General Social Survey, Univariate Analysis

2024-09-02

- **L3 MIASHS**
- **Université Paris Cité**
- Année 2024-2025
- Course Homepage
- Moodle

> **!** **Objectives**

General Social Survey (GSS)

We will explore a (small) subset of the GSS dataset

> The GSS has been a reliable source of data to help researchers, students, and journalists monitor and explain trends in American behaviors, demographics, and opinions. You'll find the complete GSS data set on this site, and can access the GSS Data Explorer to explore, analyze, extract, and share custom sets of GSS data.

Data gathering

Download the data

```r
download_data <-  function(fname,
                           baseurl = 'https://stephane-v-boucheron.fr/data',
                           datapath = "../DATA") {
  fpath <- paste(datapath, fname, sep = "/")

  if (!file.exists(fpath)) {
    url <- paste(baseurl, fname, sep = "/")

    rep <- httr::GET(url)
    stopifnot(rep$status_code == 200)

    con <- file(fpath, open = "wb")
```

```r
    writeBin(rep$content, con)
    close(con)

    print(glue('File "{fname}" downloaded!'))
  } else {
    print(glue('File "{fname}" already on hard drive!'))
  }
}
```

```r
download_data(fname="sub-data.txt")
```

```r
download_data(fname="sub-cdbk.txt")
```

> Base `R` (package `utils`) offers a function `download.file()`. There is
>
> ```r
> fname <- 'sub-data.txt'
> baseurl <- 'https://stephane-v-boucheron.fr/data'
> download.file(url=paste(baseurl, fname, sep="/"),
>               destfile=paste('./DATA', fname, sep="/"))
> ```
>
> There is no need to (always) reinvent the wheel!

Load the data in your session

File inspection shows that the data file `sub-data.txt` is indeed a `csv` file

```
09:01 $ file DATA/sub-data.txt
DATA/sub-data.txt: CSV text
```

We do not know the peculiarities of this file formatting. We load it as if fields were separated by coma (`,`, this is an American file). and prevent any type inference by asserting that all columns should be treated as `character` (`c`).

Answer the following questions:

- What are the observations/individuals/sample points?
- What do the columns stand for?
- Is the dataset tidy/messy?

Inspect the schema of dataframe (there are 540 columns!)

NULL values

In the dataframe, NULL are encoded in several ways. From the metadata, we learn

```
           VALUE  LABEL
              .d  don't know
              .i  iap
              .j  I don't have a job
              .m  dk, na, iap
```

```
.n  no answer
.p  not imputable
.q  not imputable
.r  refused
.s  skipped on web
.u  uncodeable
.x  not available in this release
.y  not available in this year
.z  see codebook
```

```
Missing-data codes: .d,.i,.j,.m,.n,.p,.q,.r,.s,.u,.x,.y,.z
```

Using a *brute force* approach, we replace the *missing data codes* with `NA`, not the string `'NA'` but NULL value for character vectors `'NA_character_'`.

We first define a *regular expression* that will allow us to detect the presence of *missing data codes* in a string and to replace the *missing data code* by `'NA_character_'`

The repeated backslashes in `na_patterns` are due to the way `R` handles escape/control characters like \ or `.` which play an important role in the definition of regular expressions.

```r
na_patterns <- '.d,.i,.j,.m,.n,.p,.q,.r,.s,.u,.x,.y,.z' |>
  str_replace_all('\\.', '\\\\.') |>
  str_replace_all(',', '|')

na_patterns
```

```
[1] "\\.d|\\.i|\\.j|\\.m|\\.n|\\.p|\\.q|\\.r|\\.s|\\.u|\\.x|\\.y|\\.z"
```

> **Regular expressions**
>
> Regular expressions are a Swiss army knife when dealing with text data. Get acquainted with them. It is useful whenver you work data or edit a file
> See [Regular expressions in R](#)
> This is also useful when programming with `Python` or querying a relational database.

```r
df <- df |>
  mutate(across(
    everything(),
    \(x) str_replace(x, na_patterns, NA_character_)))  # Anonymous function in Python 4....
```

> Our handling of the *Missing-data codes* is fast, sloppy, and dirty. The occurrence of a specific code, say `.i` rather than `.r` might be a valuable information. For some columns, a specific treatment may be indeed if we do not want to waste information.

