

Cartographie épidémique avec R : Introduction

GRAPH Network & WHO, supported by the Global Fund to
fight HIV, TB & Malaria

October 2023

This document is a draft of a lesson made by the GRAPH Network, a non-profit headquartered at the University of Geneva Global Health Institute, in collaboration with the World Health Organization, under a Global Fund 2023 grant to create e-learning modules to build in-country data capacity for epidemiological and impact analysis for National HIV, TB and malaria programs

Création de cartes choroplèthes avec ggplot2
Introduction
Préparation des données
Création d'une belle carte choroplète avec ggplot2
Échelle des couleurs
Facet Wrap vs. Grid
Conclusion
Dernières réflexions
Références
Récapitulatif
Solution

Création de cartes choroplèthes avec ggplot2

Objectifs d'apprentissage

Dans cette leçon, vous apprendrez principalement à utiliser le package `ggplot2`, en particulier les fonctions `ggplot()` et `geom_sf()`, pour créer de belles cartes choroplèthes.

Plus spécifiquement, vous apprendrez comment :

1. Faire correspondre des données brutes à des polygones
 - Obtenir des limites/polynômes
 - Obtenir des données liées aux maladies
 - Joindre par niveau administratif
 2. Appliquer une échelle de couleurs pour les variables continues et discrètes
 - Continues
 - Discrètes
 3. Appliquer Facet Wrap et Grid
 - Créer de petits multiples et utiliser `facet_wrap()`
 - Créer de petits multiples et utiliser `facet_grid()`
-

Introduction

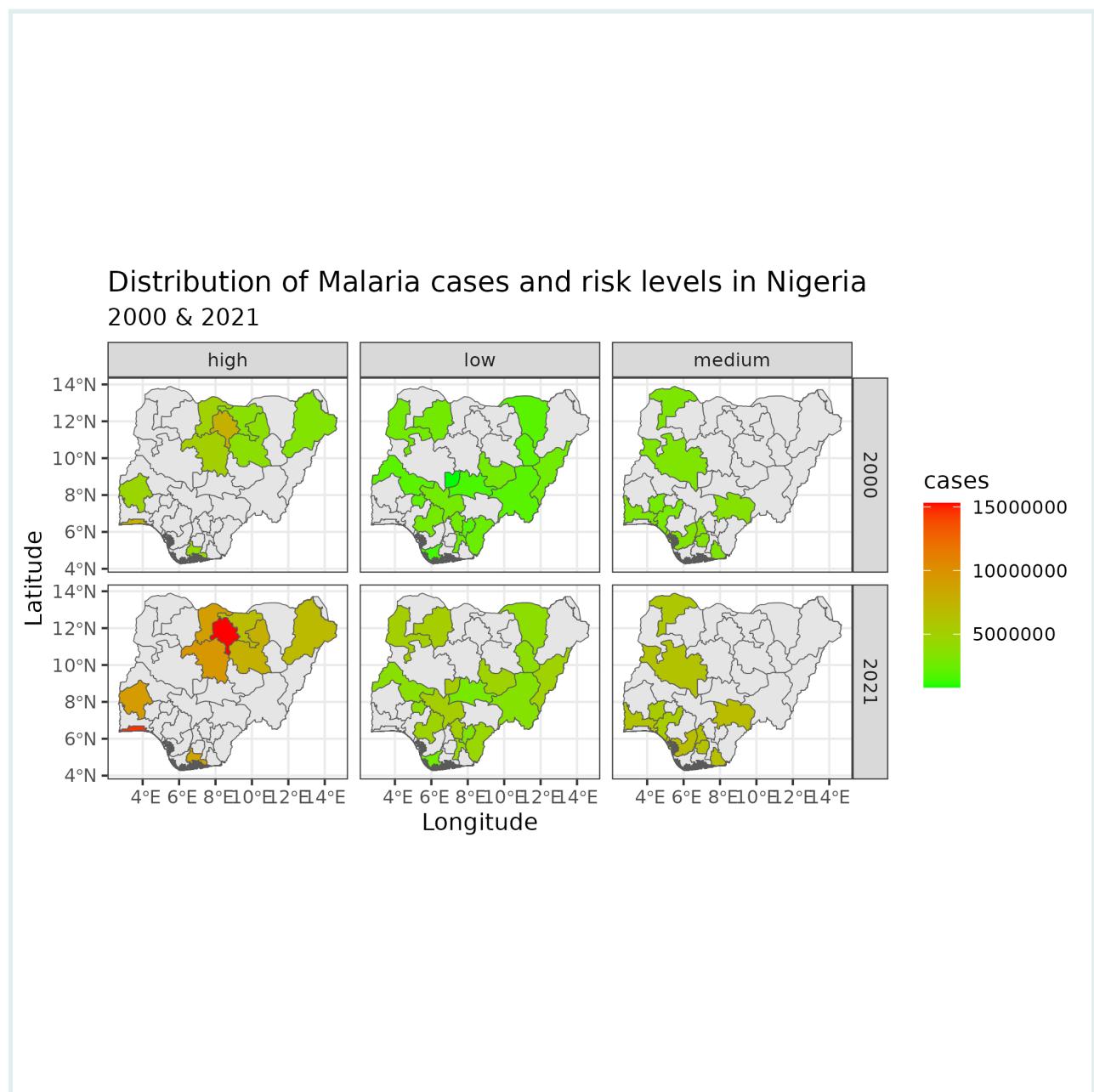
Une carte choroplète est une carte thématique dans laquelle les régions géographiques sont ombrées ou hachurées en proportion de la valeur d'une variable représentée. Cette variable peut être un indicateur épidémiologique tel que la prévalence des maladies ou le taux de mortalité. Les cartes choroplèthes sont particulièrement utiles pour visualiser les schémas spatiaux et les variations à travers différentes régions.

