LAB: Correspondance Analysis

2025-03-16

M1 MIDS/MFA/LOGOS
Université Paris Cité
Année 2024
Course Homepage
Moodle

Besides the usual packages (`tidyverse`, ...), we shall require `FactoMineR` and related packages.

```
stopifnot(
  require(FactoMineR),
  require(factoextra),
  require(FactoInvestigate)
)
```

Correspondence Analysis

The `mortality` dataset

The goal is to investigate a possible link between age group and Cause of death. We work with dataset `mortality` from package `FactoMineR`

```
data("mortality", package = "FactoMineR")
```

```
#help(mortality)
```

> A data frame with 62 rows (the different Causes of death) and 18 columns. Each column corresponds to an age interval (15-24, 25-34, 35-44, 45-54, 55-64, 65-74, 75-84, 85-94, 95 and more) in a year. The 9 first columns correspond to data in 1979 and the 9 last columns to data in 2006. In each cell, the counts of deaths for a Cause of death in an age interval (in a year) is given.

Source Centre d'épidemiologie sur les Causes de décès médicales

See also EuroStat:

- Causes of death (hlth_cdeath) Reference Metadata in Single Integrated Metadata Structure (SIMS)
-

Question

Read the documentation of the `mortality` dataset. Is this a sample? an aggregated dataset?
If you consider `mortality` as an agregated dataset, can you figure out the organization of the sample `mortality` was built from?

Solution

The `mortality` dataset is an *aggregated* dataset. It has been built from two samples. Each sample was built from the collection of *death certificates* from one calendar year in France (years 1999 and 2006). From each death certificate, two categorical pieces of information were extracted: *age group* of the deceased and a *Cause of death*. Each sample was then grouped by *age group* and *Cause of death* and counts were computed. This defines a two-ways *contingency table* in *long* form. The contingency table in *wide* form is obtained by pivoting: pick column names from column *age group* and values from counts. Column *Cause of depth* provide row names.
The final form of the dataset is obtained by concatenating the two contingency tables along the second axis.

```
mortality <- mortality |>
    mutate(Cause = rownames(mortality)) |>
    mutate(Cause = factor(Cause)) |>
    relocate(Cause)
```

```
my_gt <- function(gt_tbl){
  gt_tbl |>
  tab_style(
    style = list(
      "font-variant: small-caps;"
    ),
    locations = cells_body(columns = Cause)
  ) |>
  gt::cols_align(
    align="left",
    columns=Cause
  )

}
```

```
mortality |>
    select(Cause, ends_with('(06)')) |>
    sample_n(10) |>
    gt::gt() |>
    my_gt()
```

Cause	15-24 (06)	25-34 (06)	35-44 (06)
Homicides	52	83	65
Malignant melanoma	9	43	127
Other genito-urinary diseases	1	1	7
Malignant tumour in other parts of the uterus	0	10	52
Events of undetermined intention	39	47	76
Other malignent tumours	137	234	609
Other diseases of the nervous system and sensory organs	167	214	455
Malignant tumour of the of the pancreas	3	5	101
Tuberculosis	3	5	11
Other external injury and poisoning	14	28	42

Elementary statistics and table wrangling

Before proceeding to Correspondence Analysis (CA), let us tidy up the table and draw some elementary plots.

Question

- Start by partially *pivoting* `mortality`, so as to obtain a tibble with columns `Cause`, `year`, while keeping all columns named after age groups (tidy up the data so as to obtain a tibble in partially long format).
- Use `rowwise()` and `sum(c_cross())` so as to compute the total number of deaths per `year` and `Cause` in column `total`. This allows to mimic `rowSums()` inside a pipeline. Column `grand_total` is computed using a *window* function over grouping by `Cause`.

Solution

```
mortality_long <- mortality  |>
  pivot_longer(
    cols=-Cause,
    cols_vary="slowest",
    names_to=c(".value", "year"),
    names_pattern="([\\w\\- ]*) \\(([0-9]{2})\\)"
  )  |>
  mutate(year=ifelse(year=='06', 2006, 1979)) |>
  rowwise() |>
  mutate(total_year=sum(c_across(-c(Cause, year)))) |>
  group_by(Cause) |>
  mutate(grand_total = sum(total_year)) |>
  ungroup()
```

```
mortality_long |>
 slice_sample(n=10) |>
 gt::gt() |>
 my_gt() |>
 gt::tab_caption("A sample of rows from Mortality table in long form")
```

Cause	year	15-24	25-34	35-44	45-54	55-64	65-7
Ischemic cardiomyopathy	2006	4	60	474	1688	2974	548
Other digestive conditions	1979	86	239	406	1080	1602	408
Other infectious diseases and parasites	2006	32	32	95	260	537	100
Complications in pregnancy and childbirth	1979	24	51	16	0	0	
Cerebrovascular disease	2006	35	75	311	902	1575	371
Rhumatoid arthritis and osteoarthritis	1979	2	5	6	22	45	20
Homicides	1979	92	116	115	65	50	3
Malignant ovarian tumour	2006	4	12	63	302	603	86
Malignant melanoma	1979	12	51	65	94	115	16
Viral hepatitis	2006	0	9	76	117	94	14

Solution

A truly tidy version of the dataset can be obtained from further pivoting.

```
mortality_tidy <- mortality_long |>
  pivot_longer(
    cols=-c(year,Cause,total_year, grand_total),
    cols_vary="slowest",
    names_to=c("age_range"),
    values_to=c("#deaths")
) |>
  mutate(age_range = factor(age_range, levels=sort(unique(age_range)),ordered=T))
```

```
mortality_tidy |>
  sample_n(5) |>
  gt::gt()
```

