innt20)

- rowtotal(arg1, arg2, ….., argn)
    - Beskrivelse: Henter ut summen av arg1, arg2, ….., argn for den gitte enhet. Returnerer missingverdi dersom minst en av argumentene inneholder missing.
    - arg1, arg2, ….., argn: Numeriske verdier eller variabler med numeriske verdier
    - Output: Numeriske verdier eller missing
    - Eksempler: generate sumverdi = rowtotal(1,2,3,4)
      generate sum_innt = rowtotal(innt18,innt19,innt20)

- rowmissing(arg1, arg2, ….., argn)
    - Beskrivelse: Henter ut antallet missingverdier blant arg1, arg2, ….., argn for den gitte enhet.
    - arg1, arg2, ….., argn: Numeriske eller alfanumeriske verdier, eller variabler med numeriske eller alfanumeriske verdier
    - Output: Numeriske verdier
    - Eksempler: generate missverdi = rowmissing(1,2,3,4)
      generate miss_innt = rowmissing(innt18,innt19,innt20)

- rowvalid(arg1, arg2, ….., argn)
    - Beskrivelse: Henter ut antallet gyldige verdier blant arg1, arg2, ….., argn for den gitte enhet.
    - arg1, arg2, ….., argn: Numeriske eller alfanumeriske verdier, eller variabler med numeriske eller alfanumeriske verdier
    - Output: Numeriske verdier
    - Eksempler: generate gyldigverdi = rowvalid(1,2,3,4)
      generate gyldig_innt = rowvalid(innt18,innt19,innt20)

- rowconcat(arg1, arg2, ….., argn)
    - Beskrivelse: Slår sammen arg1, arg2, ….., argn for den gitte enhet.
    - arg1, arg2, ….., argn: Alfanumeriske verdier, eller variabler med alfanumeriske verdier
    - Output: Alfanumeriske verdier
    - Eksempler: generate concatverdi = rowconcat('1','2','3','4')
      generate fulltnavn = rowconcat('Ole ','Olsen')
      generate fulltnavn = rowconcat(var1,var2)

Strengefunksjoner

- string(arg1)
  - Beskrivelse: Konverterer verdien arg1 til string-format
  - arg1: Positive eller negative verdier eller missing
  - Output: Verdien arg1 konvertert til string-format
  - Eksempler: `string(1234567) = ‘1234567’`

- upper(arg1)
  - Beskrivelse: Konverterer tekst/string til “uppercase” (ASCII) (unicode-karakterer utenfor ASCII-spekteret ignoreres)
  - arg1: String-verdier
  - Output: String-verdier konvertert til “uppercase”
  - Eksempler: `upper(‘abcde’) = ‘ABCDE’`
    `upper(‘abcdé’) = ‘ABCDé’`

- lower(arg1)
  - Beskrivelse: Konverterer tekst/string til “lowercase” (ASCII) (unicode-karakterer utenfor ASCII-spekteret ignoreres)
  - arg1: String-verdier
  - Output: String-verdier konvertert til “lowercase”
  - Eksempler: `lower(‘ABCDE’) = ‘abcde’`
    `lower(‘ABCDÉ’) = ‘abcdÉ’`

- ltrim(arg1)
  - Beskrivelse: Fjerner ledende blanke karakterer (mellomrom) fra tekst
  - arg1: String-verdier
  - Output: String-verdier der ledende blanke karakterer er fjernet
  - Eksempler: `trim(‘    this’) = ‘this’`

- rtrim(arg1)
  - Beskrivelse: Fjerner blanke karakterer (mellomrom) fra slutten tekst
  - arg1: String-verdier
  - Output: String-verdier der alle blanke karakterer er fjernet fra slutten av tekstverdi
  - Eksempler: `trim(‘this    ’) = ‘this’`

- trim(arg1)
  - Beskrivelse: Fjerne blanke karakterer (mellomrom) både fra start og slutt på tekstverdi
  - arg1: String-verdier
  - Output: String-verdier der blanke karakterer er fjernet både fra start og slutt av tekstverdi
  - Eksempler: `trim('    this    ') = 'this'`

- length(arg1)
  - Beskrivelse: Angir antall karakterer i en tekstverdi (ASCII) (merk at for unicode-karakterer utenfor ASCII-spekteret angis verdien i antall bytes i stedet)
  - arg1: String-verdier
  - Output: Heltall >= 0
  - Eksempler: `length('ab') = 2`

- substr(arg1,arg2,arg3)
  - Beskrivelse: Henter ut delteksten som starter ved posisjon arg2 og med lengde arg3
  - arg1: String-verdier
  - arg2: Heltall >=1 og <= -1 (ved negative verdier regnes posisjon i forhold til posisjonen til siste karakter)
  - arg3: Heltall >= 1
  - Output: Deltekst av arg1
  - Eksempler:
    ```
    substr('y32ssx',2,3) = '32s'
    substr('y32ssx',-3,2) = 'ss'
    substr('y32ssx',1,1) = 'y'
    ```

Sysmiss

- sysmiss(arg1)
  - Beskrivelse: Logisk funksjon som settes til "true" dersom variabelen arg1 har verdien "missing" (= ingen observasjoner i underliggende datasett)
  - arg1: Variabel (alle typer)
  - Output: 1 ("true") eller 0 ("false")
  - Eksempler: `generate var1 = 0 if sysmiss(var2)`