Les composantes essentielles d'une carte choroplète comprennent :

Régions géographiques : Il s'agit des zones qui seront représentées sur la carte, telles que les pays, les États, les districts ou toute autre division géographique.

Valeurs des données : Ce sont les valeurs associées à chaque région géographique qui seront représentées sur la carte, y compris la densité de population, la prévalence, l'incidence, le taux de mortalité, etc.

Échelle de couleurs : Il s'agit de la gamme de couleurs utilisée pour représenter les différentes valeurs des données. Généralement, on utilise un dégradé de couleurs, les couleurs plus claires représentant des valeurs plus faibles et les couleurs plus foncées représentant des valeurs plus élevées.



SIDE NOTE



Les cartes choroplèthes présentent plusieurs avantages notables :

- **Clarté visuelle** : Elles fournissent une représentation visuelle claire de la distribution spatiale des données, ce qui facilite l'identification des schémas et des tendances.
- **Facilité de compréhension** : Elles sont intuitives et faciles à comprendre, même pour les personnes sans connaissance approfondie dans le domaine.
- **Polyvalence** : Elles peuvent être utilisées pour représenter une grande variété de données, ce qui les rend adaptées à diverses applications.

Cependant, il existe également des limitations importantes à prendre en compte :

- **Sensibilité aux données** : L'apparence de la carte peut être considérablement affectée par le choix de la classification des données et de l'échelle de couleurs.
- **Biais de la taille géographique** : Les régions plus grandes peuvent sembler plus prédominantes sur la carte, ce qui entraîne un biais visuel.

Dans la section suivante, vous apprendrez comment créer une carte choroplète à l'aide du package `ggplot2` dans R.

Packages

Cet extrait de code montre le chargement des packages nécessaires pour les analyses. Dans ce guide, nous mettons l'accent sur `p_load()` du package `pacman`, qui installe le package si nécessaire et le charge pour l'utilisation. Vous pouvez également charger des packages installés avec `library()` de base R. Consultez la page sur les bases de R pour plus d'informations sur la gestion des packages en R.

```
# Charger les packages
if(!require(pacman)) install.packages("pacman")
pacman::p_load(malariaAtlas,
                ggpplot2,
                geodata,
                dplyr,
                here,
                readr,
                sf,
                patchwork)

# Désactiver la notation scientifique
options(scipen=10000)
```

Préparation des données

Avant de créer une carte choroplèthe, il est essentiel de préparer les données. Les données doivent contenir les régions géographiques et les valeurs que vous souhaitez visualiser.

Dans cette section, vous passerez par le processus de préparation des données, qui comprend les étapes suivantes :

1. **Importation des données polygonales** : Il s'agit des données géographiques qui contiennent les limites de chaque région que vous souhaitez inclure dans votre carte.
2. **Importation des données attributaires** : Il s'agit des données contenant les valeurs que vous souhaitez visualiser sur la carte, telles que la prévalence des maladies, la densité de la population, etc.
3. **Joindre les données polygonales et attributaires** : Cette étape consiste à fusionner les données polygonales avec les données attributaires en fonction d'un identifiant commun, tel que le niveau administratif ou le nom de la région. Cela créera un seul ensemble de données contenant à la fois les limites géographiques et les valeurs de données correspondantes.

Maintenant, passons en revue chaque étape en détail !

