Automatic analysis of metadata for secondary tabular data protection

Clara Baudry [1]

[1]Insee (French National Statistical Institute)

Published tables ≠ protected tables

The R package `rtauargus` allows users to call Tau-Argus from R. It leverages the renowned SDC software Tau-Argus to automatically handle linked tables, extending its functionality beyond separate table processing [1]. It detects common cells between tables and flags it accordingly.

Nevertheless, for the package to detect common cells, a list of linked tables must be provided. The constitution of this list of tables demands a certain level of expertise in the understanding of how tables are linked from a confidentiality viewpoint [2].

Many statisticians in national institutes end up protecting tables only once a year, thus they hardly get to reach the level of expertise needed. The tool helps all staff go from the published tables metadata to the tables needing protection metadata.

Eurostat template

The template is a `.csv` file giving the structure of the `XML` files sent by the NSIs to Eurostat. Some columns are pre-filled, describing all the cells expected by the Eurostat. The analysis only focuses on these columns. The idea is to deduct from the cells description the tables to protect.

TIME_PERIOD	INDICATOR	ACTIVITY	NUMBER_EMPL	LEGAL_FORM
2022	SAL	B	E0	_T
2022	SAL	B	E1T4	_T
...	...	...	...	...
2022	SAL_DTH	BTSXO_S94	E0	_T
2021	SAL_DTH	BTSXO_S94	E1T4	_T
...	...	...	...	...

Table 1. Extract from a SBS Eurostat template

```
template_formatted <- format_template(
  data = enterprise_template,
  indicator_column = "INDICATOR",
  spanning_var_tot = list(
    ACTIVITY = "BTSXO_S94",
    NUMBER_EMPL = "_T",
    LEGAL_FORM = "_T"),
  field_columns = c("TIME_PERIOD")
)
```

In `rtauargus` [3] a function `template_formatted()` is being developed. It automatically analyses the cells of the file and returns the list of tables to protect.
The user needs to classify the variables: indicator, spanning and field variables. For the spanning variables, it is necessary to specify the modality corresponding to the total.

Table 2 shows the output of this function: a dataframe describing the tables that need protection (in the format of the input expected by the metadata analysis tool).

table_name	field	indic	span_1	span_2	hrc_span_1	hrc_span_2
2021_SAL_DTH_1	2021	SAL_DTH	ACT	LEGAL	hrc_activity_131	hrc_legal_3
2021_SAL_DTH_2	2021	SAL_DTH	ACT	NUMBER_EMPL	hrc_activity_131	hrc_number_4
2022_SAL_1	2022	SAL	ACT	LEGAL	hrc_activity_131	hrc_legal_3
2022_SAL_2	2022	SAL	ACT	NUMBER_EMPL	hrc_activity_131	hrc_number_4
2022_SAL_DTH_1	2022	SAL_DTH	ACT	LEGAL	hrc_activity_131	hrc_legal_3
2022_SAL_DTH_2	2022	SAL_DTH	ACT	NUMBER_EMPL	hrc_activity_131	hrc_number_4

Table 2. SBS metadata ready for automatic analysis

Automatic analysis of metadata

1. **Identify hierarchies**. All the spanning variables included in the same hierarchical relationship are renamed after the latter. Thus, all those variables are grouped in one dimension.
2. **Split in clusters**. Gather tables in groups that should be protected together because they share common cells. Those groups are called clusters. Each cluster is independent from the others.
3. **Detect tables included in other tables**. Analyse the spanning variables to detect tables included in other tables.
4. **Regroup tables that are included in each other**. Regroup tables that are included in each other in one table to treat.
5. **Provide dataframe with cluster assignment**. Each table needing protecting is described and assigned to a cluster. The tables assigned to the same clusters share common cells and must be treated jointly, using for example `rtauargus::tab_multi_manager()`.