Logiske funksjoner

- inlist(arg1, arg2, arg3, …., argn)
  - Beskrivelse: Logisk funksjon som settes til “true” dersom verdien til arg1 finnes blant de resterende argumentene arg2, arg 3, …., argn.
  - arg1, arg2, arg3, …, argn: Variabel eller verdi (alle typer)
  - Output: 1 (“true”) eller 0 (“false”)
  - Eksempler: `generate var1 = 1 if inlist(var2, 1, 3, 5)`
    `generate var1 = 1 if inlist('1', var2, var3, var4)`

- inrange(arg1, arg2, arg3)
  - Beskrivelse: Logisk funksjon som settes til “true” dersom verdien til arg1 er større enn eller lik arg2 og mindre enn eller lik arg3.
  - arg1, arg2, arg3: Variabel eller verdi (alle typer)
  - Output: 1 (“true”) eller 0 (“false”)
  - Eksempel: `generate var1 = 1 if inrange(var2, 500000, 1000000)`

Tetthetsfunksjoner

- ibeta(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative beta-fordelingen med formparametrene arg1 og arg2, også kalt den regulariserte ufullstendige betafunksjonen eller den ufullstendige betafunksjonsratioen (ibeta() = 0 dersom arg3 < 0, ibeta() = 1 dersom arg3 > 1)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier mellom 0 og 1

- betaden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til beta-fordelingen med formparametrene arg1 og arg2 (betaden() = 0 dersom arg3 < 0 eller arg3 > 1)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier >= 0

- ibetatail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative beta-fordelingen med formparametrene arg1 og arg2, også kalt den komplementære ufullstendige betafunksjonen (ibetatail() = 1 dersom arg3 < 0, ibetatail() = 0 dersom arg3 > 1)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier mellom 0 og 1

- invibeta(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative beta-fordelingen med formparametrene arg1 og arg2
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1

- invibetatail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative beta-fordelingen med formparametrene arg1 og arg2 (ibetatail(a,b,x) = p => invibetatail(a,b,p) = x)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1

- binomial(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) eller færre suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3. binomial() = 0 dersom arg2 < 0. binomial() = 1 dersom arg2 > arg1
  - arg1: Verdier >= 0
  - arg2: Positive eller negative verdier (relevante verdier: 0 <= arg2 < arg1)
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1

- binomialp(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3
  - arg1: Verdier mellom 1 og 1e+6
  - arg2: Verdier mellom 0 og arg1
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1

- binomialtail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) eller flere suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3. binomialtail() = 1 dersom arg2 < 0. binomialtail() = 0 dersom arg2 > arg1
  - arg1: Verdier >= 0
  - arg2: Positive eller negative verdier (relevante verdier: 0 <= arg2 < arg1)
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1

- chi2(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2() = 0 dersom arg2 < 0)
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier (relevante verdier: arg2 >= 0)
  - Output: Verdier mellom 0 og 1

- chi2den(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til kjikvadratfordelingen med arg1 frihetsgrader (chi2den() = 0 dersom arg2 < 0)
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier (relevante verdier: arg2 >= 0)
  - Output: Verdier >= 0

- chi2tail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2tail() = 1 dersom arg2 < 0). chi2tail() = 1 − chi2()
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier (relevante verdier: arg2 >= 0)
  - Output: Verdier mellom 0 og 1

- invchi2(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse av den kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2(arg1,arg2) = p => invchi2(arg1,p) = arg2)
  - arg1: Positive verdier
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier >= 0

- invchi2tail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse av den omvendte kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2tail(arg1,arg2) = p => invchi2tail(arg1,p) = arg2)
  - arg1: Positive verdier
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier >= 0

- nchi2(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1

- nchi2den(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2den() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier >= 0

- nchi2tail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2tail() = 1 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1

- t(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den kumulative Student’s t-fordelingen med arg1 frihetsgrader
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier
  - Output: Verdier mellom 0 og 1

- tden(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til Student's t-fordelingen med arg1 frihetsgrader
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier
  - Output: Verdier mellom 0 og 0.39894 . . .

- ttail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative Student's t-fordelingen med arg1 frihetsgrader
  - arg1: Positive verdier
  - arg2: Positive eller negative verdier
  - Output: Verdier mellom 0 og 1

- invt(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative Student's t-fordelingen med arg1 frihetsgrader (t(arg1,arg2) = p => invt(arg1,p) = arg2)
  - arg1: Positive verdier
  - arg2: Verdier mellom 0 og 1
  - Output: Positive eller negative verdier

- invttail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative Student's t-fordelingen med arg1 frihetsgrader (ttail(arg1,arg2) = p => invttail(arg1,p) = arg2)
  - arg1: Positive verdier
  - arg2: Verdier mellom 0 og 1
  - Output: Positive eller negative verdier

- nt(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (nt(arg1,0,arg3) = t(arg1,arg3))
  - arg1: Positive verdier
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Positive eller negative verdier
  - Output: Verdier mellom 0 og 1

- ntden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2
  - arg1: Positive verdier
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Positive eller negative verdier
  - Output: Verdier mellom 0 og 0.39894 . . .