Étape 1 : Importation des données polygonales

Les polygones sont des formes fermées avec trois côtés ou plus. Dans les données spatiales, les polygones sont utilisés pour représenter des zones telles que les limites d'une ville, d'un lac ou d'un type d'utilisation des terres. Ils sont essentiels dans les systèmes d'information géographique (SIG) pour des tâches telles que la cartographie, l'analyse spatiale et la classification de l'occupation des terres.

SIDE NOTE

SIDE NOTE

Les fichiers shapefile sont un format courant pour stocker des données spatiales. Ils se composent d'au moins trois fichiers avec les extensions .shp (forme), .shx (index) et .dbf (données attributaires).

En R, vous pouvez charger des fichiers shapefile à l'aide du package `sf`. Dans notre leçon d'aujourd'hui, nous travaillerons avec des données sur le paludisme provenant de la revue épidémiologique du Nigéria (2022).

```
# Lecture du fichier shapefile
nga_adm1 <-
  sf::st_read(here::here("data/raw/NGA_adm_shapefile/NGA_adm1.shp"))
```

```
## Reading layer `NGA_adm1' from data source
##
`/Users/kendavidn/Dropbox/tgc_github_projects/epi_reports_staging/data/raw/NGA_adm_shapefile'
##   using driver `ESRI Shapefile'
## Simple feature collection with 38 features and 9 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 2.668431 ymin: 4.270418 xmax: 14.67642 ymax: 13.89201
## Geodetic CRS:  WGS 84
```

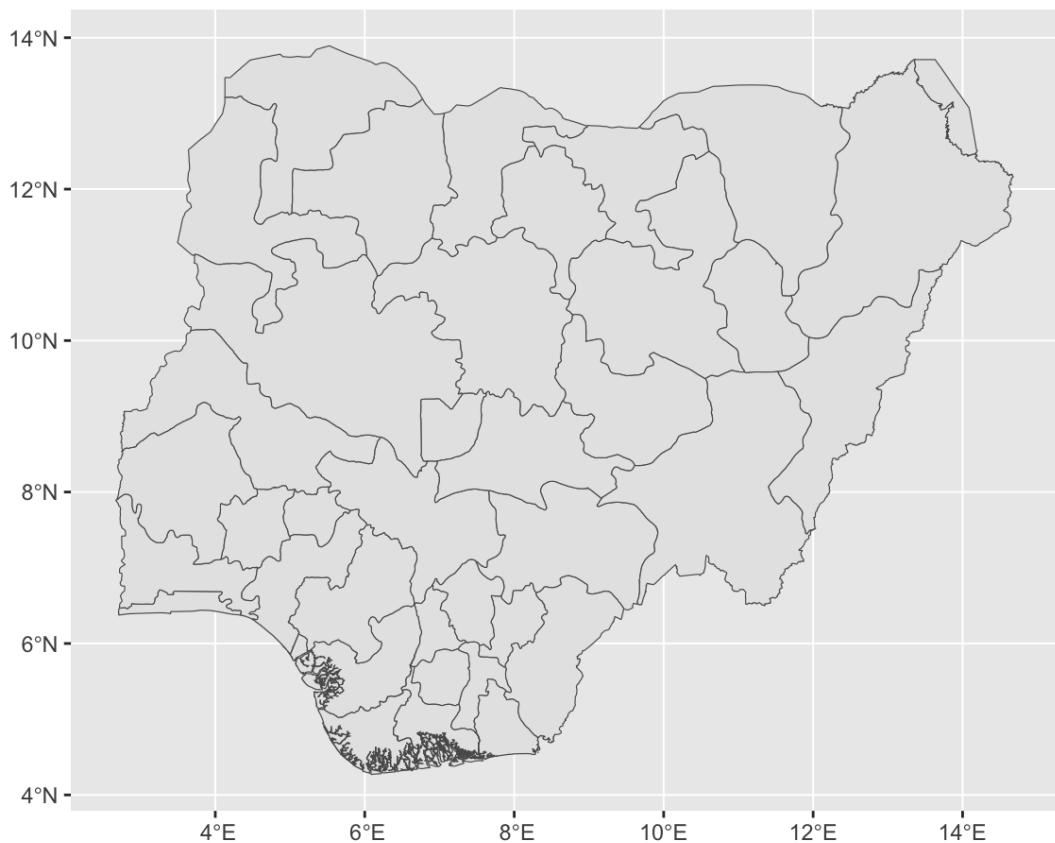
Les éléments importants de n'importe quelle couche ggplot sont les associations esthétiques `aes(x, y, ...)` qui indiquent à ggplot où placer les objets du graphique.