Cause	year	total_year	grand_total	age_range	#de
Congenital defects of the nervous system	2006	48	109	35-44	
Cerebrovascular disease	1979	66157	98795	35-44	
Unknown or unspecified causes	1979	14356	26192	15-24	
Homicides	2006	359	875	35-44	
Meningitis	1979	362	481	75-84	

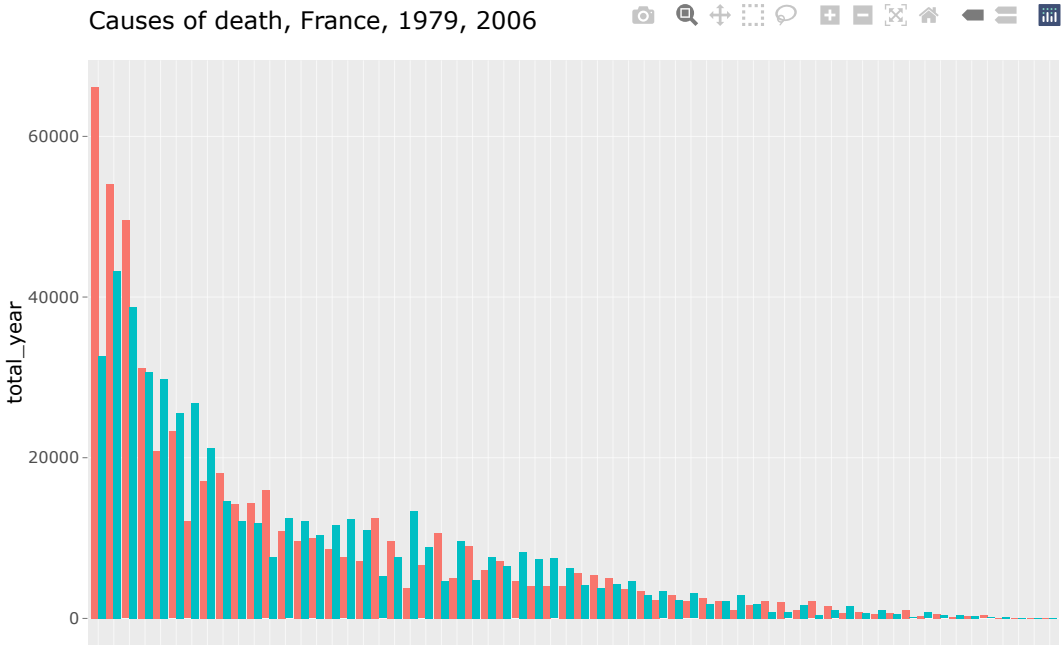
Question

Build a bar plot to display the importance of Causes of deaths in France in years 1979 and 2006

Solution

```
th <- theme_get()
(
mortality_long |>
  mutate(Cause=fct_reorder(Cause, desc(grand_total))) |>
  mutate(year=as_factor(year)) |>
  ggplot() +
  scale_fill_discrete() +
  aes(x=Cause,
      y=total_year,
      fill=year) +
  geom_col(position=position_dodge()) +
  theme(
    legend.position="none",
    axis.text.x=element_blank(), #remove x axis labels
    axis.ticks.x=element_blank(), #remove x axis ticks
  ) +
  labs(
    title = "Causes of death, France, 1979, 2006",
    subtitle= "Raw counts"
  ) +
  xlab(label=NULL)
) |>
  plotly::ggplotly()
```

```
PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is ir
```




```
oth <- theme_set(th)
```

Question

Compute and display the total number of deaths in France in years 1979 and 2006.

Solution

```
mortality_long |>
  group_by(year) |>
  summarise(total_deaths = sum(total_year)) |>
  gt::gt() |>
  gt::cols_label(
    year= "Year",
    total_deaths = "#Deaths") |>
  gt::tab_caption("Mortality in France")
```

Year	#Deaths
1979	529974
2006	510921

Question

Compute the marginal counts for each year (1979, 2006). Compare.

Solution

Counts have already been computed above.

```
mortality_long |>
  select(Cause, year, total_year, grand_total) |>
  pivot_wider(
    id_cols=c(Cause, grand_total),
    names_from = year,
    values_from = total_year) |>
  rename(Total=grand_total) |>
  arrange(desc(Total)) |>
  gt::gt() |>
  my_gt()
```

Cause	Total	1979
Cerebrovascular disease	98795	66157
Other heart disease	97297	54105
Ischemic cardiomyopathy	88338	49532
Other illnesses relating to circulation	61937	31218
Malignant tumour of the larynx, trachea, bronchus and lungs	50604	20840
Other malignent tumours	48809	23262
Other diseases of the nervous system and sensory organs	38891	12056
Other ill-defined symptoms and conditions	38330	17125
Other digestive conditions	32697	18092
Other respiratory ailments	26339	14197
Unknown or unspecified causes	26192	14356
Chronic liver disease	23596	15927

Correspondance Analysis

CA executive summary

- Start from a 2-way contingency table X with $\sum_{i,j} X_{i,j} = N$
- Normalize $P = \frac{1}{N}X$ (*correspondance matrix*)
- Let r (resp. c) be the row (resp. column) wise sums vector
- Let $D_r = \text{diag}(r)$ denote the diagonal matrix with row sums of P as coefficients
- Let $D_c = \text{diag}(c)$ denote the diagonal matrix with column sums of P as coefficients
- The *row profiles matrix* is $D_r^{-1} \times P$
- The *standardized residuals matrix* is $S = D_r^{-1/2} \times (P - rc^\top) \times D_c^{-1/2}$