- nttail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvente kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2
  - arg1: Positive verdier
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Positive eller negative verdier
  - Output: Verdier mellom 0 og 1

- invnttail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (nttail(arg1,arg2,arg3) = p => invnttail(arg1,arg2,p) = arg3)
  - arg1: Verdier mellom 1 og 1e+6
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Verdier mellom 0 og 1
  - Output: Positive eller negative verdier

- F(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (F() = 0 dersom arg3 < 0)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1

- Fden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Fden() = 0 dersom arg3 < 0)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier >= 0

- Ftail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Ftail() = 1 - F(), Ftail() = 1 dersom arg3 < 0)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Positive eller negative verdier (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1

- invF(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (F(arg1,arg2,arg3) = p => invF(arg1,arg2,p) = arg3)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier >= 0

- invFtail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Ftail(arg1,arg2,arg3) = p => invFtail(arg1,arg2,p) = arg3)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier >= 0

- nF(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nF(arg1,arg2,0,arg4) = F(arg1,arg2,arg4), nF() = 0 dersom arg4 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+8
  - arg2: Verdier mellom 2e–10 og 1e+8
  - arg3: Verdier mellom 0 og 10000
  - arg4: Positive eller negative verdier (relevante verdier: arg4 >= 0)
  - Output: Verdier mellom 0 og 1

- nFden(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFden(arg1,arg2,0,arg4) = Fden(arg1,arg2,arg4), nFden() = 0 dersom arg4 < 0)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1000
  - arg4: Positive eller negative verdier (relevante verdier: arg4 >= 0)
  - Output: Verdier >= 0

- nFtail(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFtail() = 1 dersom arg4 < 0)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1000
  - arg4: Positive eller negative verdier (relevante verdier: arg4 >= 0)
  - Output: Verdier mellom 0 og 1

- invnFtail(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFtail(arg1,arg2,arg3,arg4) = p => invnFtail(arg1,arg2,arg3,p) = arg4)
  - arg1: Positive verdier
  - arg2: Positive verdier
  - arg3: Verdier mellom 0 og 1000
  - arg4: Verdier mellom 0 og 1
  - Output: Verdier >= 0

- normal(arg1)
  - Beskrivelse: Returnerer en verdi fra den kumulative standardiserte normalfordelingen
  - arg1: Positive eller negative verdier
  - Output: Verdier mellom 0 og 1

- normalden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra normalfordelingen med snittverdi arg2 og standardavvik arg3
  - arg1: Positive eller negative verdier
  - arg2: Positive eller negative verdier
  - arg3: Positive verdier
  - Output: Verdier >= 0

Datofunksjoner

Dateringer i microdata.no benytter et datoformat som angir antall dager målt fra 01.01.1970. Fordelen med dette er at en enkelt kan gjøre beregninger på antall dager mellom to tidspunkter, og for eksempel beregne varighet i en tilstand (varighet = stoppdato - startdato).

Datofunksjonene listet opp nedenfor kan brukes til å gjøre om fra innebygd datoformat til mer intuitive verdier, som for eksempel årstall, måned, dag i uken osv.

- date(arg1, arg2, arg3)
  - Beskrivelse: Gjør om fra angitt dato til innebygd datoformat (antall dager fra 01.01.1970)
  - arg1: Årstall (4-sifret)
  - arg2: Måned (1-12)
  - arg3: Dag (1-31)
  - Output: Innebygd datoformat (antall dager fra 01.01.1970)
  - Eksempler:
    - date(2015,12,31) = 16800
    - date(1970,1,1) = 0
    - date(1967,5,27) = -950

- year(arg1)
  - Beskrivelse: Henter ut årstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Årstall tilsvarende datoverdi
  - Eksempler:
    - year(16800) = 2015
    - year(0) = 1970
    - year(-950) = 1967

- month(arg1)
  - Beskrivelse: Henter ut månedsverdi fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Månedsverdi tilsvarende datoverdi (1-12)
  - Eksempler:
    - month(16800) = 12
    - month(0) = 1
    - month(-950) = 5

- day(arg1)
  - Beskrivelse: Henter ut verdi for dag i måneden fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i måneden tilsvarende datoverdi (1-31)
  - Eksempler:
    - day(16800) = 31
    - day(0) = 1
    - day(-950) = 27

- dow(arg1)
  - Beskrivelse: Henter ut verdi for dag i uken fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i uken tilsvarende datoverdi (1-7) (1 = mandag, 2 = tirsdag etc)
  - Eksempler:
    - dow(16800) = 4
    - dow(0) = 4
    - dow(-950) = 6

- doy(arg1)
  - Beskrivelse: Henter ut verdi for dag i året fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i året tilsvarende datoverdi (1-366)
  - Eksempler:
    - doy(16800) = 365
    - doy(0) = 1
    - doy(-950) = 147

- week(arg1)
  - Beskrivelse: Henter ut ukenummer fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Ukenummer tilsvarende datoverdi (1-53)
  - Eksempler:
    - week(16800) = 53
    - week(0) = 1
    - week(-950) = 21

- quarter(arg1)
  - Beskrivelse: Henter ut kvartalstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Kvartalstall tilsvarende datoverdi (1-4)
  - Eksempler:
    - quarter(16800) = 4
    - quarter(0) = 1
    - quarter(-950) = 2