Downsizing the data

Project the dataframe `df` onto columns `year`, `age`, `sex`, `race`, `ethnic`, columns ending with `educ`, ending with `deg`, starting with `dwel`, starting with `income`, `hompop`, `earnrs`, `coninc`, `conrinc`.

Call the resulting dataframe `df_redux`.

Open the metadata file `sub-cdbk.txt` in your favorite editor to get a feeling of the column names meaning and of encoding conventions.

Howm many missing values per column ?

Drop NULL columns

Count the number of observations per year

Count for each year

```
df_redux |>
  count(`year`)
```

```
# A tibble: 8 x 2
  year      n
  <chr> <int>
1 2008   2023
2 2010   2044
3 2012   1974
4 2014   2538
5 2016   2867
6 2018   2348
7 2021   4032
8 2022   3544
```

`count()` is a shortcut for

```
df_redux |>
  group_by(`year`)
  summarize(n=n())
```

In SQL, we would write:

```sql
SELECT df."year", COUNT(*) AS n
FROM df_redux AS df
GROUP BY df."year"
```

Plot the number of rows per year as a *barplot*

Explore columns with name containing `inc`

Find the number of unique values in each column.

What are the unique values in columns whose name contains `income` ?

Make `income` and `rincome` a factor

Summarize and Visualize the distributions of `income` and `rincome`

The factors need reordering

Recode factors

Distribution of `year`

Make year an integer column

Plot `rincome` and `income` distributions with respect to `year`

Scatterplot of `conrinc` (y) with respect to `coninc`, facet by `sex`

Facet histogram for `conrinc` according to `income`

TODO

- Retype `age`
- Distribution of `age` (summary and visualization)
- Distribution of `age` (summary and visualization) with respect to `sex`
- Scatterplot of `conrinc` with respect to `age`
- Boxplot of `conrinc` with respect to `sex`

Retype `age`

Column `age` should be numeric.

Compute the numerical summary of column `age`

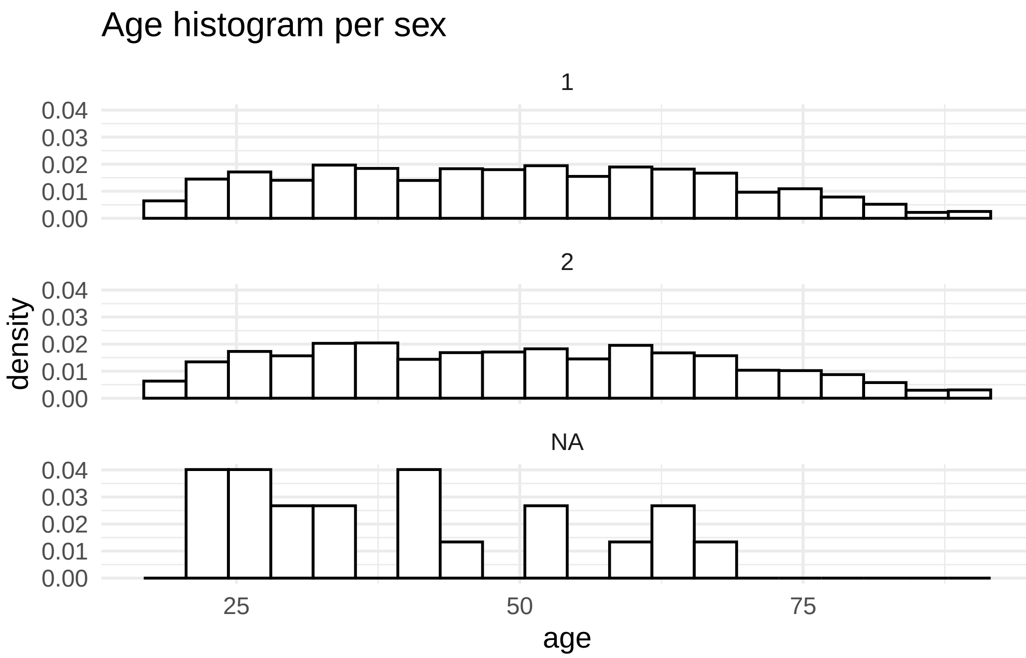
Boxplot of age with respect to sex

The `boxplot` delivers a graphical output starting from the robust estimators (the quartiles) of location and scale.