CA consists in computing the SVD of the standardized residuals matrix $S = U \times D \times V^\top$

From the SVD, we get

- $D_r^{-1/2} \times U$ *standardized coordinates of rows*
- $D_c^{-1/2} \times V$ *standardized coordinates of columns*
- $D_r^{-1/2} \times U \times D$ *principal coordinates of rows*
- $D_c^{-1/2} \times V \times D$ *principal coordinates of columns*
- Squared singular values: the principal *inertia*

When calling `svd(.)`, the argument should be

$$D_r^{1/2} \times (D_r^{-1} \times P \times D_c^{-1} - \mathbf{I} \times \mathbf{I}^\top) \times D_c^{1/2} = D_r^{-1/2} \times (P - r \times c^\top) \times D_c^{-1/2}$$

CA and extended SVD

As

$$D_r^{-1} \times P \times D_c^{-1} - \mathbf{I}\mathbf{I}^\top = (D_r^{-1/2} \times U) \times D \times (D_c^{-1/2} \times V)^\top$$

$(D_r^{-1/2} \times U) \times D \times (D_c^{-1/2} \times V)^\top$ is the *extended SVD* of

$$D_r^{-1} \times P \times D_c^{-1} - \mathbf{I}\mathbf{I}^\top$$

with respect to D_r and D_c

Question

Perform CA on the two contingency tables.

You may use `FactoMineR::CA()`. It is interesting to compute the correspondence analysis in your own way, by preparing the matrix that is handled to `svd()` and returning a named list containing all relevant information.

> Do the Jedi and Sith build their own light sabers? Jedi do. It's a key part of the religion to have a kyber crystal close to you, to build the saber through the power of the force creating a blade unique and in tune with them

Solution

```
lst_ca <- list()

for (y in c('79', '06')) {
  lst_ca[[y]] <- mortality |>
    select(ends_with(glue('({y})'))) |>
    FactoMineR::CA(ncp=8, graph = F)
}
```

```
lst <- map(c('79', '06'),
             \(x) select(mortality, ends_with(glue('({x})'))) |>
             FactoMineR::CA(ncp=8, graph = F)
           )
```

Question

If you did use `FactoMineR::CA()`, explain the organization of the result.

Solution

The result of `FactoMineR::CA(...)` is a named and nested list with five elements:

`eig` a matrix/array containing enough information to build a screeplot.

`call` a list of 9, containing the call to `CA()`, an object of type `language`, telling (in principle) the user how `CA()` was called. However, this is a *quoted expression*. Here we need to guess the value of `y` in the calling environment understand what's going on.

```
lst_ca[[1]]$call$call
```

```
FactoMineR::CA(X = select(mortality, ends_with(glue("({y})"))),
    ncp = 8, graph = F)
```

Element `call` also contains the table margin distributions `marge.col` and `marge.row`. The truncation rank `ncp` (number of components) can be assigned before computing the SVD (default value is 5). Element X stores the contingency table that was effectively used for computing Correpondence Analysis.

`row` Information gathered from SVD to facilitate row profiles analysis.

`col` a list structured in the same way as element `row`. Used for column profiles analysis

`svd` a list of 3, just as the resuld of `svd()` containing the singular values, the left and right singular vectors of matrix ...

In principle, all relevant information can be gathered from components `svd`, `call.marge.row`, and `call.marge.col`.

Screeplots

Question

Draw screeplots. Why are they useful? Comment briefly.

Solution

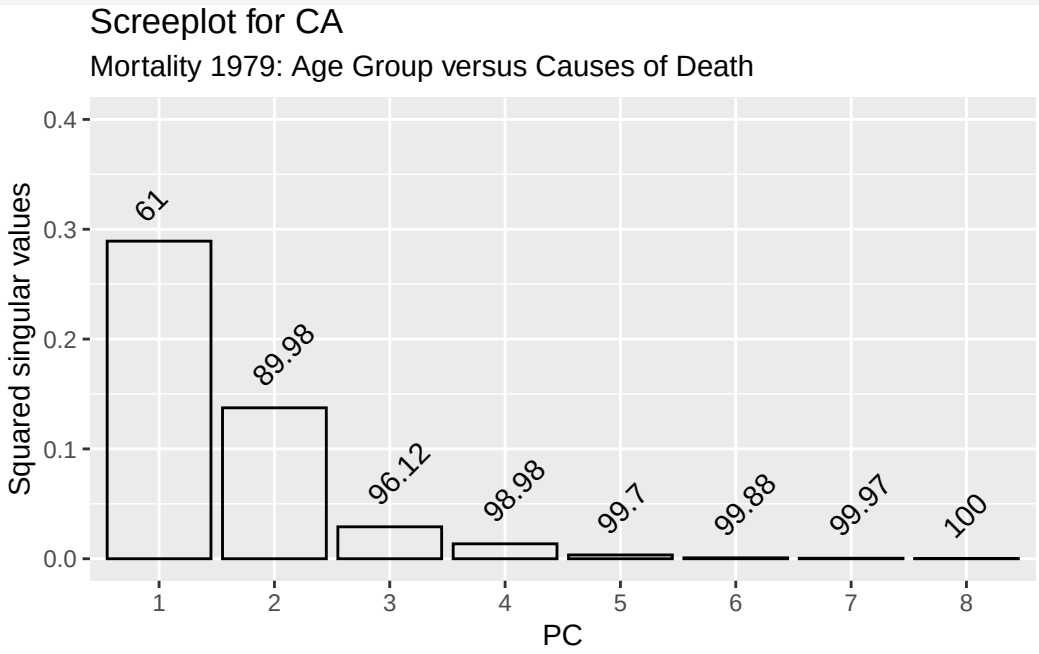
```
ca_79 <- lst_ca[[1]]

ca_79$eig |>
  as_tibble() |>
  mutate(across(where(is.numeric), ~ round(.x, digits=2))) |>
  gt::gt()
```

eigenvalue	percentage of variance	cumulative percentage of variance
0.29	61.00	61.00
0.14	28.98	89.98
0.03	6.13	96.12
0.01	2.86	98.98
0.00	0.73	99.70
0.00	0.17	99.88
0.00	0.09	99.97
0.00	0.03	100.00