Nous pouvons imaginer une carte comme un graphique avec toutes les caractéristiques associées à un axe x et un axe y. Tous les types de géométrie (`geom_`) dans ggplot ont une sorte d'association esthétique, et celles-ci peuvent être déclarées au niveau du graphique, par exemple, `ggplot(data, aes(x = variable1, y = variable2))`, ou au niveau de la couche individuelle, par exemple, `geom_point(aes(color = variable3))`.

Ci-dessous, vous pouvez voir que `geom_sf()` est utilisé pour tracer les limites des différents États du Nigéria. De manière similaire, différentes couches peuvent être ajoutées par-dessus.

```
ggplot() +
  geom_sf(data = nga_adm1) +
  labs(title = "Aperçu du fichier shapefile du Nigéria")
```

Aperçu du fichier shapefile du Nigéria



Ici, nous lisons d'abord le fichier shapefile du Nigéria en utilisant `sf::st_read()` puis nous le représentons à l'aide de `ggplot2`. La fonction `geom_sf()` est utilisée pour afficher les données spatiales, et `labs()` est utilisé pour ajouter un titre au graphique.



PRO TIP Étant donné que le fichier shapefile a été chargé à l'aide de `sf::st_read()`, nous n'avons pas besoin de spécifier les noms des axes. En fait, les coordonnées sont stockées en tant qu'objet multipolygone dans la variable de géométrie, et `geom_sf()` les reconnaît automatiquement.

Étape 2 : Importation des données attributaires



VOCAB

Dans le contexte des cartes choroplèthes, les “données attributaires” font référence aux informations quantitatives ou qualitatives qui seront utilisées pour ombrer ou colorer les différentes zones géographiques sur la carte. Par exemple, si vous avez une carte des régions d'un pays et que vous souhaitez ombrer chaque région en fonction de sa population, les

VOCAB

données de population seront considérées comme des données attributaires.

Comme indiqué ci-dessus, nous utiliserons le nombre de cas de paludisme signalés au Nigéria pour les années 2000, 2006, 2010, 2015 et 2021.

```
# Lecture des données attributaires
malaria_cases <- read_csv(here::here("data/malaria.csv"))
malaria_cases
```

Étape 3 : Vérification des Données Jointes

Il est essentiel de vérifier et de valider les données jointes pour garantir le succès de la fusion et l'exactitude des données.

```
# Valider les données jointes
all.equal(unique(nga_adm1$NAME_1), unique(malaria_cases$state_name))
```

```
## [1] "Lengths (38, 37) differ (string compare on first 37)"
## [2] "2 string mismatches"
```

```
# Identifier les divergences
setdiff(unique(nga_adm1$NAME_1), unique(malaria_cases$state_name))
```

```
## [1] "Water body"
```

Dans l'extrait de code fourni, nous comparons les noms de région uniques entre les ensembles de données `nga_adm1` et `malaria_cases` en utilisant la fonction `all.equal()`. Cela vise à garantir que toutes les régions dans le fichier de formes correspondent à leurs homologues dans les données attributives.

Si les noms de région sont identiques, `all.equal()` renverra `TRUE`. Cependant, s'il y a des divergences, il détaillera les différences entre les deux ensembles de noms de région.

Il est à noter que “Water body” est présent dans le fichier de formes mais pas dans les `malaria_cases`. Étant donné que “Water body” n'est pas une région proprement dite, il doit être supprimé avant de fusionner les ensembles de données. Cela peut être fait avec la fonction `filter()`.

```
nga_adm1 <- filter(nga_adm1, NAME_1 != "Water body")
```

Étape 4 : Fusion des Données par Niveaux Administratifs

Maintenant, nous allons obtenir les données que nous souhaitons représenter sur la carte. L'élément essentiel est que les noms de région soient les mêmes dans le fichier de formes que dans les données que vous souhaitez représenter, car cela sera nécessaire pour les fusionner correctement.

Avant de fusionner les deux ensembles de données, nous devons définir la clé de fusion (by =).

REMINDER



N'oubliez pas de consulter la leçon sur la fusion de tables pour plus de détails sur la manière de fusionner des data.frames dans R si vous n'êtes pas familier avec la fusion.

```
# Ajouter les données de population et calculer les cas pour 10 000 habitants
malaria <- malaria_cases %>%
  left_join(nga_adm1,
            by = c("state_name" = "NAME_1")) %>%
  st_as_sf() # convertir en fichier de formes

# Sélectionner les colonnes les plus importantes
malaria2 <- malaria %>%
  select(state_name, cases_2000, cases_2006, cases_2010, cases_2015,
         cases_2021, geometry)
```

Dans cette étape, nous fusionnons les données du fichier de formes `nga_adm1` avec les données `malaria_cases` en utilisant `left_join()` du package `dplyr`. L'argument `by` est utilisé pour spécifier la colonne commune sur laquelle fusionner les ensembles de données.