- halfyear(arg1)
  - Beskrivelse: Henter ut halvårstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Halvårstall tilsvarende datoverdi (1-2)
  - Eksempler:
    - halfyear(16800) = 2
    - halfyear(0) = 1
    - halfyear(-950) = 1

- isoformatdate(arg1)
  - Beskrivelse: Konverterer fra datoverdi til formatet YYYY-MM-DD. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Datering på formatet YYYY-MM-DD (string-verdier)
  - Eksempler:
    - isoformatdate(16800) = '2015-12-31'
    - isoformatdate(0) = '1970-01-01'
    - isoformatdate(-950) = '1967-05-27'

Vedlegg C: Konfidensialitet i microdata.no

Bakgrunn

Lov om offisiell statistikk og Statistisk sentralbyrå (LOV-2019-06-21-32) § 14 (tilgang til opplysninger for statistiske resultater og analyser) ledd (5) sier at «Taushetsplikten etter § 8 gjelder tilsvarende for den som får tilgang til opplysninger». Slike data kan kun utleveres til forskere i godkjente forskningsinstitusjoner og til offentlige myndigheter. Det stilles derfor strenge krav for tilgang til mikrodata, og søknad om tilgang til mikrodata for forskning er en lang prosess. Kriteriene for å søke om tilgang til registerdata for forskning kan du finne på SSBs sider om data til forskning.

Analysesystemet microdata.no er utviklet for å gjøre det mulig å få tilgang til mikrodata fra registre uten å måtte gå gjennom den omstendelige søknadsprosessen det er for å få data utlevert. Men det er en betingelse for en slik forenkling at sikkerheten og konfidensialiteten til mikrodataene er like godt ivaretatt som ved utlevering, helst bedre. Det har derfor fra begynnelsen vært et eksplisitt krav at brukerne ikke skal kunne se mikrodata eller på annen måte være i stand til å avsløre informasjon om enkeltpersoner. Når SSB publiserer offisiell statistikk er det aggregerte data. Likevel må SSB passe på ved ulike typer tiltak at det ikke er mulig å føre informasjon tilbake til enkeltpersoner eller andre typer statistiske enheter som statistikken dreier seg om.

De resultatene som microdata.no produserer for sine brukere, tabeller eller analyser, er i likhet med SSBs statistikk aggregerte data. Men uten begrensninger vil en bruker av microdata.no lett kunne produsere tabeller og andre typer statistiske resultater som SSB ikke ville kunne publisere. For å forhindre at det skjer er det innført flere typer tiltak som skal begrense mulighetene for å kunne avsløre informasjon som skal være konfidensiell.

Dette vedlegget vil beskrive de tiltakene som er implementert for å ivareta konfidensialitet i microdata.no. Tiltakene er basert på scenarier for hvordan konfidensialitet i microdata.no kan angripes eller utilsiktet komme i fare. Disse scenariene vil ikke bli beskrevet. Det vil bli lagt vekt på det som er nødvendig for at brukeren skal kunne forstå tiltakene og forholde seg til de statistiske resultatene på riktig måte.

De tiltakene som er beskrevet nedenfor er de som er implementert så langt. Det vil kunne komme flere tiltak etter hvert eller også justeringer av de tiltakene som er beskrevet nedenfor. Dette vedlegget vil bli oppdatert når det kommer endringer.

Tiltak 1: Minste populasjonsstørrelse

Det er ikke tillatt å definere undersøkelsespopulasjoner med færre enn 1000 personer. Forsøk på å definere slike populasjoner vil bli møtt med en melding av typen

```
Problem på linje 3:
Stoppet av avsløringskontroll. For få enheter i måldatasettet
```

Tiltak 2: Winsorisering

Winsorisering er en teknikk som ofte brukes i analyser for å hindre at ekstreme observasjoner skal få for stor innvirkning på analyseresultatene. Teknikken anvendes på alle numeriske variabler og består i å kutte av fordelingen i begge ender ved bestemte prosentiler. Vi benytter 2% winsorisering som betyr at de 1% høyeste verdiene settes til 99-prosentilen (nedre grenseverdi) og de 1% laveste verdiene settes til 1-prosentilen (øvre grenseverdi). Dette skjer kun ved visning av statistiske resultater, med utgangspunkt i den aktuelle populasjon som statistikken beregnes ut i fra.

Siden fordelingene til mange numeriske statistiske variabler er skjevfordelte, typisk med lange haler i øvre ende (eks. inntekter og formuer), vil winsorisering påvirke gjennomsnitt og standardavvik til en viss grad. Begge typer statistikker vil typisk bli estimert for lave. På den andre siden vil medianer, kvartiler og andre prosentiler ikke bli påvirket.