Solution

Screeplot

```
ca_79$eig |>
  as_tibble() |>
  rownames_to_column(var="PC") |>
  rename(percent=eigenvalue, cumulative=`cumulative percentage of variance`) |>
  ggplot() +
  aes(x=PC, y=percent, label=round(cumulative,2)) +                    ③
  geom_text(angle=45, vjust=-1, hjust=-0.1) +
  geom_col(fill=NA, colour="black") +                                  ②
  ylab("Squared singular values") +
  ylim(c(0, .4)) +
  labs(
    title="Screeplot for CA",
    subtitle = "Mortality 1979: Age Group versus Causes of Death"
  )
```



Row profiles analysis

Question

Perform row profiles analysis.
What are the classical plots? How can you build them from the output of `FactoMiner::CA`?
Build the table of row contributions (the so-called $\cos^2$)

Solution

Attribute `row` of objects of class `CA` (exported from `FactoMineR`) is the starting point of any row profiles analysis.

```
ca_79_row <- ca_79$row
```

Attribute `row` is a named list made of 4 components.

Solution

inertia a numerical vector with length matching the number of rows of `coord`, `contrib` and `cos2`.
Inertia is the way CA measures variation between row profiles. Total inertia is the χ^2 statistic divided by sample size.

Row inertia can be obtained by multiplying the row marginal probability by the squared Euclidean norm of the row in the principal coordinate matrix.

```
with (ca_79_row,
  sum(abs(r* (rowSums(coord^2)) - inertia))
)
```

```
[1] 1.877449e-16
```

Solution

cos2 Coefficients of matrix `cos2` are the share of row inertia from the corresponding cell in `coord`

```
with (ca_79_row,
  norm((diag(r/inertia) %*% coord^2) - cos2, type='F')
)
```

```
[1] 5.432216e-16
```

Solution

`contrib`

Not too surprisingly, `coord`, `contrib`, and `cos2` share the same row names and column names.

```
sum(ca_79$call$X)
```

```
[1] 529974
```

```
sum((rowSums(ca_79$call$X)/sum(ca_79$call$X) - r)^2)
```

```
[1] 6.311339e-35
```

The Row Profiles are the rows of matrix `R` below

```
P <- as.matrix(with(ca_79$call, Xtot/N))
coord <- ca_79_row$coord
inertia <- ca_79_row$inertia

r <- ca_79$call$marge.row
c <- colSums(P)

n <- nrow(P)
p <- ncol(P)

R <- diag(r^(-1)) %*% P

Q <- R - matrix(1, nrow = n, ncol = n) %*% P
```

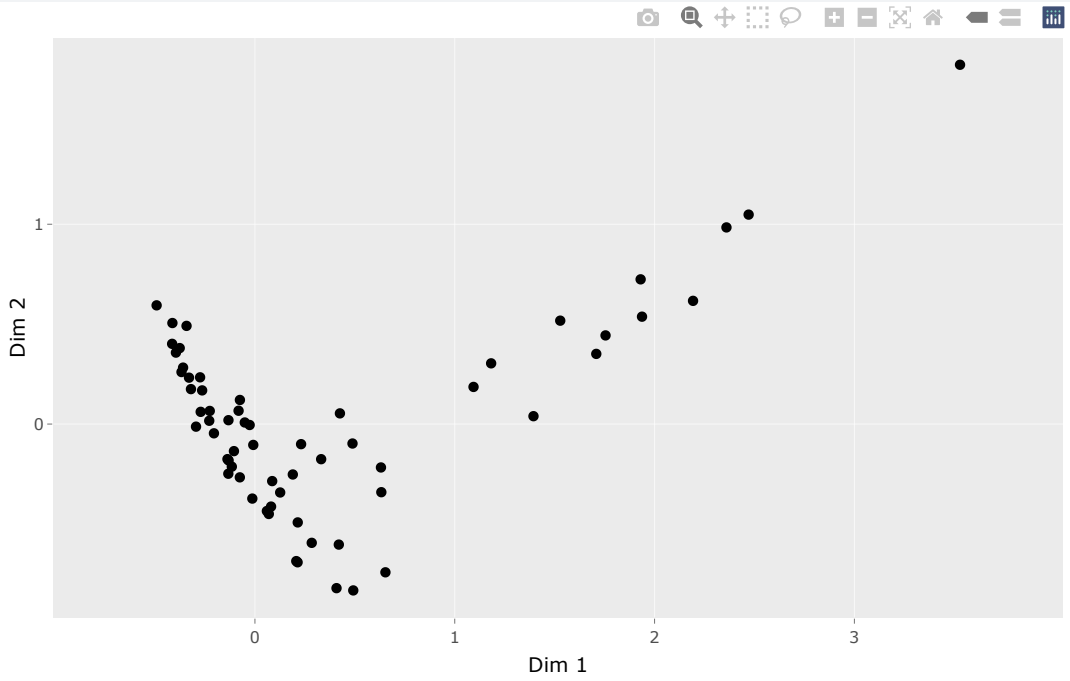
```
M <- diag(r^(-1)) %*% P %*% diag(c^(-1)) - matrix(1, nrow=n, ncol=p)

n * norm(diag(r^(1/2)) %*% M %*% diag(c^(1/2)), type = "F")^2
```

```
[1] 29.39279
```

Solution

We can now display a scatterplot from component `coord`. This is called a *Row Plot*.

