

Formation-R-perfectionnement

Module Cartographie

Sommaire

- 1 Les packages**
- 2 Le package tmap**
- 3 Cas pratique : réalisation pas à pas d'une première carte**
- 4 Classes**
- 5 Facettes**
- 6 Cartogramme**
- 7 Grilles**
- 8 Bibliographie**
- 9 Exercices**

0.1 Avant-propos

Ce diaporama de formation a été rédigé dans le but d'être le support visuel des formations dispensées au **SSM Agriculture**.

Ces formations s'adressent à des agents qui ont suivi la **formation R initialisation**.

0.2 Avant-propos

Elles sont données en présentiel sur une durée **de trois journée**, les modules de cette formation sont ajustables suivants le choix des agents.

Champ couvert par cette formation

Ce support couvre le module cartographie dans l'environnement du Minsitère.

Pour information, Les Modules de la formation R-perfectionnement sont:

- 01 - Module Rappels
- 02 - Module Fonctions
- 03 - Module Quarto
- 04 - Module Création de graphiques avec ggplot2
- 05 - Module Cartes statiques et interactives
- 06 - Module Parquet
- 07 - Module Initiation à l'écriture d'applications Shiny

Ils sont orientés pour être utile aux agents du SSM Agriculture et se concentrent sur une utilisation de R via [RStudio](#) qui est mise à disposition des agents sur la plateforme interne Cerise basée sur RStudio Workbench.

1 Les packages





1.1 Les packages

Les packages utilisés dans ce module sont :

- {sf} : ensemble de fonctions pour gérer les objets spatiaux (importation, traitement, exportation)
- {tmap} : réalisation simple de cartes
- {RcolorBrewer} : palettes de couleurs
- ...

Autres possibilités pour la carto avec : {ggplot2} + {ggspatial}, {maps}, {leaflet} ou {mapview},...

```
1 library(tidyverse)
2 library(sf)
3 library(tmap)
4 library(janitor)
5 library(ragg)
6 # library(systemfonts)
7 # library(textshaping)
8 options(outDec = ".")
```

2 Le package tmap



2.1 Le package tmap

- Depuis 2014
- S'inspire de la grammaire des graphiques (`{ggplot2}`)
- Cartes statiques ou interactives
- Site officiel : <https://r-tmap.github.io/tmap/>

Les différentes étapes de la réalisation d'une carte :

- Import du fond de carte et des données
- Choix de la variable à représenter et construction éventuelle des classes
- Représentation graphique
- Habillage du graphique avec titre, légende, sources, etc.

3 Cas pratique : réalisation pas à pas d'une première carte

3.1 Cas pratique : réalisation pas à pas d'une première carte

Objectif : carte départementale métropolitaine des parts d'exploitations ayant au moins une parcelle conduite en agriculture biologique et du nombre d'exploitations (source : RA 2020)

3.2 Import d'un fond de carte

Ouverture du fond de carte

La fonction `read_sf()` permet d'ouvrir des fonds carto au format `.TAB` (mapinfo), `.shp` (shapefile), `*.gpkg` (geopackage) et bien d'autres (PostGIS).

```
1 gpkg <- "~/CERISE/03-Espace-de-Diffusion/000_Référentiels/0040_Geo/IGN/adminexpress/adminexpress_cog_simpl_000_2023.gpkg"
2
3 st_layers(gpkg)
4 dep <- read_sf(gpkg, layer = "departement") |>
5   filter(insee_reg > "06") |>
6   st_transform("EPSG:2154")
```

```
Driver: GPKG
Available layers:
  layer_name  geometry_type features fields crs_name
1  commune     Multi Polygon  34947    13  WGS 84
2  departement  Multi Polygon   103      4  WGS 84
3   region     Multi Polygon    19      3  WGS 84
4   cheflieu      Point       34947    7  WGS 84
5     epci     Multi Polygon   1266      4  WGS 84
6  commune_int Multi Line String 102371    0  WGS 84
7 departement_int Multi Line String   238      0  WGS 84
8   region_int Multi Line String    23      0  WGS 84
9     epci_int Multi Line String   3527      0  WGS 84
```

L'objet combine des polygones (le contour des départements ici) et des données regroupées dans un objet de type `data.frame`.

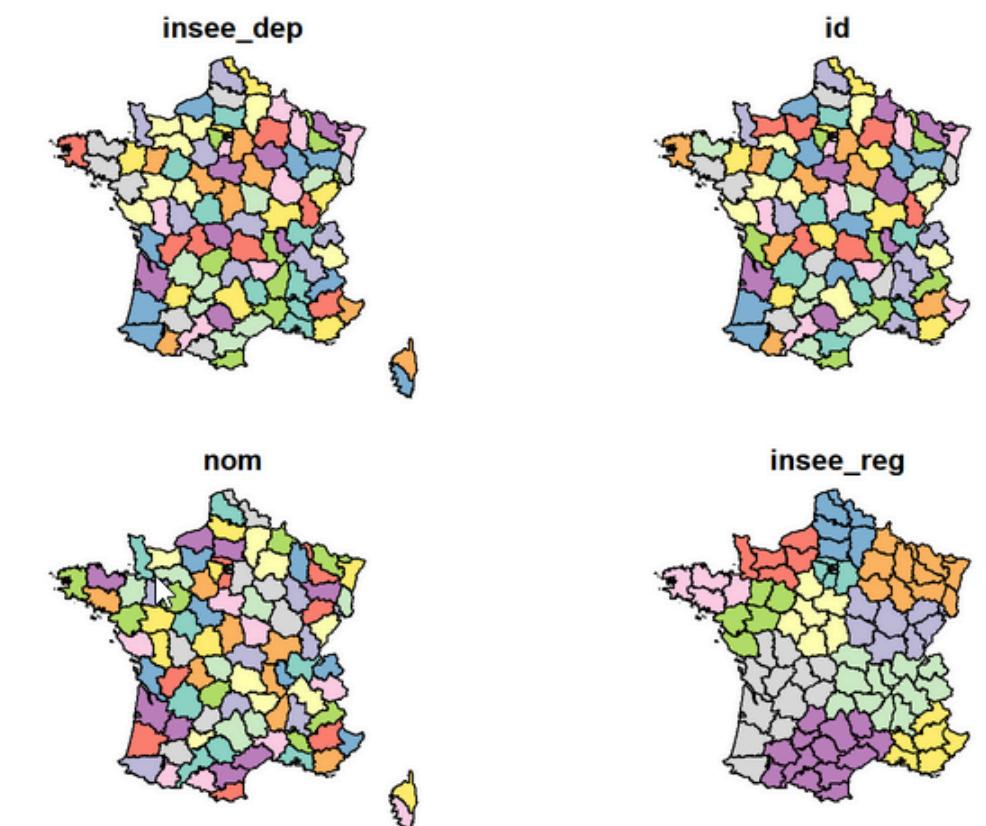
Ce data.frame se comporte comme n'importe quel autre (même utilisation des fonctions `str()`, `names()`...).

```
1 head(dep)
```

insee_dep	id	nom	insee_reg	geom
71	DEPARTEM_FXX_00000000072	Saône-et-Loire	27	MULTIPOLYGON (((779854.6631...
27	DEPARTEM_FXX_00000000028	Eure	28	MULTIPOLYGON (((509451.668...
08	DEPARTEM_FXX_00000000010	Ardennes	44	MULTIPOLYGON (((855756.269...
70	DEPARTEM_FXX_00000000071	Haute-Saône	27	MULTIPOLYGON (((894833.367...
01	DEPARTEM_FXX_00000000001	Ain	84	MULTIPOLYGON (((917787.265...
88	DEPARTEM_FXX_00000000089	Vosges	44	MULTIPOLYGON (((905286.368...

`plot()` permet d'afficher le fond spatial pour toutes les variables dans la fenêtre graphique.

```
1 plot(dep)
```



Jointure

```

1 exp_dep <- read_rds("~/CERISE/03-Espace-de-Diffusion/030_Structures_exploitations/3020_Recensements/RA_2020/01_BASES DIFFUSION RA2020/DEF_240"
2   as_tibble(.name_repair = make_clean_names) |>
3   filter(champ_geo == "1") |>
4   group_by(siege_dep) |>
5   summarise(n_exp = n(),
6             n_exp_bio = sum(bio_fil, na.rm = TRUE)) |>
7   mutate(part_exp_bio = n_exp_bio / n_exp * 100)
8
9 head(exp_dep)
10
11 bio <- dep %>%
12   left_join(exp_dep, by = c("insee_dep" = "siege_dep"))

```

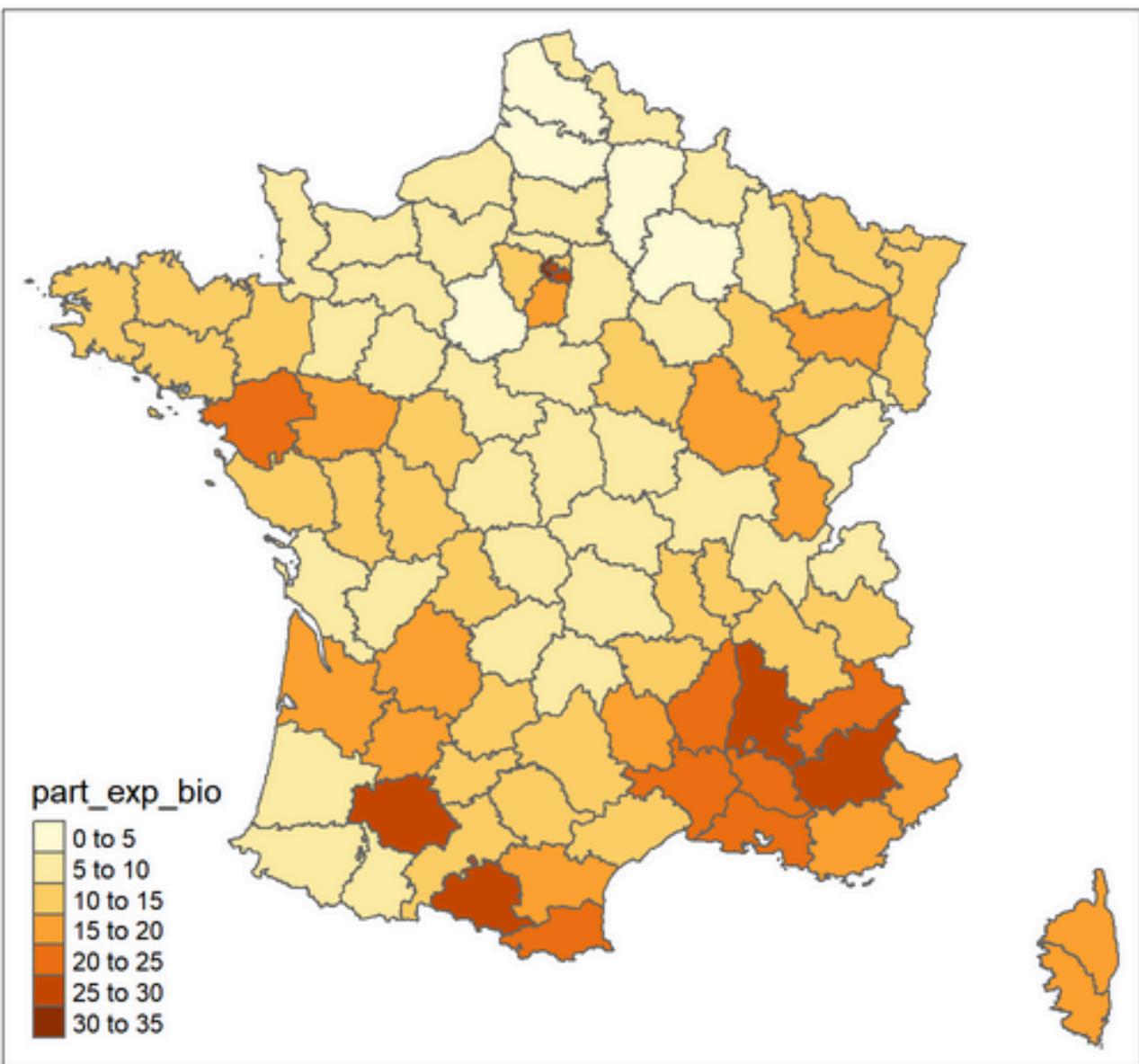
siege_dep	n_exp	n_exp_bio	part_exp_bio
01	3017	301	9.976798
02	4472	221	4.941860
03	4356	328	7.529844
04	2064	520	25.193798
05	1646	367	22.296476
06	1192	235	19.714765