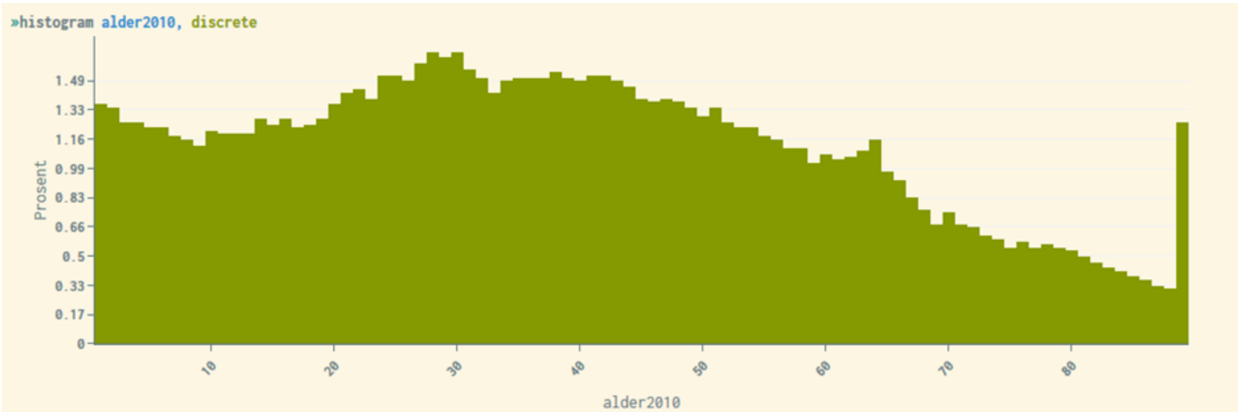
Eksempel: Betrakt følgende skript hvor målet er å lage deskriptive statistikker og histogram for aldersfordelingen i befolkningen per 1. januar 2010.

```
import BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate faar = floor(faarmnd/100)
drop faarmnd
generate alder2010 = 2010 - faar
summarize alder2010
histogram alder2010, discrete
```

Resultatet vil se slik ut:

`summarize alder2010`

Variabel	mean	std	count	1%	25%	50%	75%	99%
alder2010	38.7467	22.681	255743	1	21	37	55	89



Alle som er eldre enn 89 år vil bli satt til 89 år og 0-åringer vil bli satt til 1 år. Dette er årsaken til den store søylen til høyre i histogrammet. 0-åringer er her kun dem født 1. januar 2010. Vi er oppmerksomme på at winsoriseringen kan skape problemer ved statistiske fremstillinger av de aller eldste. Det samme gjelder andre grupper som er definert ved å tilhøre halen i fordelingen til en numerisk variabel.

Winsorisering påvirker alle deskriptive statistikker og grafiske plott, og hindrer at de mest ekstreme verdiene blir synlige.

Resultater som fremstilles gjennom regresjonsanalyser er ikke å betrakte som personidentifiserende informasjon. Ved slike analyser benyttes derfor de underliggende ikke-winsoriserte data. Regresjonsestimater vil derfor ikke bli påvirket av winsorisering.

Tiltak 3: Støylegging

Alle opptellinger av antall enheter i et datasett som vises i forbindelse med diverse operasjoner, eller statistiske opptellinger som vises gjennom blant andre kommandoene **tabulate** eller **summarize** er støylagte. Summer av numeriske statistikkvariabler knyttet til enhetene i for eksempel en tabellcelle, for eksempel inntekter, er justert proporsjonalt med støyleggingen slik at gjennomsnittstall er upåvirket. Der hvor støyleggingen resulterer i at antall enheter bak summen blir 0, blir summen satt til 0 og gjennomsnittet, som da blir 0/0 blir satt til NaN.

La

n være det originale antallet uten støylegging (for eksempel i en tabellcelle) og,

X er støyen (stokastisk, helt tall)

Da vil microdata.no vise det støylagte tallet

$Y = X + n$

Støyleggingen er definert ut fra statistiske fordelinger med følgende krav

1. Minste positive tallet som kan vises i opptellinger, Y, skal være 5. Tallene 1-4 skal ikke kunne vises i tabellene. Men $Y = 0$ vil kunne forekomme
2. Ingen opptellinger (antall) skal støylegges med mer enn ±5, dvs. $-5 \leq X \leq 5$.
3. Det skal ikke være mulig å gjenta samme opptelling flere ganger innen rammen av samme populasjon og få ulike resultater. Støyleggingen skal i den forstand være *konstant*.
4. Det må ikke være mulig å skille mellom virkelige nuller og nuller som er et resultat av støylegging.
5. Støyen er stokastisk med forventning 0, $E(X) = 0$.
6. Under betingelsen 1-3 og 5 skal støyfordelingen som genererer X være en maksimum entropifordeling, det vil si at hvis hvis $p(x) = P(X = x), -5 \leq x \leq 5$, så skal $p(x)$ maksimere
$$\mathcal{E}(p|n) = -\sum_{x=-5}^{5} p(x|n) \log\big(p(x|n)\big) = -E(\log p(X|n))$$

Maksimum entropifordelingen er i en viss forstand den støyfordeling som fjerner mest informasjon om den originale verdien n under de gitte betingelsene.

For å støtte brukerens tolkning av den usikkerheten som støyleggingen medfører presenterer vi de eksakte støyfordelingene som betingelsene 1-3 og 5-6 genererer for ulike verdier av n i tabell C1. På grunnlag av tabell C1 utleder vi *konfidensforedlinger* er for n gitt Y i tabellene C2 og C3. En konfidensfordeling er selv en stokastisk størrelse som avhenger av den

observerte verdien av Y og ikke en sannsynlighetsfordeling for n. (Og her har *konfidens* ingenting å gjøre med konfidensialitet.)