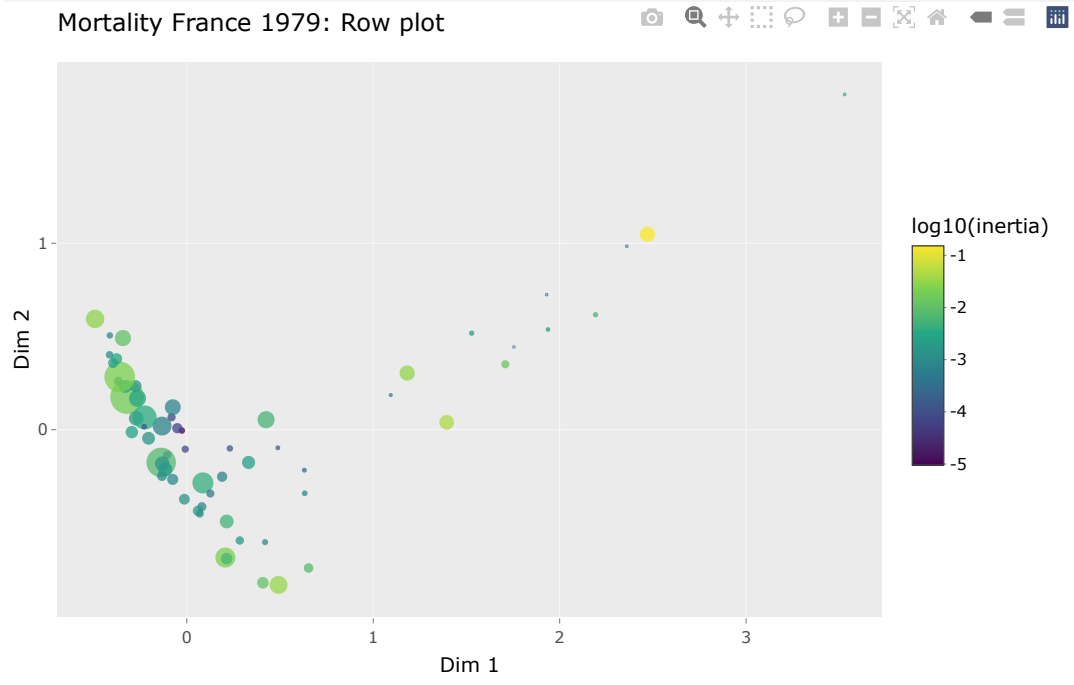
```
p_scat <-  (
  prep_rows |>
    ggplot() +
    aes(x=`Dim 1`, y=`Dim 2`, label=name) +
    geom_point() +
    coord_fixed()
  )

p_scat |> plotly::ggplotly()
```



Solution

With little effort, it is possible to scale the points so as to tell the reader the relative numerical importance of each Cause of death. Coloring/filling the points using *inertia* also helps: high inertia rows match light-colored points.

```
ppp <- prep_rows |>
    ggplot() +
    aes(x=`Dim 1`,
        y=`Dim 2`,
        label=name,
        size=prop,
        fill=log10(inertia),
        color=log10(inertia)) +
    geom_point(alpha=0.75) +
    scale_size_area() +
    coord_fixed() +
    scale_fill_viridis_c(aesthetics=c("fill", "color"),
                         guide="colorbar",
                         direction = 1) +
    ggtitle(
      "Mortality France 1979: Row plot"
    )

ppp |> plotly::ggplotly()
```




```
# (ca_79$row)$contrib
```

Question

Plot the result of row profile analysis using `plot.CA` from `FactoMineR`.

Question

Perform column profiles analysis

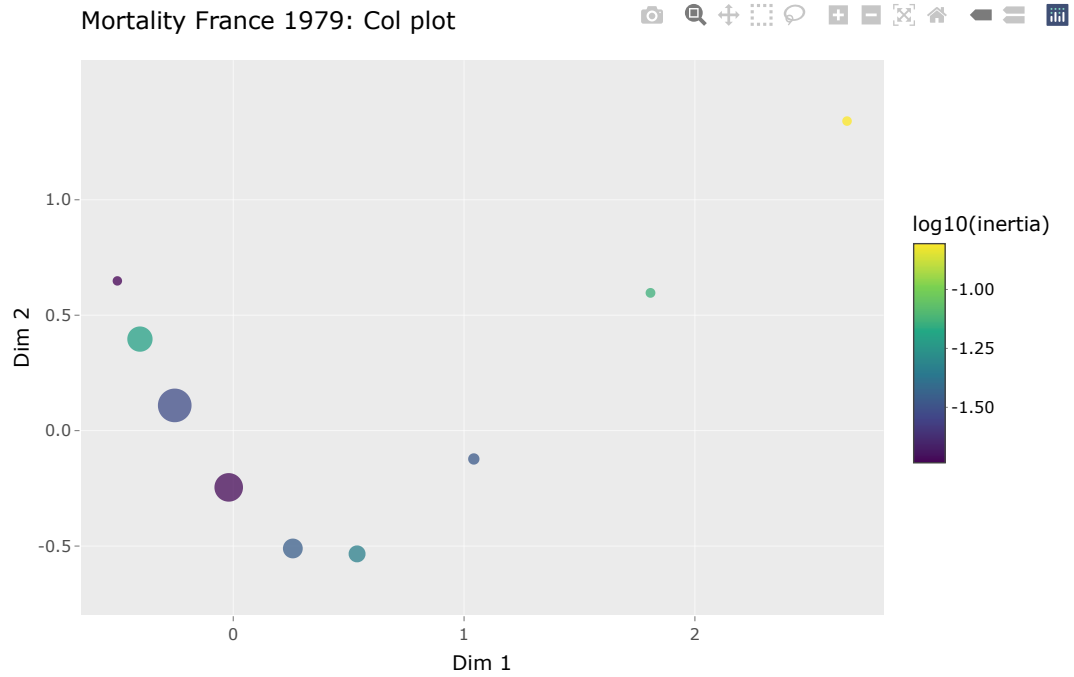
```
names(ca_79_row)
```

```
[1] "coord"   "contrib" "cos2"    "inertia"
```