*Liberté
Égalité
Fraternité*

3.3 Carte choroplète avec tm_polygons

```
1 tm_shape(bio) +  
2   tm_polygons("part_exp_bio")
```

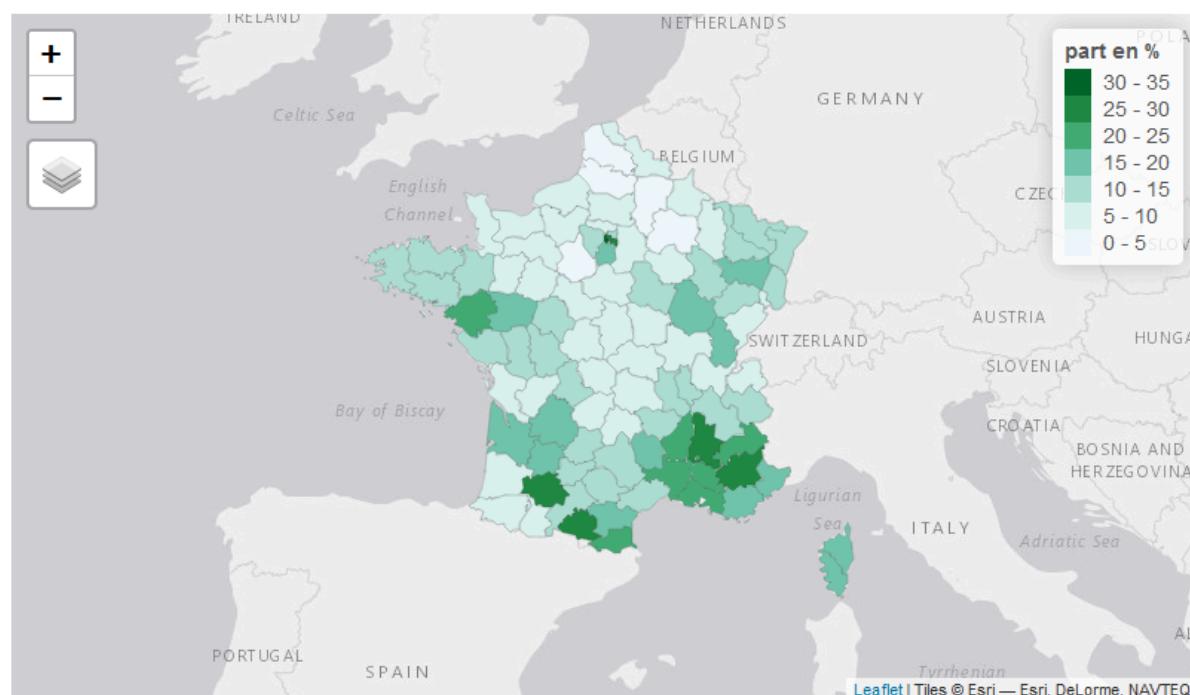


3.4 Carte interactive

```

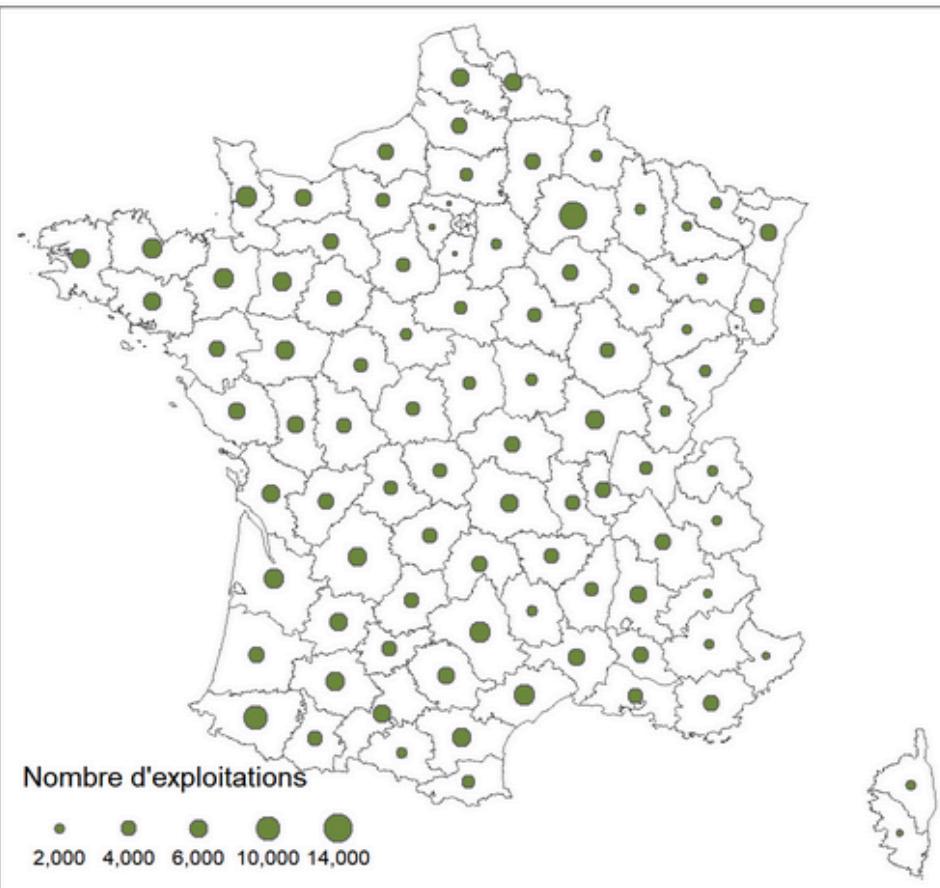
1 tmap_mode("view")
2
3 tm_shape(bio) +
4   tm_polygons("part_exp_bio",
5     style = "pretty",
6     n = 5,
7     title = "part en %",
8     palette = "BuGn",
9     border.col = "grey30",
10    lwd = 0.25,
11    legend.reverse = TRUE,
12    legend.format = list(text.separator = " - "))

```



3.5 Carte à symboles proportionnels

```
1 tmap_mode("plot")
2
3 tm_shape(bio) +
4   tm_borders(col = "grey30",
5             lwd = 0.25) +
6   tm_symbols(size = "n_exp",
7              col = "darkolivegreen4",
8              title.size = "Nombre d'exploitations")
```

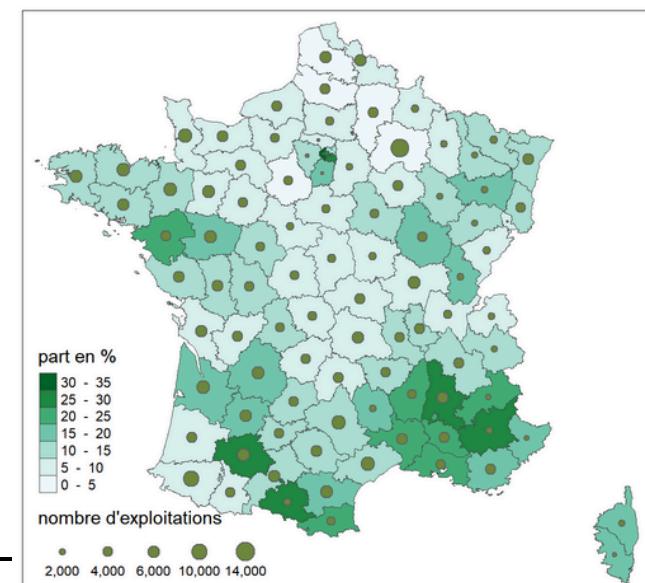


3.6 Carte choroplète + symboles proportionnels

```

1 tm_shape(bio) +
2   tm_polygons(
3     "part_exp_bio",
4     style = "pretty",
5     n = 5,
6     title = "part en %",
7     palette = "BuGn",
8     border.col = "grey30",
9     lwd = 0.25,
10    legend.reverse = TRUE,
11    legend.format = list(text.separator = " - "))
12  tm_symbols(size = "n_exp",
13    col = "darkolivegreen4",
14    title.size = "nombre d'exploitations")