$p(x)$	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n>=10$
$x=-5$	0,0000	0,0000	0,0000	0,0000	0,0000	0,2987	0,0000	0,0000	0,0000	0,0000	0,0909
$x=-4$	0,0000	0,0000	0,0000	0,0000	0,3988	0,0000	0,0000	0,0000	0,0000	0,1296	0,0909
$x=-3$	0,0000	0,0000	0,0000	0,5175	0,0000	0,0000	0,0000	0,0000	0,1908	0,1219	0,0909
$x=-2$	0,0000	0,0000	0,6555	0,0000	0,0000	0,0000	0,0000	0,2940	0,1634	0,1147	0,0909
$x=-1$	0,0000	0,8149	0,0000	0,0000	0,0000	0,0000	0,4880	0,2145	0,1400	0,1079	0,0909
$x=0$	1,0000	0,0000	0,0000	0,0000	0,0000	0,1573	0,2522	0,1565	0,1199	0,1015	0,0909
$x=1$	0,0000	0,0000	0,0000	0,0000	0,1657	0,1383	0,1303	0,1142	0,1027	0,0955	0,0909
$x=2$	0,0000	0,0000	0,0000	0,1646	0,1390	0,1217	0,0674	0,0833	0,0880	0,0899	0,0909
$x=3$	0,0000	0,0000	0,1499	0,1309	0,1166	0,1070	0,0348	0,0608	0,0754	0,0846	0,0909
$x=4$	0,0000	0,1108	0,1116	0,1041	0,0978	0,0941	0,0180	0,0444	0,0645	0,0796	0,0909
$x=5$	0,0000	0,0743	0,0830	0,0828	0,0821	0,0828	0,0093	0,0324	0,0553	0,0748	0,0909
$\text{Sum}(p(x))$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
$E(X\|n)$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$Var(X\|n)$	0,0000	4,4460	7,8320	10,2306	11,7688	12,6325	1,7215	3,9038	6,0585	8,0973	10,0000
$\mathcal{E}(p\|n)$	0,0000	0,6038	1,0126	1,3459	1,6219	1,8497	1,3775	1,8547	2,1210	2,2874	2,3979

Tabell C1 Sannsynlighetsfordelinger for støy for ulike verdier av n.

Kombinasjoner der $Y = n + X < 0$.

Kombibasjoner der $1 \leq Y = n + X \leq 5$

Ved å summere de tallene i tabell C1 som gir samme verdi for y= $x+n$, og dividere med summen (standardisere til sum lik 1) kan en utlede tabell C2. Tabell C2 angir det vi kan kalle *konfidensgrader* for hver verdi av n gitt den verdi y som microdata.no har returnert for Y. Siden vi her ser på n som et fast tall og ikke en stokastisk variabel, er konfidensgradene $cf(n|y)$ *ikke* sannsynligheter i vanlig forstand, selv om de summerer seg til 1.

For $n > 10$ vil støyfordelingen være flat med $p(x|n) = \frac{1}{11} \approx 0,0909$ for alle $x \epsilon \{-5, \ldots, 5\}$ som for $n = 10$.

Vær oppmerksom på at i tabeller støylegges indre celler og marginalceller uavhengig av hverandre. ***Støylagte tabeller blir derfor ikke additive****. Støyvariansen på marginalceller blir den samme som på indre celler, og mindre enn for summen av de indre cellene som de uten støylegging skal være en sum av.*

$cf(n\|Y=y)$	$Y=0$	$Y=5$	$Y=6$	$Y=7$	$Y=8$	$Y=9$	$Y=10$	$Y=11$	$Y=12$	$Y=13$	$Y=14$	$Y=15$
$n=0$	**0,2713**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=1$	**0,2211**	**0,0571**	**0,0486**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=2$	**0,1779**	**0,0772**	**0,0730**	**0,0670**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=3$	**0,1404**	**0,0848**	**0,0857**	**0,0840**	**0,0781**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=4$	**0,1082**	**0,0853**	**0,0910**	**0,0941**	**0,0922**	**0,0861**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=5$	**0,0810**	**0,0810**	**0,0905**	**0,0982**	**0,1009**	**0,0988**	**0,0930**	0,0000	0,0000	0,0000	0,0000	0,0000
$n=6$	0,0000	**0,2513**	**0,1650**	**0,1051**	**0,0635**	**0,0365**	**0,0202**	**0,0109**	0,0000	0,0000	0,0000	0,0000
$n=7$	0,0000	**0,1514**	**0,1404**	**0,1262**	**0,1077**	**0,0874**	**0,0683**	**0,0519**	**0,0356**	0,0000	0,0000	0,0000
$n=8$	0,0000	**0,0983**	**0,1070**	**0,1129**	**0,1130**	**0,1078**	**0,0988**	**0,0881**	**0,0710**	**0,0580**	0,0000	0,0000
$n=9$	0,0000	**0,0667**	**0,0798**	**0,0925**	**0,1017**	**0,1065**	**0,1073**	**0,1051**	**0,0930**	**0,0835**	**0,0761**	0,0000
$n=10$	0,0000	**0,0468**	**0,0595**	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=11$	0,0000	0,0000	**0,0595**	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=12$	0,0000	0,0000	0,0000	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=13$	0,0000	0,0000	0,0000	0,0000	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=14$	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=15$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=16$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=17$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=18$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0954**	**0,0924**	**0,0909**
$n=19$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0924**	**0,0909**
$n=20$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0909**
Sum	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tabell C2 Konfidensfordeling for n for ulike verdier av y. Positive konfidensgrader er gitt fet skrift.