Histogram of age distribution facetted by `sex`

```
df_redux |>
  ggplot() +
  aes(x=age) +
  geom_histogram(aes(y=after_stat(density)), fill="white", color="black", bins = 20) +
  facet_wrap(~ sex, ncol = 1) +
  ggtitle("Age histogram per sex")
```

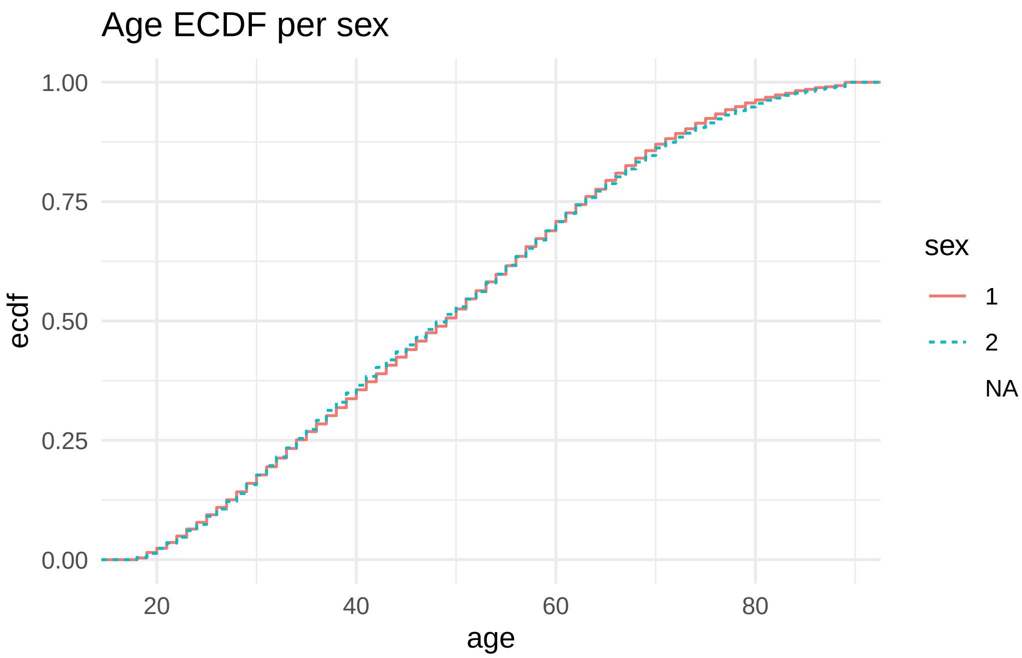
```
Warning: Removed 585 rows containing non-finite outside the scale range
(`stat_bin()`).
```



ECDF of age distribution by sex

```
df_redux |>
  ggplot() +
  aes(x=age, linetype=sex, color=sex) +
  stat_ecdf() +
  ggtitle("Age ECDF per sex")
```

```
Warning: Removed 585 rows containing non-finite outside the scale range
(`stat_ecdf()`).
```



Is "not responding to the question about `age`" associated with `sex`?

```
df_redux |>
  select(age, sex) |>
  mutate(age=is.na(age)) |>
  table() |>
  chisq.test()
```

```

	Pearson's Chi-squared test with Yates' continuity correction

data:  table(mutate(select(df_redux, age, sex), age = is.na(age)))
X-squared = 1.5752, df = 1, p-value = 0.2095
```

Scatterplot of `conrinc` with respect to `age`

Play with `geom_jitter`, transparency (alpha), point size, and logarithmic scale for income.