```



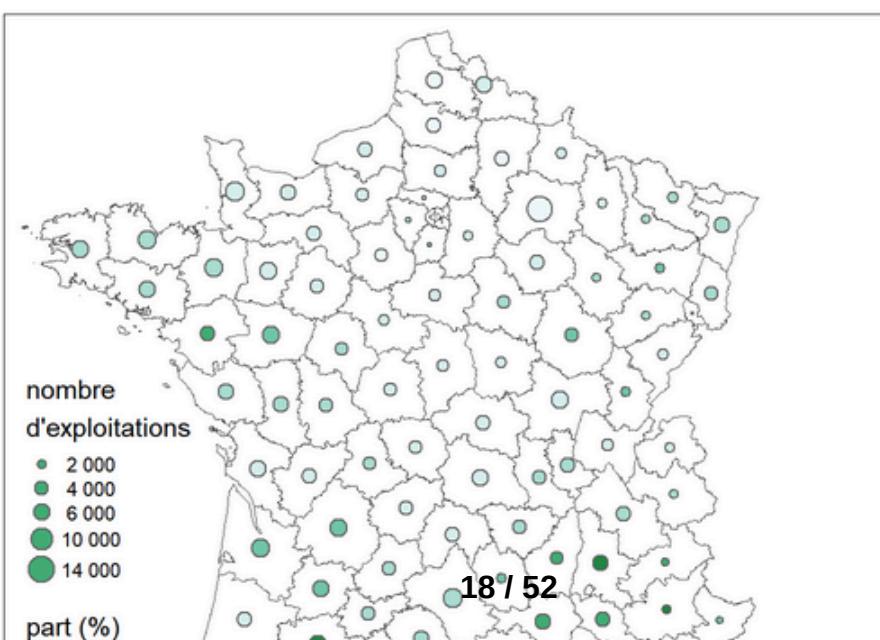
3.7 Variante

Une alternative consiste à faire varier la taille + la couleur du cercle : les couleurs des classes de la variable représentée en aplats sont placées dans les ronds proportionnels

```

1 m <- tm_shape(bio) +
2   tm_borders(col = "grey30",
3               lwd = 0.25,) +
4   tm_symbols(size = "n_exp",
5               col = "part_exp_bio",
6               palette = "BuGn",
7               title.col = "part (%)",
8               title.size = "nombre\nd'exploitations",
9               legend.format = list(fun = \x) format(x, big.mark = " "),
10              text.separator = " - "),
11               legend.col.reverse = TRUE,
12               legend.size.is.portrait = TRUE)

```



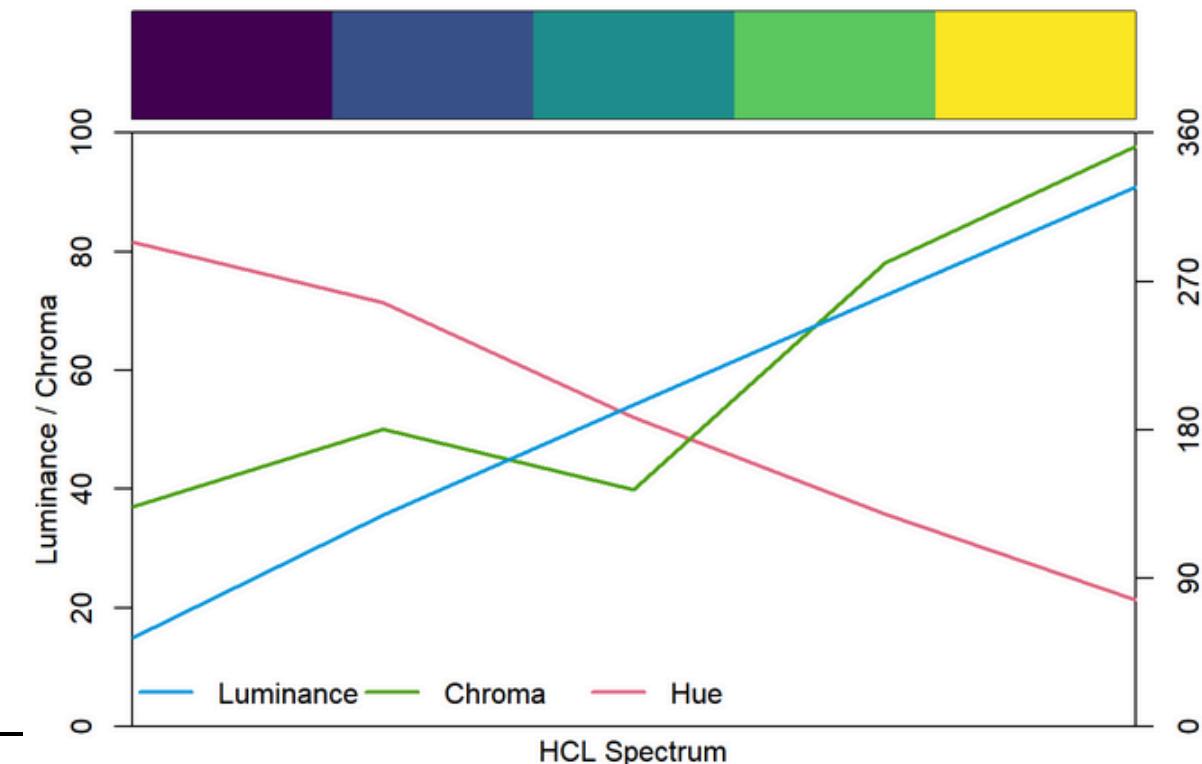
3.8 Couleurs et palettes

Les couleurs sont à préciser dans l'argument palette de la fonction `tm_polygons()`. Il faut autant de couleurs qu'il y a de classes.

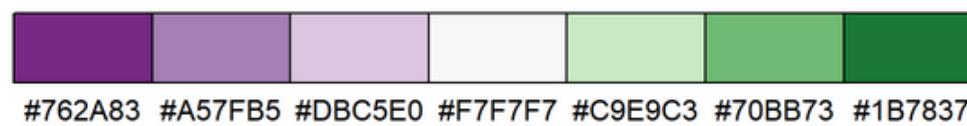
Lien utile : [Colors in R](#). Ce document présente l'ensemble des couleurs disponibles sous R

La fonction `tmaptools::palette_explorer()` permet de visualiser des palettes de couleurs

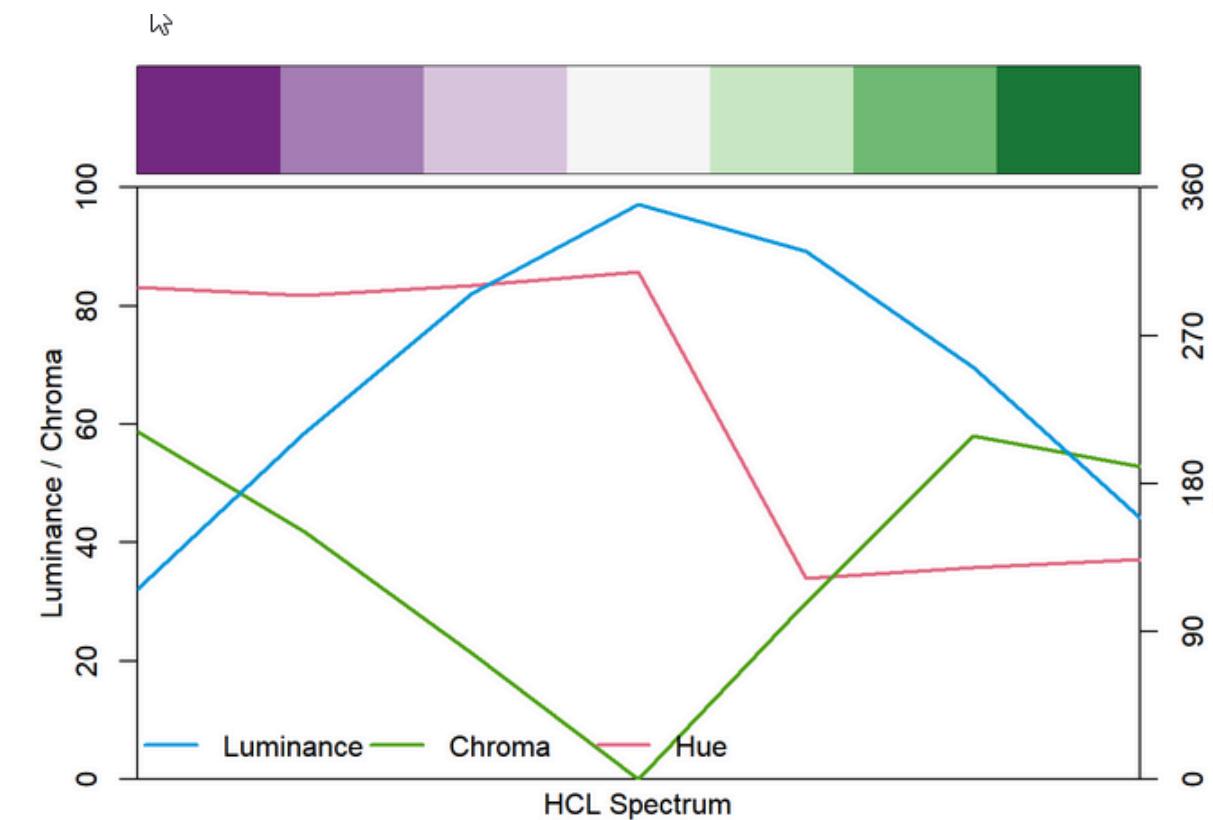
```
1 pal_vir <- viridisLite::viridis(5)
2 colorspace::specplot(pal_vir)
```



```
1 pal_div <- tmaptools::get_brewer_pal("PRGn", n = 7)
```



```
1 colorspace::specplot(pal_div)
```



```
1 library(RColorBrewer)  
2 head(brewer.pal.info)
```

	maxcolors	category	colorblind
BrBG	11	div	TRUE
PiYG	11	div	TRUE
PRGn	11	div	TRUE
PuOr	11	div	TRUE
RdBu	11	div	TRUE
RdGy	11	div	FALSE

[Code](#)

```
{.' .r} str(brewer.pal.info)

'data.frame':   35 obs. of  3 variables:
 $ maxcolors : num  11 11 11 11 11 11 11 11 11 8 ...
 $ category  : chr  "div" "div" "div" "div" ...
 $ colorblind: logi  TRUE TRUE TRUE TRUE TRUE FALSE ...