$CD(n|y)$ i tabell C3 er kumulative aggregeringer av konfidensgradene i tabell C2 definert ved

$$CD(n|y) = \sum_{j=0}^{n} cd(j|y)$$

$CD(n\|y)$	$Y=0$	$Y=5$	$Y=6$	$Y=7$	$Y=8$	$Y=9$	$Y=10$	$Y=11$	$Y=12$	$Y=13$	$Y=14$	$Y=15$
$n=0$	0,2713	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=1$	0,4924	0,0571	0,0486	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=2$	0,6703	0,1343	0,1217	0,0670	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=3$	0,8107	0,2190	0,2073	0,1510	0,0781	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=4$	0,9190	0,3044	0,2983	0,2450	0,1703	0,0861	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=5$	1,0000	0,3854	0,3888	0,3432	0,2712	0,1849	0,0930	0,0000	0,0000	0,0000	0,0000	0,0000
$n=6$	1,0000	0,6367	0,5539	0,4483	0,3347	0,2214	0,1132	0,0109	0,0000	0,0000	0,0000	0,0000
$n=7$	1,0000	0,7882	0,6943	0,5746	0,4424	0,3088	0,1815	0,0627	0,0356	0,0000	0,0000	0,0000
$n=8$	1,0000	0,8864	0,8012	0,6875	0,5554	0,4166	0,2802	0,1508	0,1066	0,0580	0,0000	0,0000
$n=9$	1,0000	0,9532	0,8810	0,7800	0,6572	0,5231	0,3875	0,2559	0,1997	0,1415	0,0761	0,0000
$n=10$	1,0000	1,0000	0,9405	0,8533	0,7429	0,6185	0,4896	0,3622	0,2997	0,2369	0,1685	0,0909
$n=11$	1,0000	1,0000	1,0000	0,9267	0,8286	0,7139	0,5917	0,4685	0,3998	0,3323	0,2609	0,1818
$n=12$	1,0000	1,0000	1,0000	1,0000	0,9143	0,8092	0,6938	0,5748	0,4998	0,4277	0,3532	0,2727
$n=13$	1,0000	1,0000	1,0000	1,0000	1,0000	0,9046	0,7958	0,6811	0,5998	0,5230	0,4456	0,3636
$n=14$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,8979	0,7874	0,6999	0,6184	0,5380	0,4545
$n=15$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,8937	0,7999	0,7138	0,6304	0,5455
$n=16$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9000	0,8092	0,7228	0,6364
$n=17$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9046	0,8152	0,7273
$n=18$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9076	0,8182
$n=19$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9091
$n=20$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tabell C3 Kumulativ konfidensfordeling for ulike verdier av y.

La $\{5-9\}$ betegne verdimengden $\{5,6,7,8,9\}$. Konfidensgraden til verdimengden $\{5-9\}$ hvis vi observerer for eksempel $Y=7$, blir da

$$cd(\{5-9\}|Y=7) = CD(9|Y=7) - CD(4|Y=7) = 0{,}7800 - 0{,}2450 = 0{,}5350$$

Dette tilsvarer et konfidensintervall. Merk igjen at det ikke er det samme som sannsynligheter siden n ikke er stokastisk.

Hvis $Y > 15$ vil konfidensfordelingen $cd(y|n)$ i tabell C2 være flat lik $\frac{1}{11} \approx 0{,}0909$ for alle verdier av n fra $Y-5$ til $Y+5$ som for $Y = 15$. La a og b være hele tall med $b > a$. Anta at $Y = y \geq 15$. Da blir

$$cd(\{a-b\}|y) = CD(b|y) - CD(a|y) = (\min(b, y+5) - \max(a, y-5))/11$$

Eksempel: La $a = 37, b = 44$ og $y = 39$. Da blir

$$cd(\{37-44\}) = \frac{\min(44{,}44) - \max(37{,}34)}{11} = \frac{44-37}{11} = \frac{7}{11} \approx 0{,}6364.$$

Ved aggregering av numeriske størrelser, for eksempel i tabellceller, vil summer bil justert i forhold til støyleggingen.

La Z_i er verdien på en numerisk variabel (for eksempel en inntektsvariabel) på person nr. i og T_c den originale summen av denne variabelen i en celle c med n_c personer, altså

$$T_c = \sum_{i \epsilon c} Z_i,$$

Anta n_c er støylagt til Y_c. Da vil T_c bli justert til

$$T_c^* = \frac{Y_c}{n_c} T_c$$

Denne justeringen kan være dramatisk for celler med få observasjoner men vil ha mindre betydning i celler med mange observasjoner. Det er også meningen. Merk også at