Solution

```
age_group_names <-  str_match(rownames(ca_79$col$coord), '([\\w \\-]*) \\(79\\)')[,2]

prep_cols <- ca_79$col$coord |>
  as_tibble() |>
  mutate(name= age_group_names) |>
  relocate(name) |>
  mutate(prop=c, inertia=ca_79$col$inertia)
```

```
(
  prep_cols |>
    ggplot() +
    aes(x=`Dim 1`,
        y=`Dim 2`,
        label=name,
        size=prop,
        fill=log10(inertia),
        color=log10(inertia)) +
    geom_point(alpha=0.75) +
    scale_size_area() +
    coord_fixed() +
    scale_fill_viridis_c(aesthetics=c("fill", "color"),direction = 1) +
    ggtitle(
      "Mortality France 1979: Col plot"
    )) |> plotly::ggplotly()
```



Symmetric plots

> **Question**
>
> Build the symmetric plots (biplots) for correspondence analysis of Mortalitity data

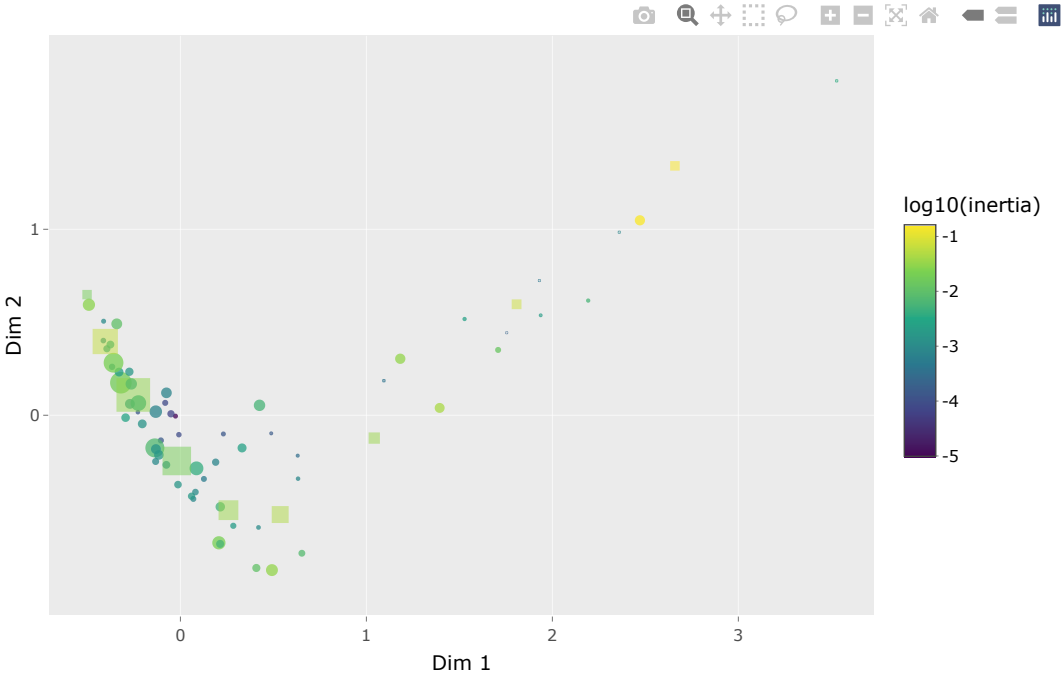
From the shelf

```
plot.CA(ca_79)
```

Solution

```
(
prep_rows |>
    ggplot() +
    aes(x=`Dim 1`,
        y=`Dim 2`,
        label=name,
        size=prop,
        fill=log10(inertia),
        color=log10(inertia)) +
    geom_point(alpha=0.75) +
    scale_size_area() +
    coord_fixed() +
    scale_fill_viridis_c(aesthetics=c("fill", "color"),direction = 1) +
    geom_point(data = prep_cols,
      aes(x=`Dim 1`,
        y=`Dim 2`,
        label=name,
        size=prop,
        fill=log10(inertia),
        color=log10(inertia)
      ),
      shape="square",
      alpha=.5,
    )
) |> plotly::ggplotly()
```

```
Warning in geom_point(data = prep_cols, aes(x = `Dim 1`, y = `Dim 2`, label =
name, : Ignoring unknown aesthetics: label
```