```



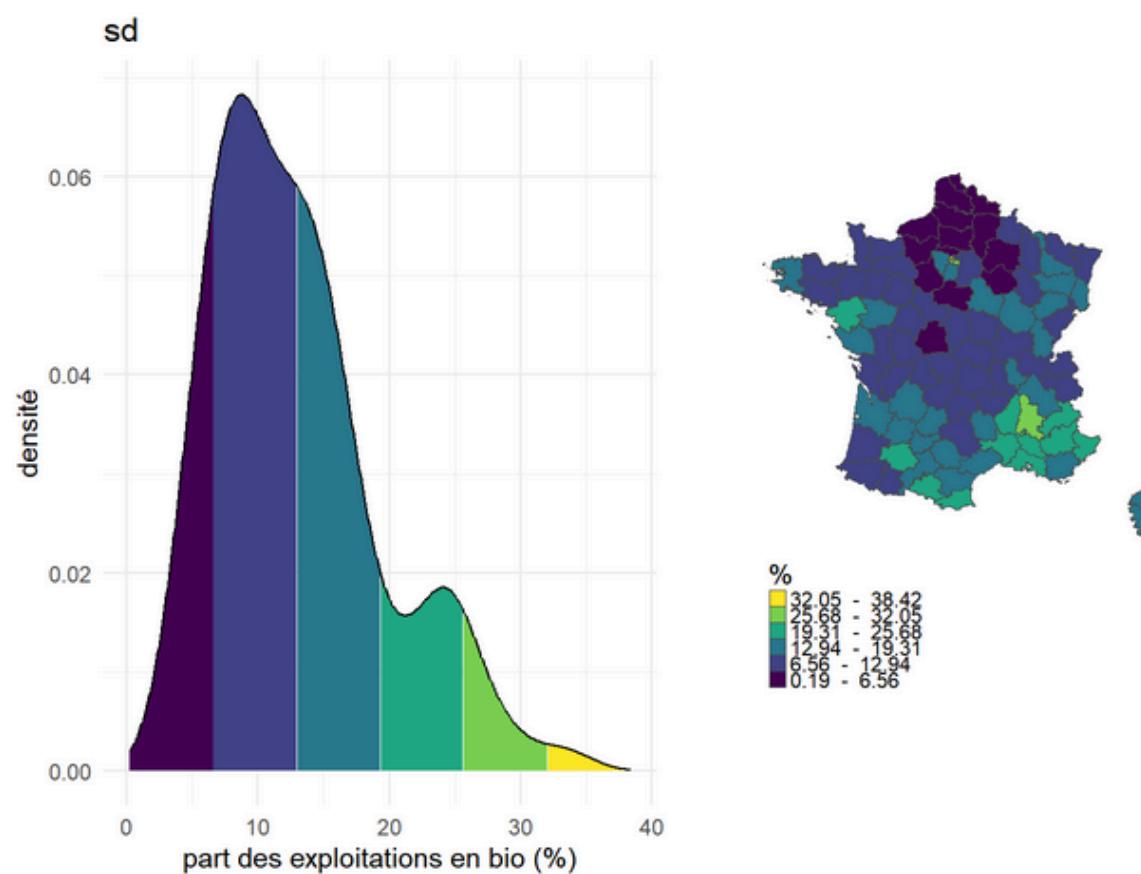
4 Classes

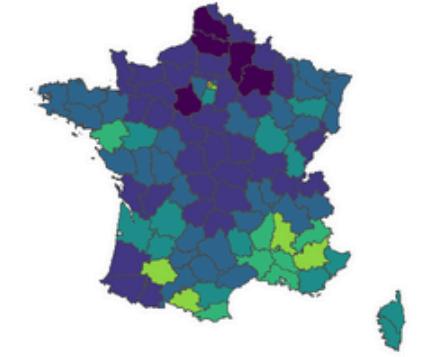
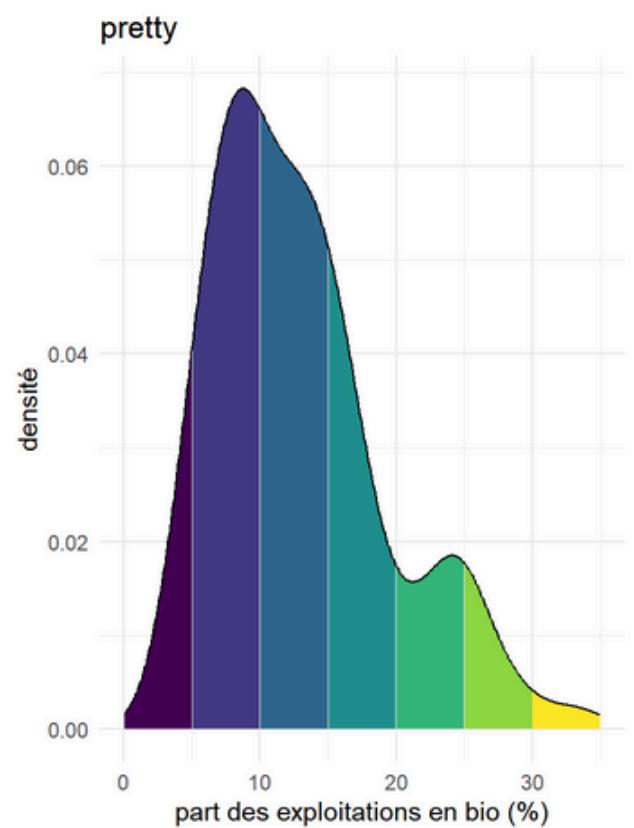
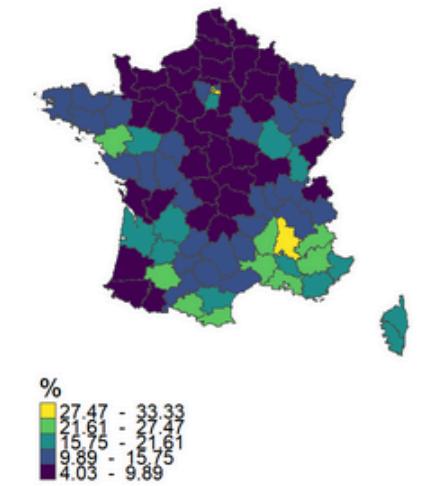
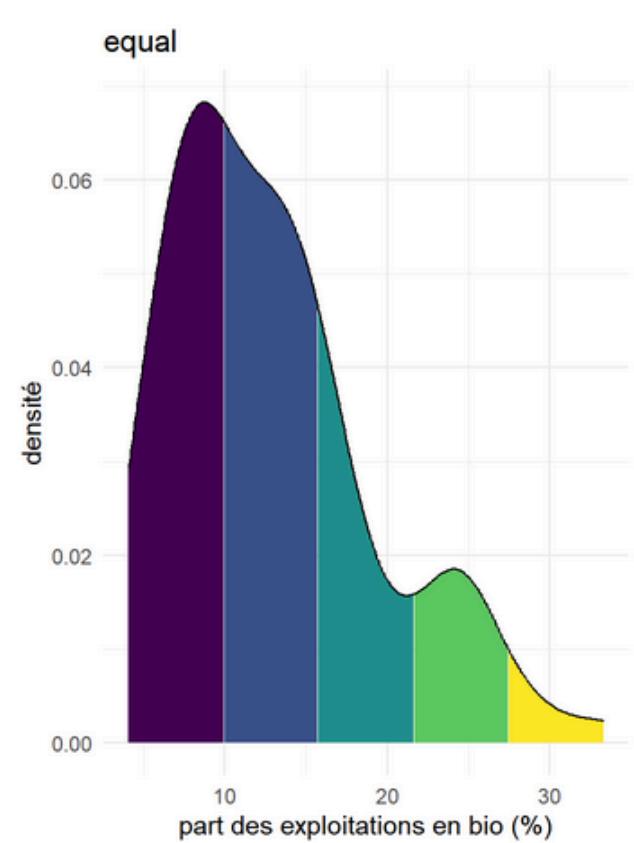
4.1 Construction des classes → méthode automatique

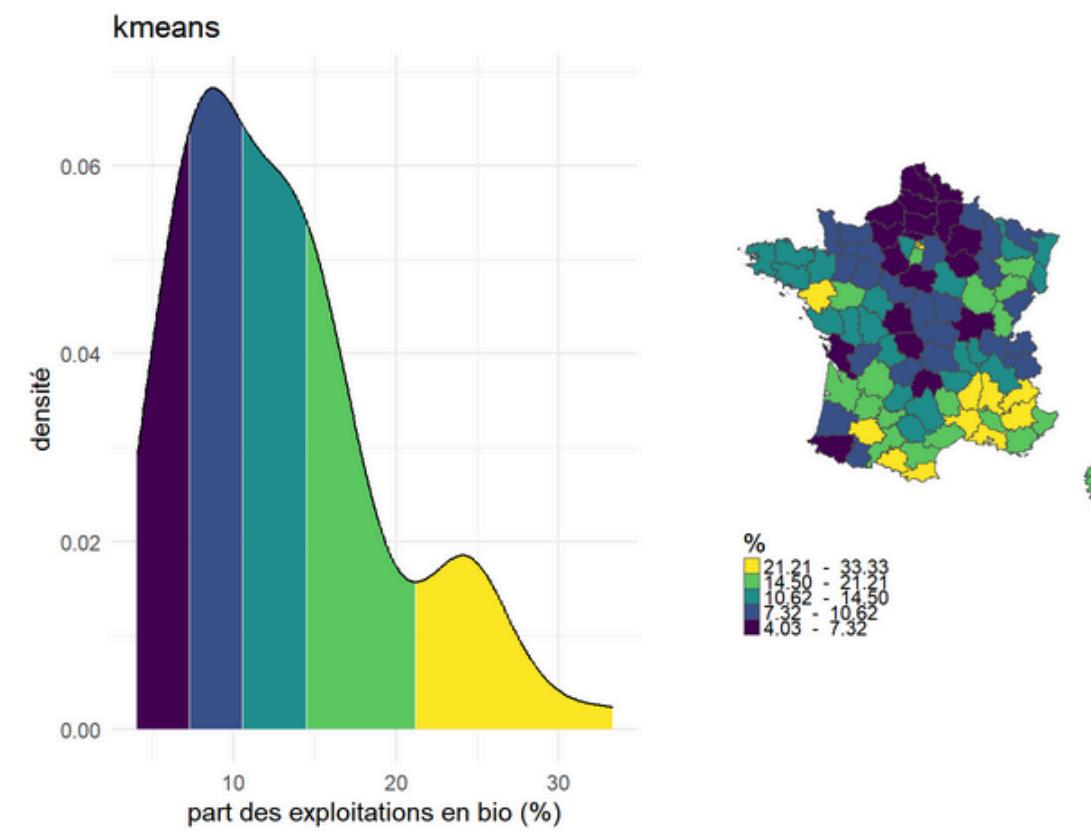
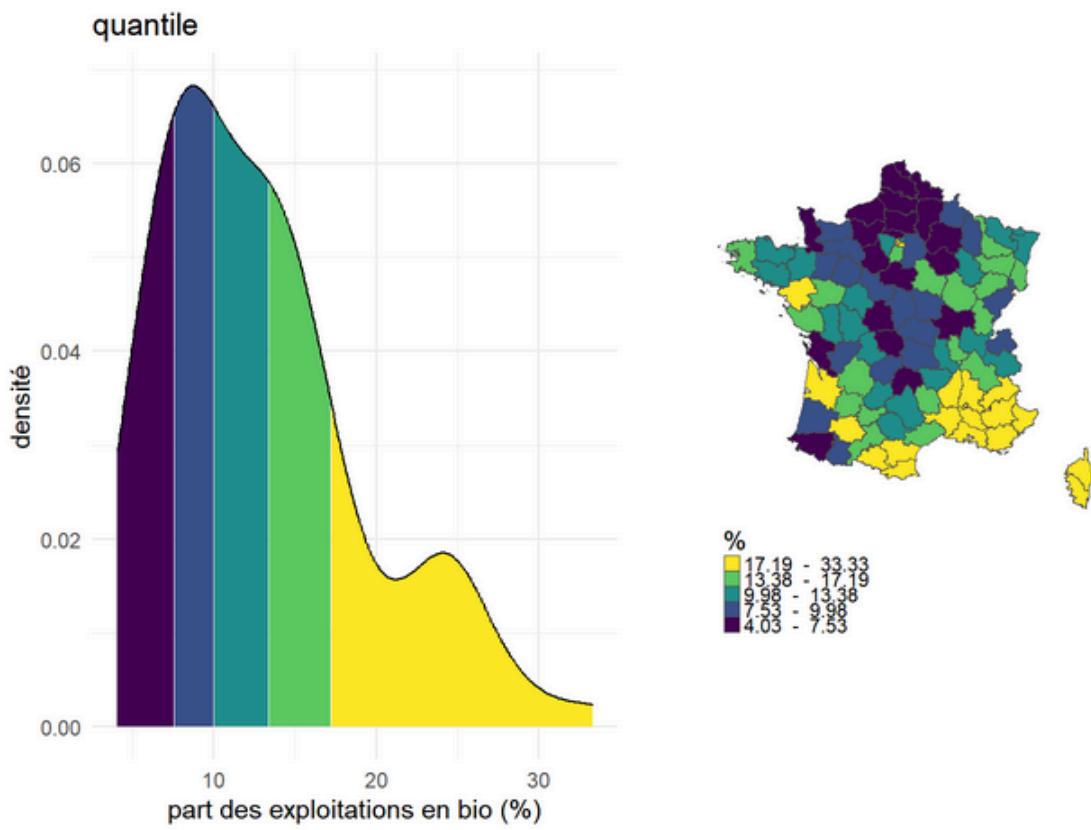
L'argument style de la fonction tm_polygons() peut recevoir les valeurs suivantes : “cat”, “fixed”, “sd”, “equal”, “pretty”, “quantile”, “kmeans”, “hclust”, “bclust”, “fisher” etc. et l'argument n reçoit le nombre de classes à construire