$$\bar{Z}_c = \frac{T_c^*}{Y_c} = \frac{T_c}{n_c}$$

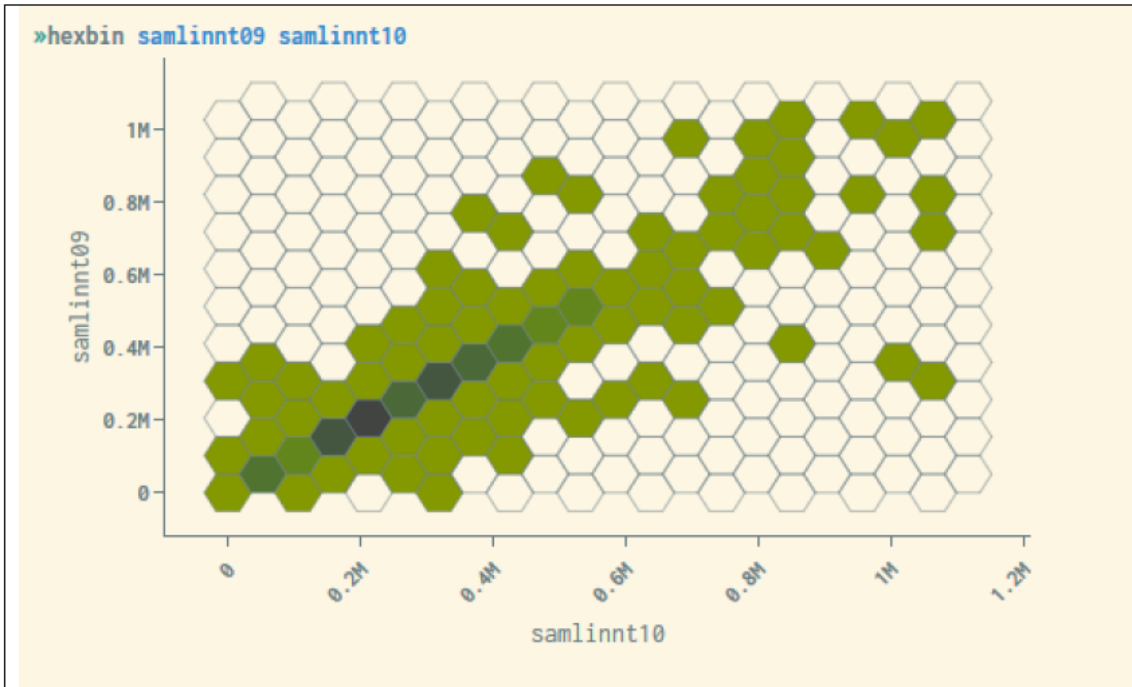
Slik at gjennomsnittet ikke blir berørt, bortsett fra hvis $Y_c = 0$. Da settes $\bar{Z}_c = NaN$.

Hvis $Y_c \leq 9$ vil også standardavvik, median og kvartiler, som kan bestilles ved **summarize** opsjonen i **tabulate** settes til NaN.

Tiltak 4: Grafiske plott. Hexbinplott

Det er vanlig å bruke spredningsplott for å etablere et visuelt bilde av data eller også vise sammenheng mellom numeriske variabler. Slike plott kan være veldig avslørende, spesielt hvis det er få observasjoner i forhold til det grafiske området og for de punktene i plottet som ligger utenfor hovedmassen av punkter. Hvis vi for en enhet/person i populasjonen kjenner verdien på en av de variablene som spenner ut plottet kan vi ofte lese av verdien til den andre variabelen med for stor nøyaktighet.

For å hindre at dette kan skje har vi i microdata.no valgt å *glatte* slike plott med en glatteteknikk. For dette formålet har vi forsøksvis valgt å fokusere på en teknikk som kalles *hexbin-plott*. I et hexbin-plott deles det grafiske området inn i regulære sekskanter. Et eksempel på et hexbin-plott laget i microdata.no er følgende:



Figur C1. Hexbinplott over samlet inntekt i 2009 mot 2010 for en liten populasjon.

I et hexbin-plott skaleres det grafiske området på grunnlag av de største og minste verdiene som forekommer for de variablene som plottes. De største og minste verdiene er påvirket av winsoriseringen omtalt i tiltak 2. Sekskantene gis en farge eller fargetone som angir et intervall for hvor mange enheter det er i dem, for eksempel 30-59, 60-89 osv. Intervallet for antall enheter/personer hver fargetone representerer er like langt for hver fargetone og tilpasses automatisk av fordelingen i data.

Hexbin plot er under utprøving. I den versjonen som foreløpig er lagt ut er alle sekskanter hvor antall personer er færre enn 20% av det mest befolkede hexagonet blanket. Dette kriteriet vil bli justert så snart det er mulig å gi det prioritet.

Tiltak 5: Skjuling av tabeller med for mange lave verdier

Tabeller som lages ved kommandoen **tabulate** vil i noen tilfeller kunne inneholde mange celler med lave verdier for antallet enheter. Dette kan være problematisk ettersom det gjør det lettere å indirekte identifisere individer ved å studere kombinasjoner av verdier for de kategoriske variablene som utgjør en tabell. Et annet problem med slike tabeller er at støyleggingen beskrevet under «Tiltak 3» gir en relativt høy usikkerhet for de aktuelle celleverdiene (den prosentvise støyen blir relativt stor ved små tall), slik at den statistiske nytteverdien av tabellen blir lav.

I microdata.no opereres det med en grenseverdi på 50%, dvs. tabeller der mer enn 50% av cellene inneholder frekvensverdier lavere enn 5, vil bli stoppet. I tillegg vil det vises en feilmelding om dette.

Det er mulig å unngå problemet med tabeller som stoppes pga. for mange lave verdier: Ved å lage grovere inndelinger for de kategoriske variablene som utgjør tabellen, eller ved å øke størrelsen på tabellpopulasjonen, vil en kunne øke antallet enheter i hver celle og dermed komme under 50%-grensen slik at tabellen blir godkjent og vises.