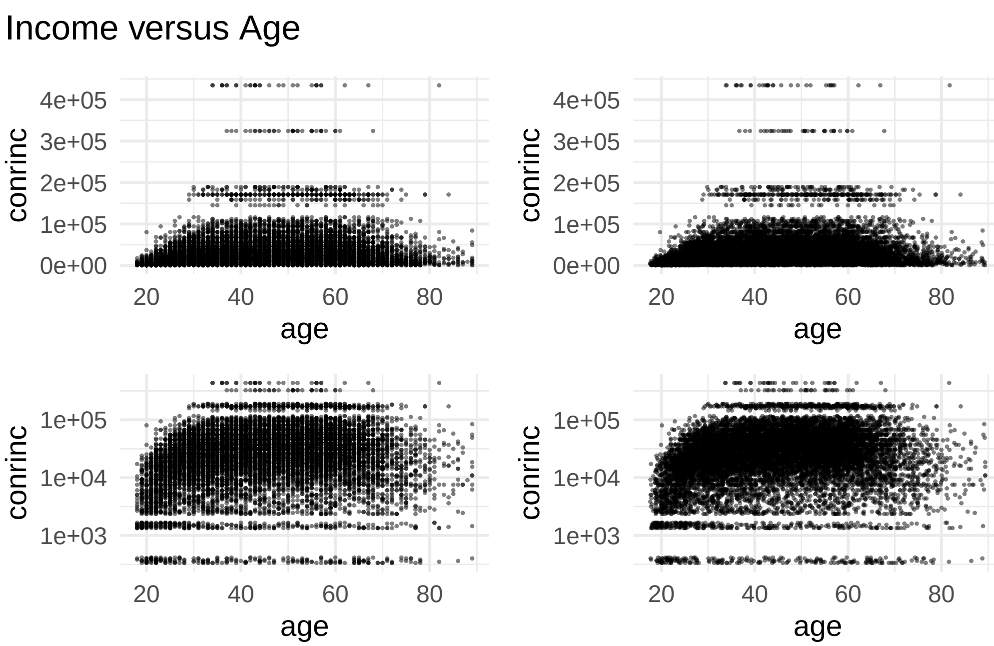
```
p <- df_redux |>
  ggplot() +
  aes(x=age, y=conrinc)

p1 <- p + geom_point(size=.1, alpha=.5)
p2 <- p + geom_jitter(size=.1, alpha=.5)
p3 <- p + geom_point(size=.1, alpha=.5) + scale_y_log10()
p4 <- p + geom_jitter(size=.1, alpha=.5) + scale_y_log10()

((p1 + p2) / (p3 + p4) ) + plot_annotation(
```

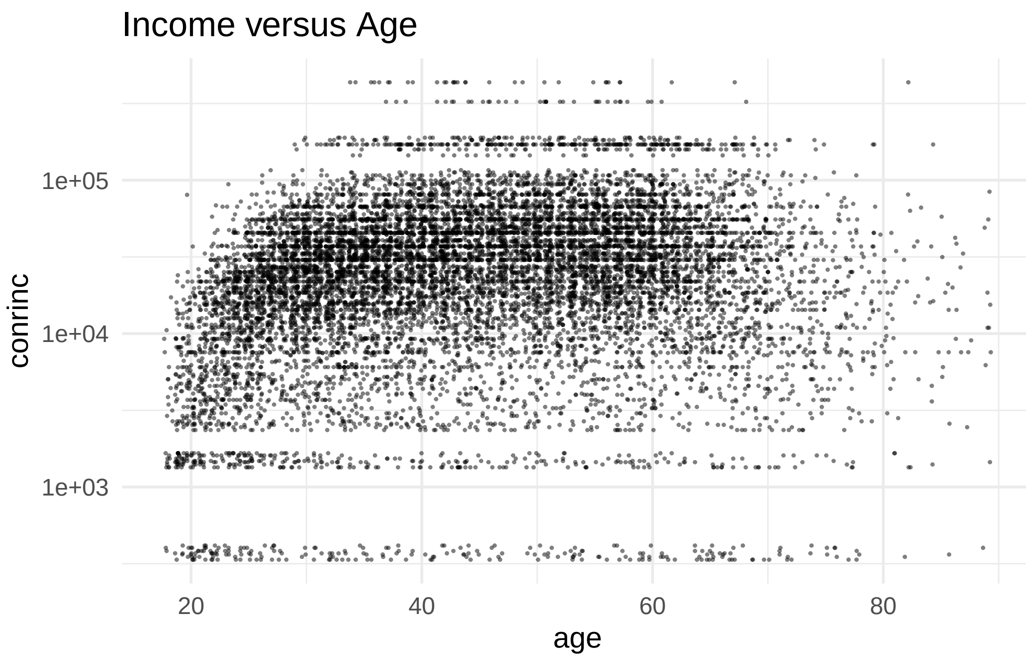
```
  title= 'Income versus Age'
)
```

```
Warning: Removed 9065 rows containing missing values or values outside the scale range
(`geom_point()`).
Removed 9065 rows containing missing values or values outside the scale range
(`geom_point()`).
Removed 9065 rows containing missing values or values outside the scale range
(`geom_point()`).
Removed 9065 rows containing missing values or values outside the scale range
(`geom_point()`).
```




```
p4 + ggtitle('Income versus Age')
```