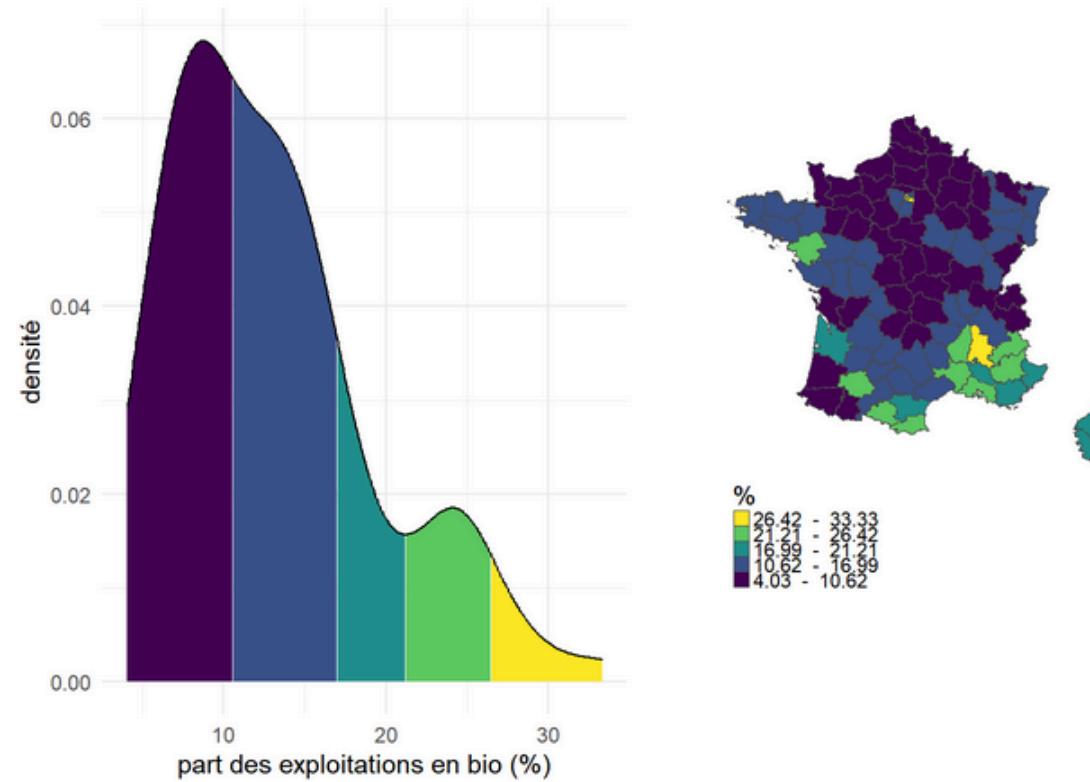
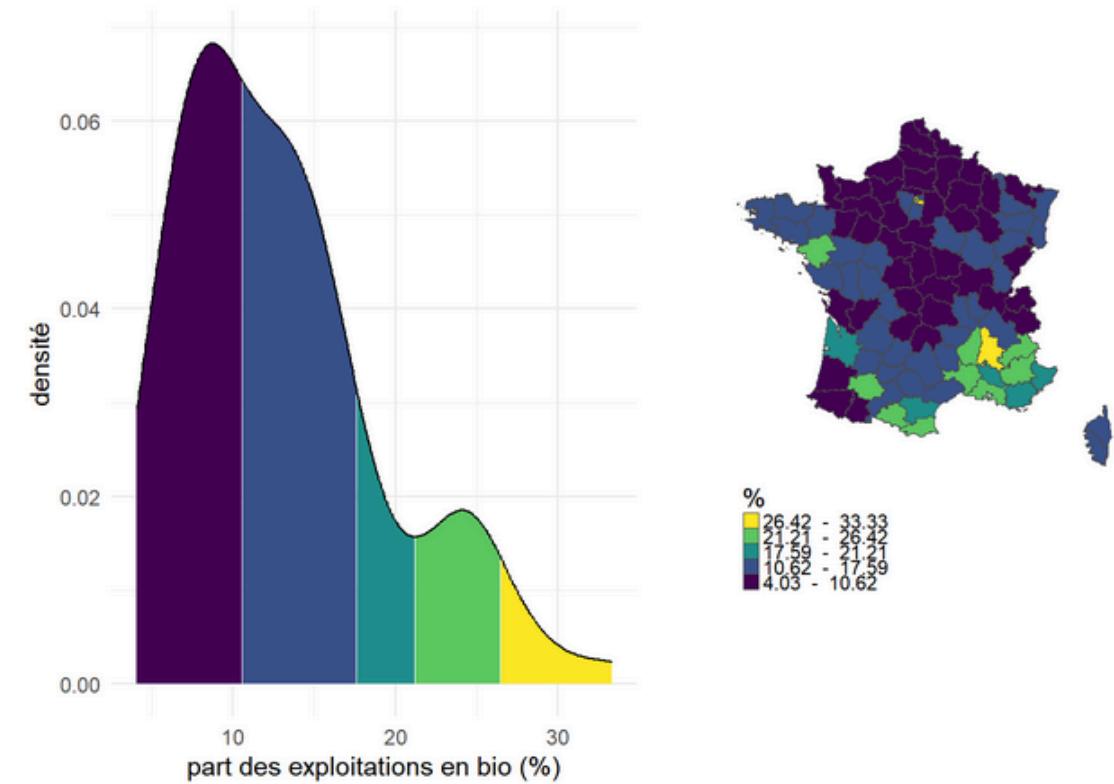
- “sd” : n classes construites à partir de la moyenne et dont les amplitudes valent un écart-type (sauf parfois les classes extrêmes)
- “equal” : n classes d'amplitudes égales
- “quantile” : n classes de même effectif
- “fisher” ou “jenks” : n classes construites avec une méthode qui minimise les variances intra-classes et maximise les variances inter-classes
- ...

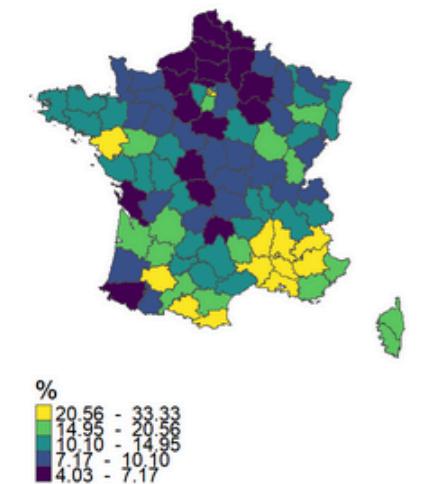
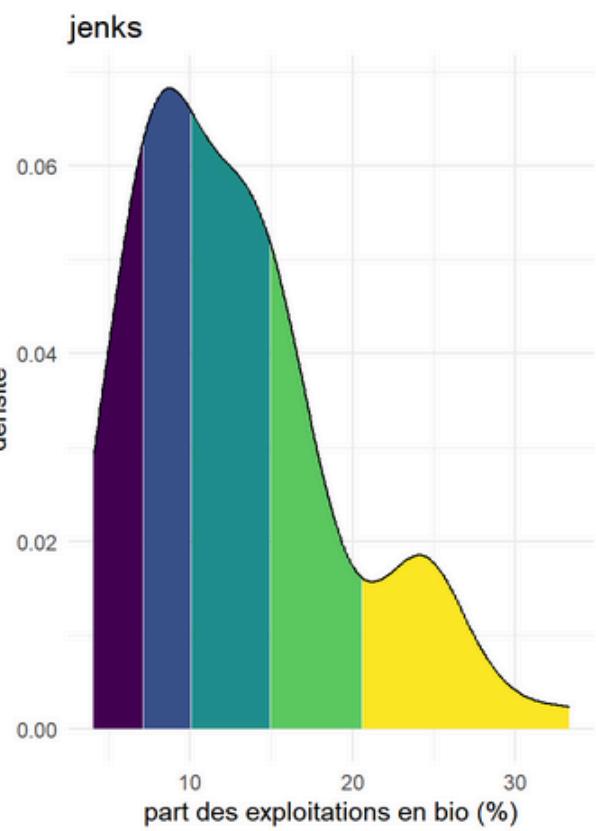
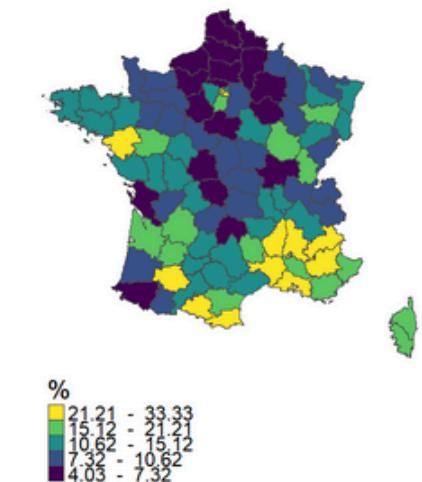
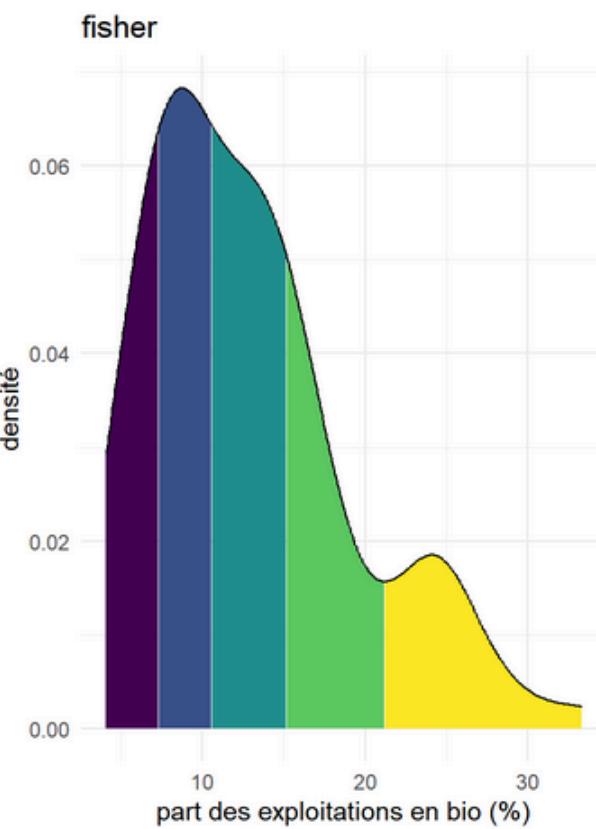
```
1 #| code-fold: true
2 #| results: hide
3 #| output: false
4
5 tmap_mode("plot")
6
7 library(classInt)
8 library(patchwork)
9
10 n_classes <- 5
11 methodes <- c("sd", "equal", "pretty", "quantile", "kmeans", "hclust", "bclust",
12               "fisher", "jenks", "dphi", "headtails")
13
14 generer <- function(methode) {
15   cesures <- classIntervals(bio$part_exp_bio, style = methode, n = n_classes)
16   palette <- viridisLite::viridis(length(cesures$brks) - 1)
17
18   carte <- tm_shape(bio) +
```