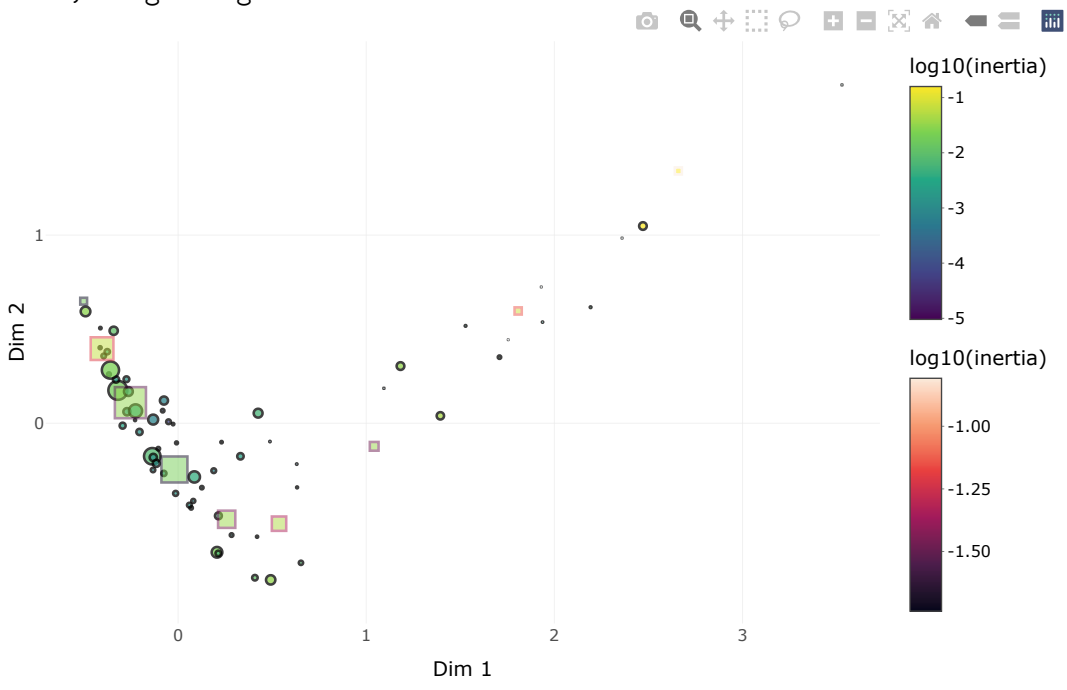


Solution

It is convenient to use distinct color scales for rows and columns.

```
(
prep_rows |>
    ggplot() +
    scale_size_area() +
    coord_fixed() +
    aes(x=`Dim 1`,
        y=`Dim 2`,
        text=name,
        size=prop,
        fill=log10(inertia)) +
    geom_point(alpha=0.75) +
    scale_fill_viridis_c(option="D") +
    geom_point(data = prep_cols,
      aes(x=`Dim 1`,
        y=`Dim 2`,
        text=name,
        size=prop,
        color=log10(inertia)
      ),
      shape="square",
      alpha=.5,
    ) +
    scale_color_viridis_c(option="F") +
    theme_minimal(
    )
)  |> plotly::ggplotly()
```

```
Warning in geom_point(data = prep_cols, aes(x = `Dim 1`, y = `Dim 2`, text =
name, : Ignoring unknown aesthetics: text
```



Mosaicplots

Question

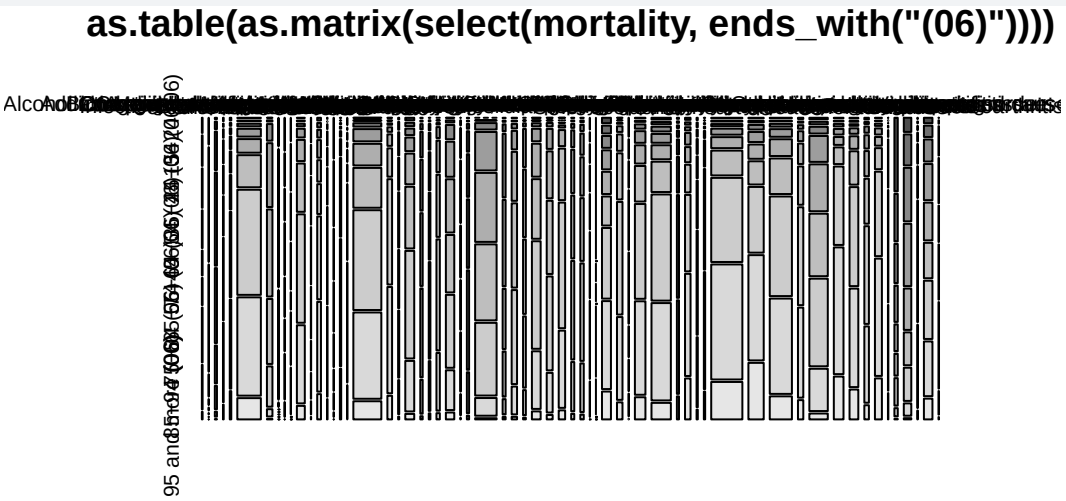
Mosaic plots provide an alternative way of exploring contingency tables. They are particularly handy when handling 2-way contingency tables.
Draw mosaic plots for the two contingency tables living inside `mortality` datasets.

Solution

```
mortality |>
  select(ends_with('(06)')) |>
  chisq.test() |>
  broom::glance()
```

```
Warning in chisq.test(select(mortality, ends_with("(06)"))): Chi-squared
approximation may be incorrect
# A tibble: 1 x 4
  statistic p.value parameter method
      <dbl>   <dbl>     <int> <chr>
1   229784.       0       488 Pearson's Chi-squared test
```

```
mortality |>
  select(ends_with('(06)')) |>
  as.matrix() |>
  as.table() |>
  mosaicplot(color = T)
```



Question

Are you able to deliver an interpretation of this Correspondence Analysis?