Theoretical example: tables on pizza and lettuce turnover

The input is a dataframe containing the metadata. Table 3 shows an example of the said dataframe. It describes the tables of the turnover from the sales of pizzas and lettuces that will be published.

table	field	hrc_field	indic	hrc_indic	span_1	hrc_span_1	span_2	hrc_span_2
T1	2023	*NA*	to_pizza	*NA*	nuts2	hrc_nuts	size	*NA*
T2	2023	*NA*	to_pizza	*NA*	nuts3	hrc_nuts	size	*NA*
T3	2023	*NA*	to_pizza	*NA*	act	hrc_nace	nuts2	hrc_nuts
T4	2023	*NA*	to_pizza	*NA*	act	hrc_nace	nuts3	hrc_nuts
T5	2023	*NA*	to_batavia	hrc_lettuce	act	hrc_nace	size	*NA*
T6	2023	*NA*	to_arugula	hrc_lettuce	act	hrc_nace	size	*NA*
T7	2023	*NA*	to_lettuce	hrc_lettuce	act	hrc_nace	size	*NA*

Table 3. Pizza and lettuce turnover metadata

```
res <- analyse_metadata(
        df_metadata = meta_pizza_lettuce,
        verbose = TRUE # to get all the steps
)
```

Identify hierarchies and split in clusters

The tool renames all the variables by the name of their hierarchies. The confidentiality approach is in terms of hierarchies more than variables. Then, it analyses the fields and indicators in order to determine which tables are independent from each other. Tables are independent when they share no common cells. In this example, **2 clusters** are detected.

The tables T1, T2, T3 and T4 all refer to pizza turnover, they need to be treated together.

The tables T5, T6 and T7 all refer to lettuce turnover, they need to be treated together.

Tables included in each other

The tool analyses the spanning variables to detect tables included in other tables.

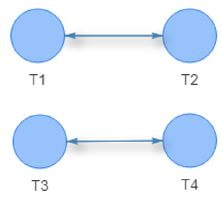

Figure 1. Cluster α: tables on pizza turnover

Only **2 tables** are needed in order to protect all the cells counting pizza turnover.

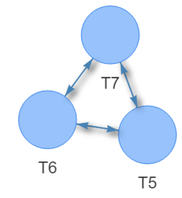

Figure 2. Cluster β: tables on lettuce turnover

Only **1 table** is needed in order to protect all the cells counting lettuce turnover.

Result of the analysis

Once the tool has gone through all stages it returns a dataframe with a cluster assignment variable.

cluster	table	indic	span_1	span_2	span_3	hrc_span_1	hrc_span_2	hrc_span_3
β	T5.T6.T7	lettuce	hrc_nace	size	hrc_lettuce			hrc_lettuce
α	T1.T2	to_pizza	hrc_nuts	size		hrc_nuts		
α	T3.T4	to_pizza	hrc_nace	hrc_nuts		hrc_nace	hrc_nuts	

Table 4. Result of the analysis: metadata of the tables to protect

The analysis shows that in order to protect the 7 tables to be published only 3 tables need to be protected. Moreover, 2 of them need a joint treatment which can be performed like so:

```
safe_tables <- tab_multi_manager(
  list_tables = list(T1T2 = pizza_nuts_size,
                     T3T4 = pizza_nace_nuts),
  list_explanatory_vars = list(T1T2 = c("NUTS", "size"),
                               T3T4 = c("act", "NUTS")),
  hrc = c(NUTS = "hrc_nuts.hrc", act = "hrc_nace.hrc"),
  freq = "N_OBS", value = "to", totcode =  "Total",
  secret_var = "is_secret_prim")
```

References

[1] J. Jamme and N. Rastout. *Protect several linked tables at once with rtauargus*. 2023. URL: `https://inseefrlab.github.io/rtauargus/articles/protect_multi_tables.html`.

[2] A. Hundepool et al. "Frequency tables". In: *Handbook on Statistical disclosure control, 2nd Edition*. STACE, 2024, pp. 177–179. URL: `https://raw.githubusercontent.com/sdcTools/HandbookSDC/gh-pages/Handbook-on-Statistical-Disclosure-Control.pdf`.

[3] Insee. *rtauargus dev_auto branch*. GitHub repository. 2025. URL: `https://github.com/InseeFrLab/rtauargus/tree/dev_auto`.