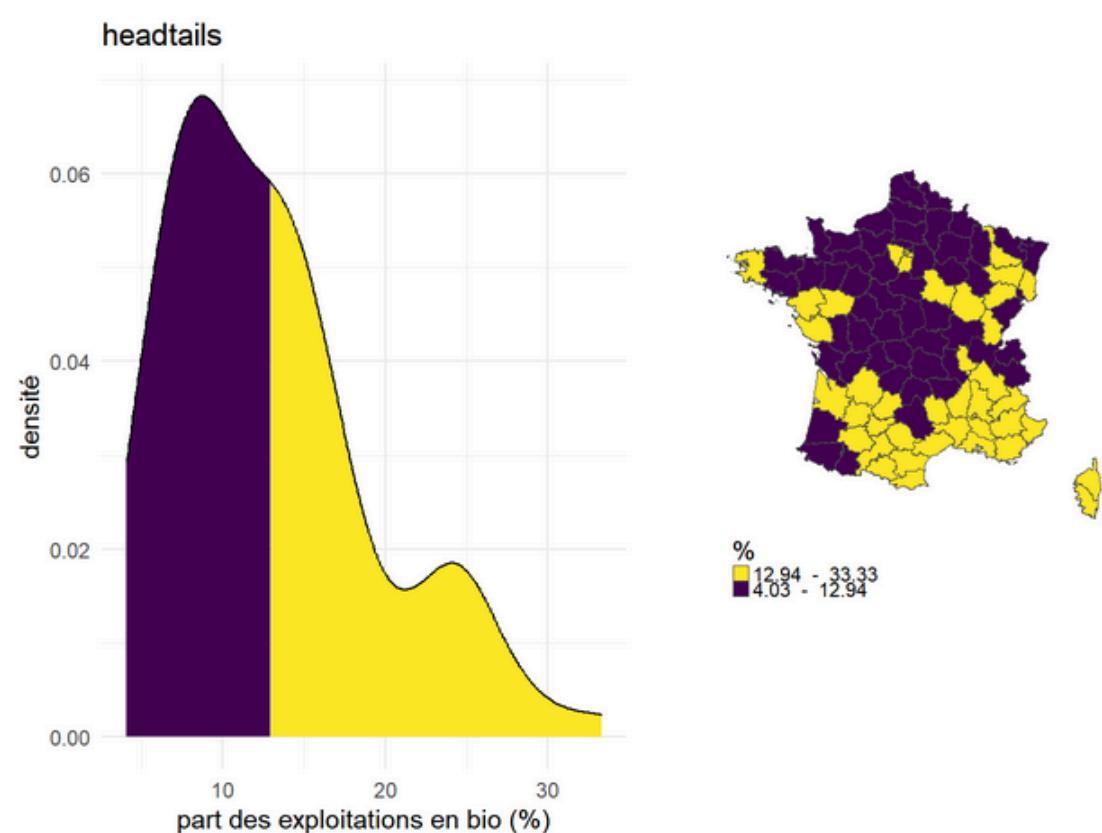
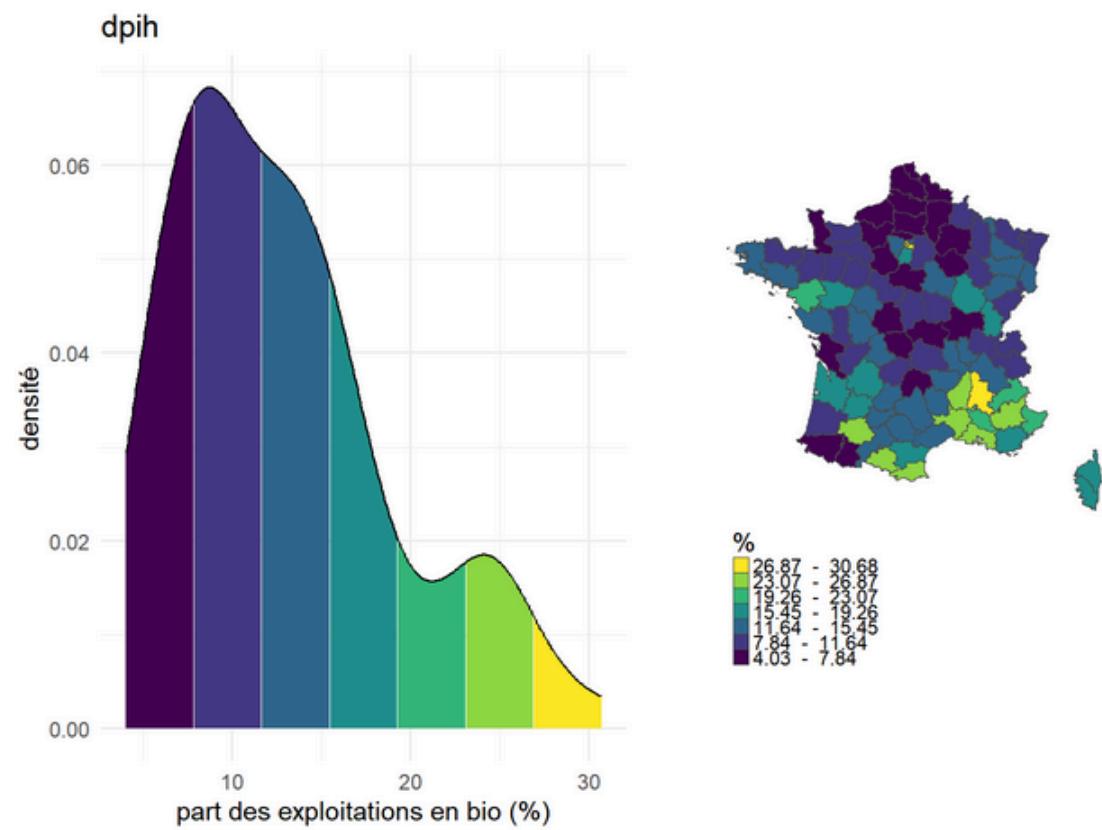






hclust**bclust**





```
1 #| echo: false
2 #| output: false
3 #| results: asis
4
5 # exemples_classes |>
6 #   map(print)
7
8 for (i in seq_along(exemples_classes)) {
9   print(exemples_classes[[i]])
10  cat("\n\n")
11 }
```

4.2 Construction des classes → méthode manuelle

Il est possible de renseigner ses propres classes dans l'argument `breaks` de la fonction `tm_polygons()`. Pour renseigner un découpage en n classes, l'argument doit contenir un vecteur de taille $n+1$: `c(min, borne1, borne2, ..., bornen-1, max)`.

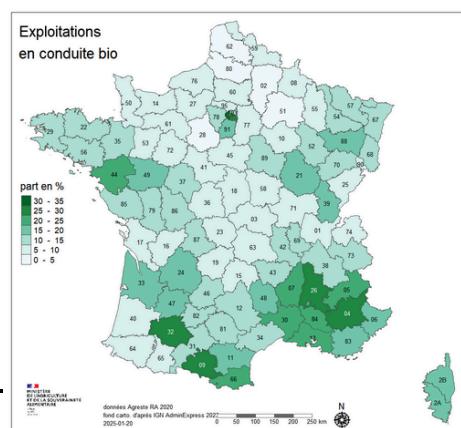
Exemple - La méthode quantile à 5 classes donne les ruptures suivantes : {2.085026 ; 4.880998 ; 6.820057 ; 9.213732 ; 13.183048 ; 50.000000}. Pour avoir des valeurs arrondies , je peux renseigner `breaks = c(2, 5, 7, 9, 13, 50)` dans la fonction.