Hierarchical clusetring of row profiles

Question

Build the standardized matrix for row profiles analysis. Compute the pairwise distance matrix using the χ^2 distances. Should you work centered row profiles?

We use the weighted ℓ_2 distances defined by the product of the two marginal distributions. The squared distance between the conditional probabilities defined by rows a and a' is

$$\sum_b \frac{\left(N_{a,b}/N_{a,\cdot} - N_{a',b}/N_{a',\cdot}\right)^2}{N_{\cdot,b}/N}$$

The ℓ_2 distance between the rows of the principal coordinates matrix `row$coord` coincides since they are all centered and normalized with respect to $(N_{\cdot,b}/N)$.

```
dist_Causes_79 <- ca_79$row$coord[,1:8] |>
  dist()
```

```
hc_79 <- hclust(dist_Causes_79, method = "single")
```

```
stopifnot(
  require(ggdendro),
  require(dendextend),
  require(sloop)
)
```

The instance of `hclust` is transformed into a an object of class `dendro`. Class `dendro` is equipped with a variety of functions/methods for analyzing, visualizing, and exploiting the result of `hclust()`.

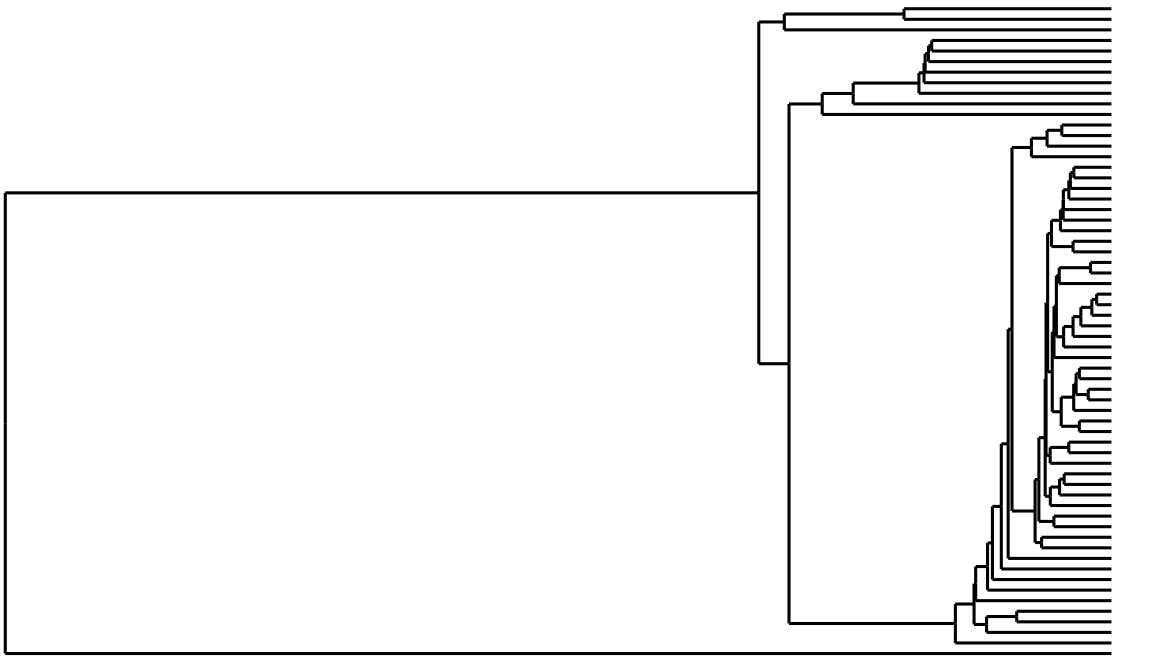
```
dendro_79 <- dendro_data(hc_79)
```

```
class(dendro_79)
```

```
[1] "dendro"
```

```
(
dendro_79 |>
  ggdendrogram(
    leaf_labels = T,
    rotate = T) +
  ggdendro::theme_dendro() +
  scale_y_reverse()
  ) |> plotly::ggplotly()
```

```
Scale for y is already present.
Adding another scale for y, which will replace the existing scale.
```



Question

Perform hierarchical clustering of row profiles with method/linkage `"single"`. Check the definition of the method. Did you know the underlying algorithm? If yes, in which context did you get acquainted with this algorithm?

Question

Choose the number of classes (provide justification).

Question

Can you explain the size of the different classes in the partition?

Atypical row profiles

Question

Row profiles that do not belong to the majority class are called *atypical*.

1. Compute the share of inertia of atypical row profiles.
2. Draw a symmetric plot (biplot) outlining the atypical row profiles.

Investigating independence/association

Question

1. Calculate the theoretical population table for `deces`. Do you possible to carry out a chi-squared test?
2. Perform a hierarchical classification of the line profiles into two classes.
3. Merge the rows of `deces` corresponding to the same class (you can use the the `tapply` function), and perform a chi-square test. chi-square test. What's the conclusion?
4. Why is it more advantageous to carry out this grouping into two classes compared to arbitrarily grouping two classes, in order to prove the dependence between these two variables?

About the "average profile"

Question

1. Represent individuals from the majority class. Do they all seem to you to correspond to an average profile?
2. Try to explain this phenomenon considering the way in which hierarchical classification uses the Single Linkage method.

Caveat

The `mortality` dataset should be taken with grain of salt. Assigning a single *Cause* to every death is not a trivial task. It is even questionable: if somebody dies from some infection beCause she could not be cured using an available drug due to another preexisting pathology, who is the culprit?