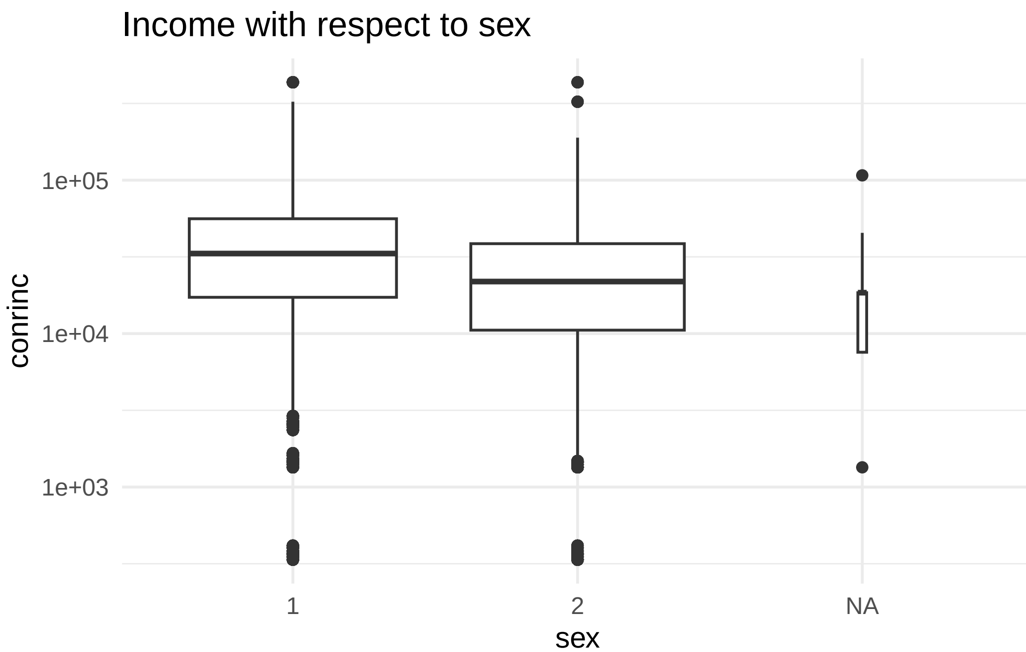
```
Warning: Removed 9065 rows containing missing values or values outside the scale range
(`geom_point()`).
```



Boxplot of `conrinc` with respect to `sex`

```
df_redux |>
  ggplot() +
  aes(x=sex, y=conrinc) +
  geom_boxplot(varwidth=T) +
  scale_y_log10() +
  ggtitle("Income with respect to sex")
```

```
Warning: Removed 8869 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```


Income with respect to sex
conrinc
1e+05
1e+04
1e+03
1
2
NA
sex