4.3 Habillage de la carte

```

1 tmap_mode("plot")
2
3 m <- tm_shape(bio) +
4   tm_polygons("part_exp_bio",
5     style = "pretty",
6     n = 5,
7     title = "part en %",
8     palette = "BuGn",
9     border.col = "grey30",
10    lwd = 0.25,
11    legend.reverse = TRUE,
12    legend.format = list(text.separator = " - ")) +
13  tm_text("insee_dep",
14    size = 0.5,
15    remove.overlap = TRUE) +
16  tm_layout("Exploitations\nenen conduite bio",
17    scale = .8,
18    legend.position = c("left", "center")),

```



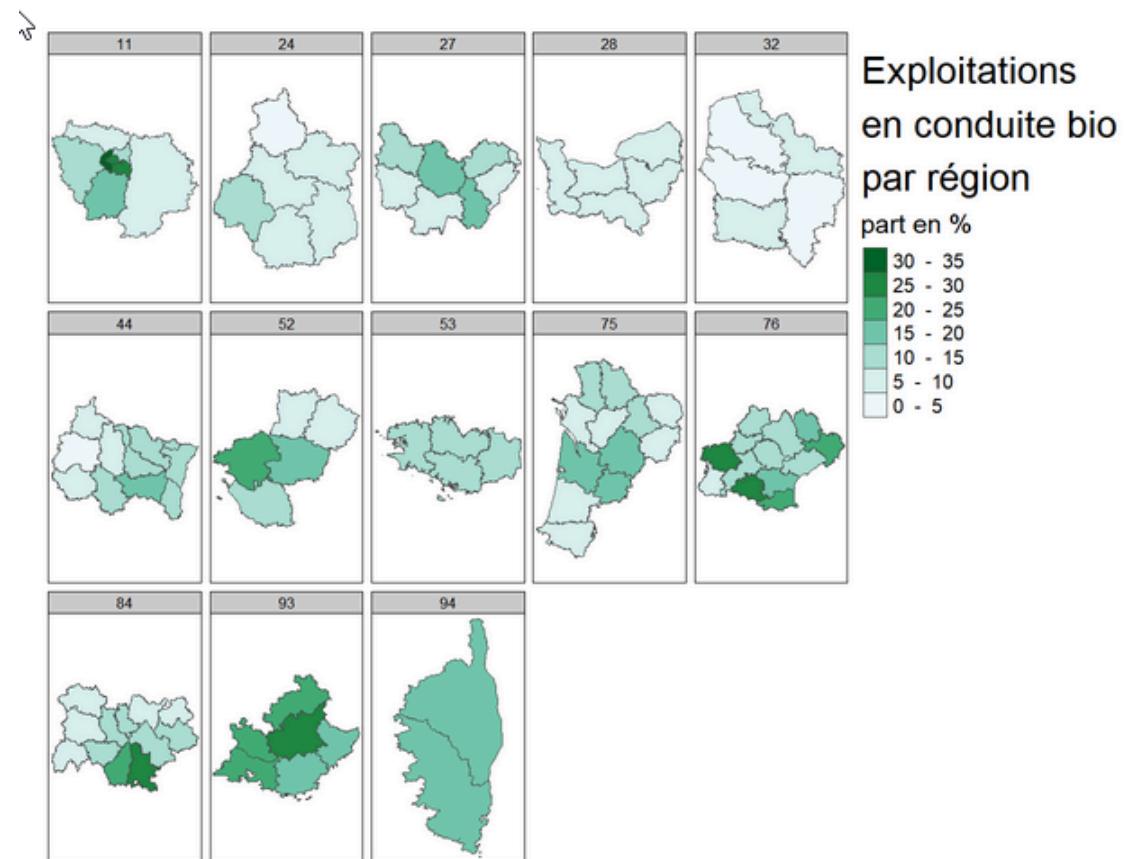
5 Facettes

5.1 Exemple Facettes

```

1 tmap_mode("plot")
2
3 tm_shape(bio) +
4   tm_polygons("part_exp_bio",
5               style = "pretty",
6               n = 5,
7               title = "part en %",
8               palette = "BuGn",
9               border.col = "grey30",
10              lwd = 0.25,
11              legend.reverse = TRUE,
12              legend.format = list(text.separator = " - ")) +
13 tm_facets("insee_reg") +
14 tm_layout("Exploitations\nen conduite bio\npar région")

```



6 Cartogramme

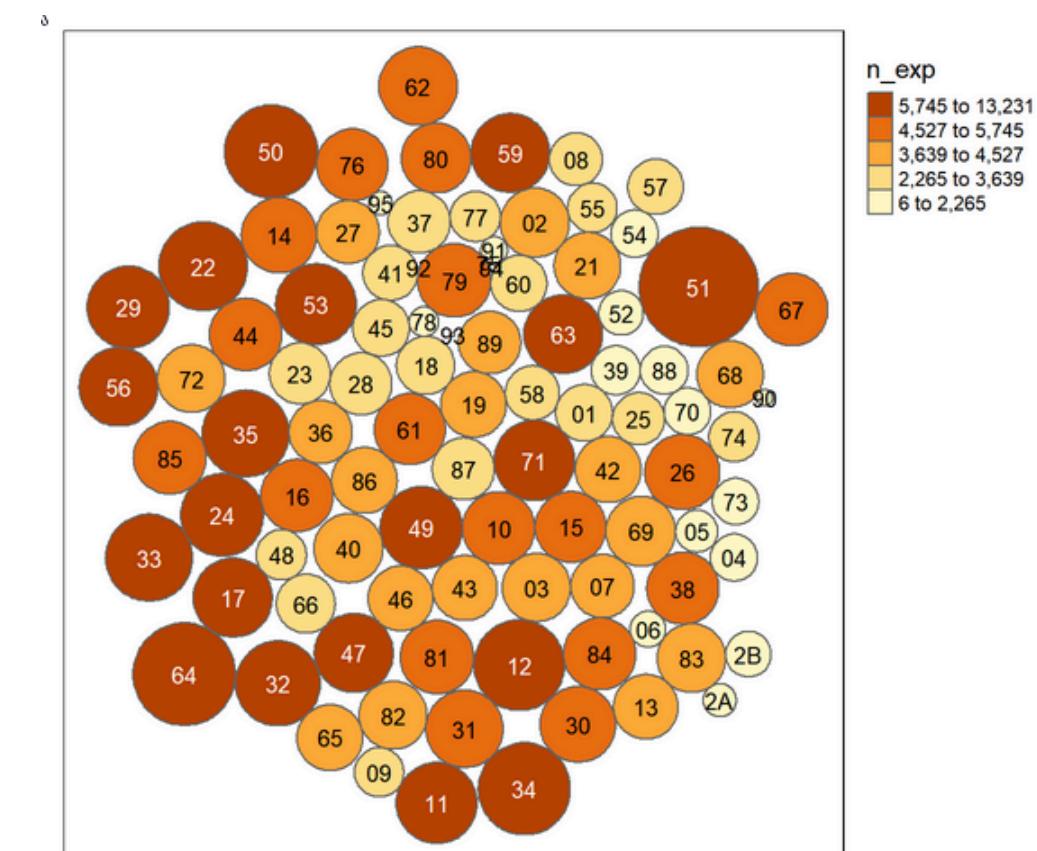


6.1 Exemple Cartogramme

```

1 library(cartogram)
2
3 bio |>
4   cartogram_dorling("n_exp") |>
5   tm_shape() +
6   tm_polygons("n_exp",
7     legend.reverse = TRUE,
8     style = "quantile",
9     n = 5) +
10  tm_text("insee_dep", size = 0.8) +
11  tm_layout(legend.outside = TRUE)

```



7 Grilles

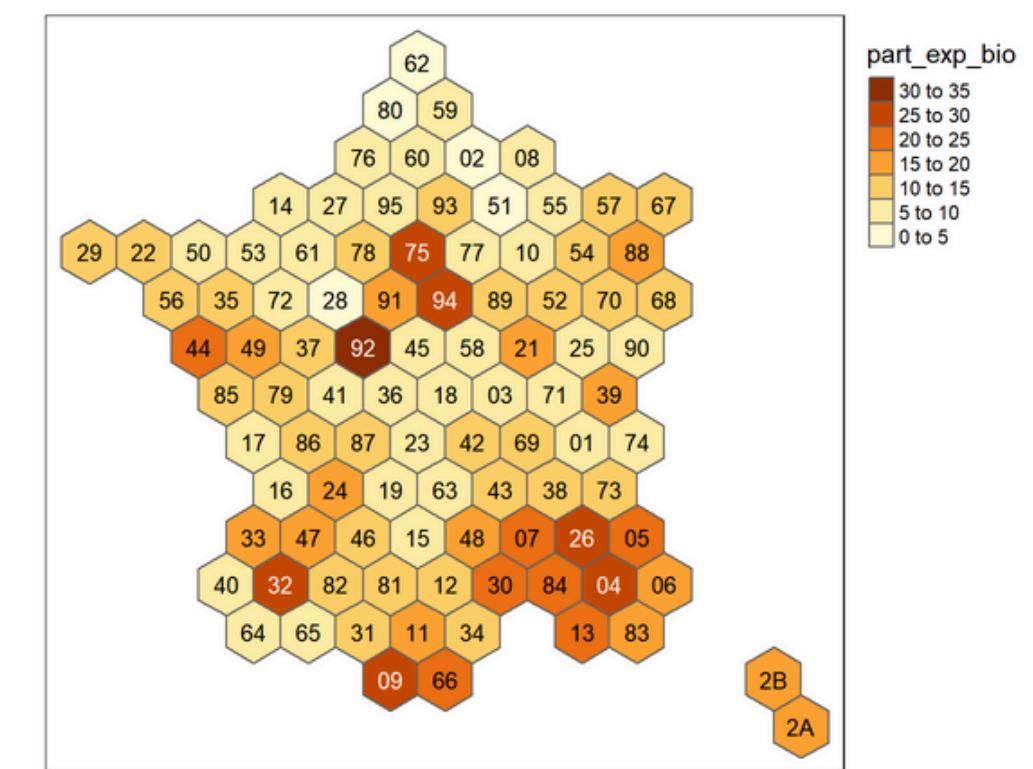


7.1 Exemple Grilles

```

1 library(geogrid)
2
3 bio |>
4   calculate_grid(grid_type = "hexagonal", seed = 3) |>
5   assign_polygons(bio, new_polygons = _) |>
6   tm_shape() +
7   tm_polygons("part_exp_bio",
8             legend.reverse = TRUE) +
9   tm_text("insee_dep", size = 0.8) +
10  tm_layout(legend.outside = TRUE)

```



8 Bibliographie





*Liberté
Égalité
Fraternité*

8.1 Liens utiles

- La [vignette du package tmap](#)
- Le [livre](#)
- Les [couleurs de R](#)

•
•
•



*Liberté
Égalité
Fraternité*

8.2 Aller plus loin

- GUR : [Cartographie avec R](#)
- La formation MTES [Analyses spatiales avec R](#)



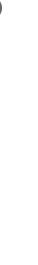
8.3 Références

- Cramer, Fabio, Grace E. Shephard, et Philip J. Heron. 2020. « The Misuse of Colour in Science Communication ». *Nature Communications* 11 (1): 5444. <https://doi.org/10.1038/s41467-020-19160-7>.
- Engelaere-Lefebvre, Juliette, Maël Theulière, et Jean-Daniel Lomenède. 2023. « Analyses spatiales avec R ». *Formations R aux MTES & MCTRCT*. https://mtes-mct.github.io/parcours_r_module_analyse_spatiale/
- Harrower, Mark, et Cynthia A. Brewer. 2003. « ColorBrewer. Org: An Online Tool for Selecting Colour Schemes for Maps ». *The Cartographic Journal* 40 (1): 27-37. <https://doi.org/10.1179/000870403235002042>.
- Loonis, Vincent, et Marie-Pierre de Bellefon. 2018. Manuel d'analyse spatiale. Théorie et mise en œuvre pratique avec R. Insee Méthodes 131. Montrouge: INSEE - Eurostat. <https://www.insee.fr/fr/information/3635442>.
- Lovelace, Robin, Jakub Nowosad, et Jannes Muenchow. 2023. Geocomputation with R. <https://r.geocompx.org/>.
- Pebesma, Edzer, et Roger Bivand. 2023. Spatial Data Science: With Applications in R. New York: Chapman and Hall/CRC. <https://doi.org/10.1201/9780429459016>.
- Rowe, Francisco, et Dani Arribas-Bel. 2022. Spatial Modelling for Data Scientists. University of Liverpool. <https://gdsl-ul.github.io/san/>.

- Theuli  re, Ma  l. 2019. « Les donn  es spatiales avec R ». <https://maeltheuliere.github.io/rspatial/index.html>.
- Wimberly, Michael C. 2023. Geographic Data Science with R: Visualizing and Analyzing Environmental Change. <https://bookdown.org/mcwimberly/gdswr-book/>.
- Zeileis, Achim, Kurt Hornik, et Paul Murrell. 2009. « Escaping RGBland: Selecting Colors for Statistical Graphics ». Research {{Report Series}} 61. Vienna: Department of Statistics and Mathematics, WU Vienna University of Economics and Business. <http://www.sciencedirect.com/science/article/pii/S0167947308005549>.
- Zeileis, Achim, et Paul Murrell. 2023. « Coloring in R's Blind Spot ». The R Journal 15 (3): 240-56. <https://doi.org/10.32614/RJ-2023-071>.

: :

9 Exercices



9.1 Données

Réaliser la carte de densité de population communale du département de l'Aisne.

Le fichier de données est le fichier *popD02_2013.rds*. Le fond de carte est le fichier *ComD02.TAB*.

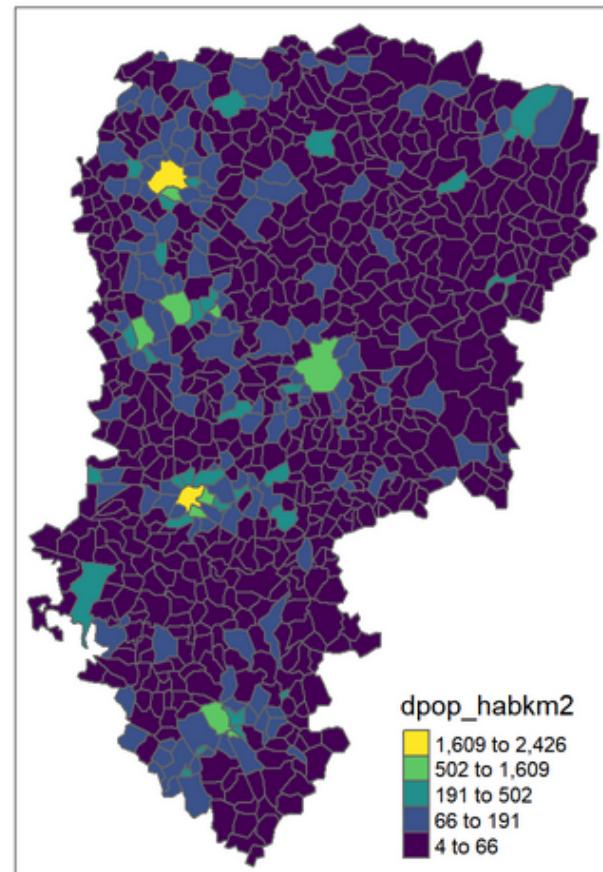
```

1 #| code-fold: true
2
3 com <- read_sf("donnees_exercices/ComD02.TAB") |>
4   clean_names()
5
6 pop <- read_rds("donnees_exercices/popD02_2013.rds") |>
7   as_tibble(.name_repair = make_clean_names())
8
9 com_pop <- com |>
10  select(-libgeo, -surf) |>
11  left_join(pop, join_by(codgeo)) |>
12  mutate(dpop_habkm2 = pop13 / surf,
13         evol_pop_pcent = (pop13 - pop08) / pop08 * 100)

```

9.2 Densité de pop

```
1 #| code-fold: true
2
3 com_pop |>
4   tm_shape() +
5   tm_polygons("dpop_habkm2",
6                 style = "kmeans",
7                 legend.reverse = TRUE,
8                 palette = "viridis")
```

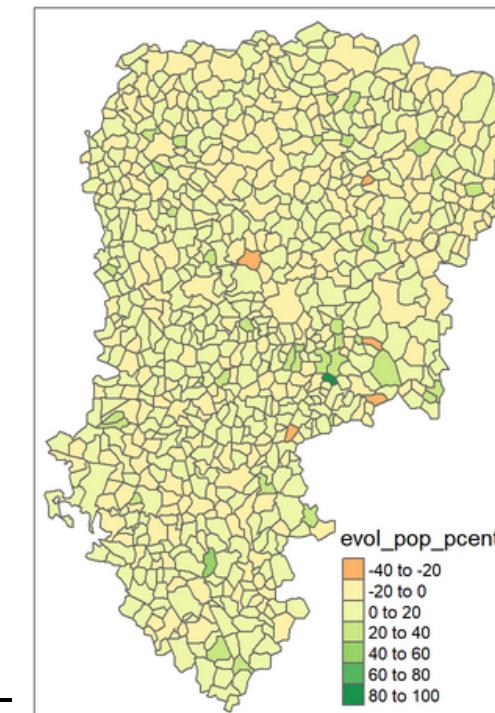


9.3 Évolution pop

Réaliser la carte d'évolution de la population communale du département de l'Aisne. On différenciera les évolutions positives et les évolutions négatives.

Le fichier de données et le fond de carte sont les mêmes que pour l'exercice précédent.

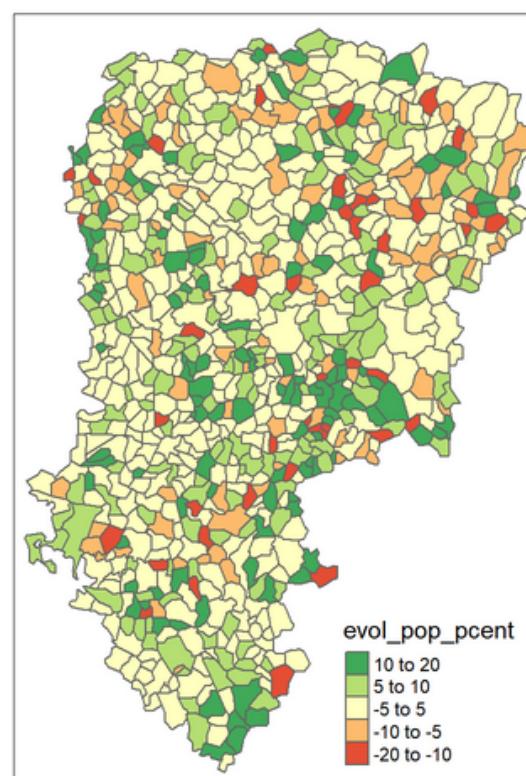
```
1 #| code-fold: true
2
3 com_pop |>
4   tm_shape() +
5   tm_polygons("evol_pop_pcent")
```



9.4 Classes

Reprendre la carte précédente en construisant des classes manuelles.

```
1 #| code-fold: true
2
3 com_pop |>
4   tm_shape() +
5   tm_polygons("evol_pop_pcent",
6                 breaks = c(-20, -10, -5, 5, 10, 20),
7                 midpoint = 0,
8                 legend.reverse = TRUE)
```



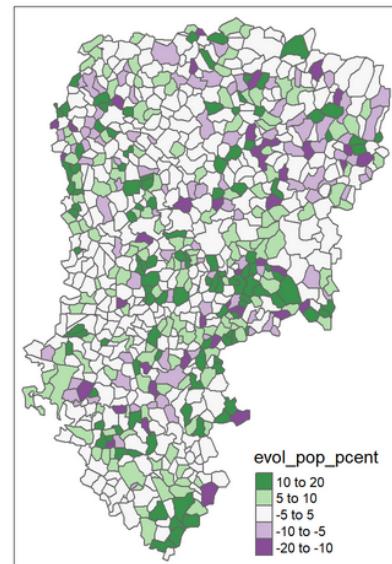
9.5 Couleurs

Reprendre la carte précédente en personnalisant les couleurs.

```

1 #| code-fold: true
2
3 com_pop |>
4   tm_shape() +
5     tm_polygons("evol_pop_pcent",
6                   style = "fixed",
7                   breaks = c(-20, -10, -5, 5, 10, 20),
8                   legend.reverse = TRUE,
9                   midpoint = 0,
10                  palette = "PRGn")

```



~~pour inverser la palette: "-PRGn"~~

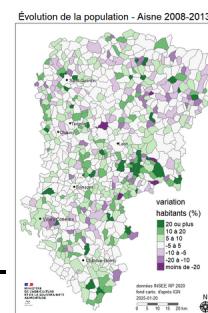
9.6 Finalisation

Habiller la carte construite au fil des exercices précédents.

```

1 #| code-fold: true
2
3 villes <- com_pop |>
4   filter(pop13 > 10000)
5
6 m <- com_pop |>
7   tm_shape() +
8     tm_polygons("evol_pop_pcent",
9                   title = "variation\nhabitants (%)",
10                  style = "fixed",
11                  breaks = c(-Inf, -20, -10, -5, 5, 10, 20, Inf),
12                  legend.reverse = TRUE,
13                  midpoint = 0,
14                  legend.format = list(text.separator = "à",
15                                       text.less.than = "moins de",
16                                       text.or.more = "ou plus"),
17                  palette = "PRGn") +
18   tm_shape(villes) +

```





*Liberté
Égalité
Fraternité*

9.7 Exporter en PDF

Exporter la carte au format PDF.

```
1 #| code-fold: true
2
3 tmap_save(m, "population_002_2008-2013.pdf",
4           width = 20, height = 28.7, units = "cm",
5           dpi = 300,
6           device = cairo_pdf)
```



•
•
•
•
•
•