Enfin, nous convertissons les données fusionnées en un fichier de formes en utilisant `st_as_sf()`.

Nous pouvons conserver uniquement les variables importantes qui seront utilisées dans la construction des graphiques en utilisant `select()`.



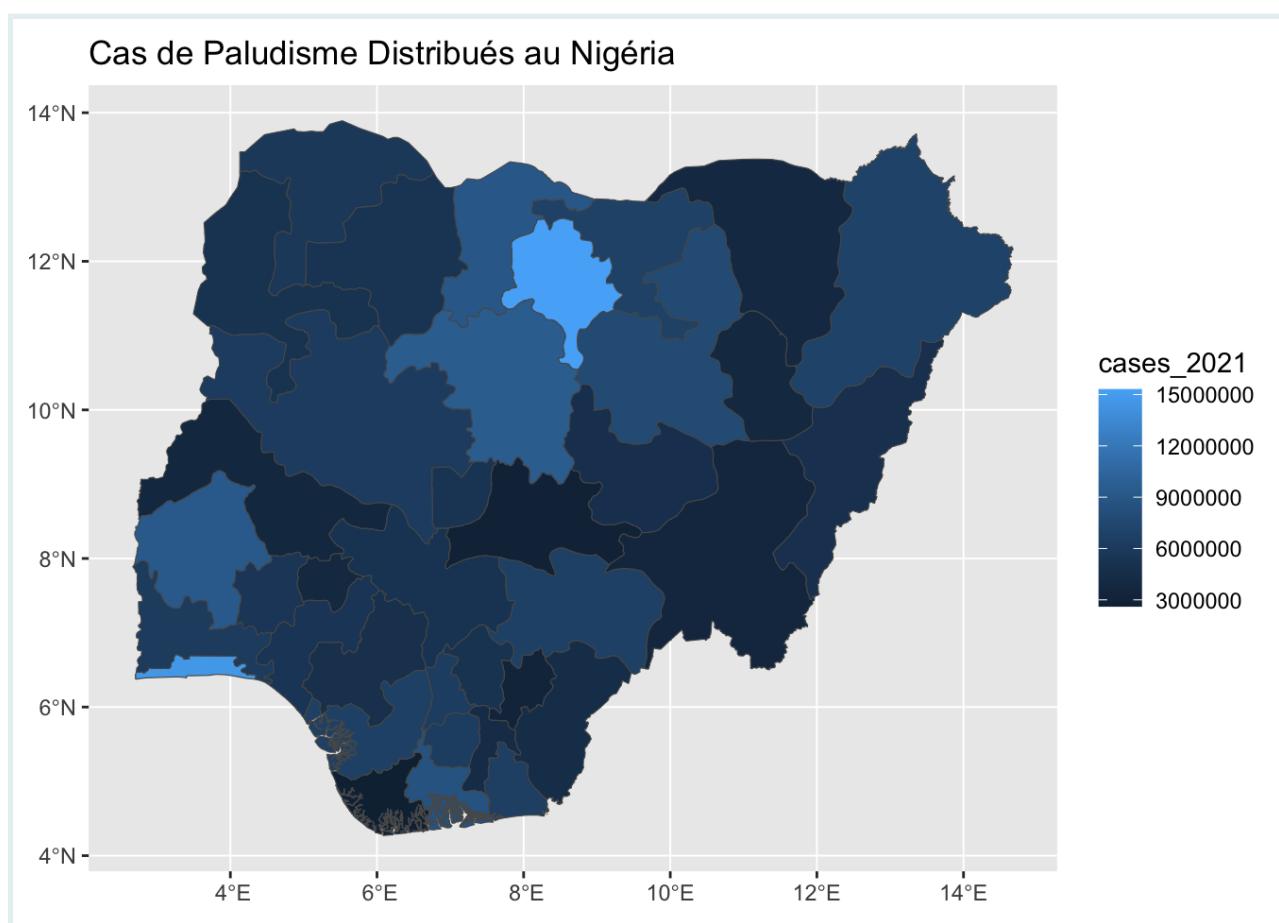
Dans cette partie de la leçon, nous avons appris l'importance des niveaux administratifs dans les données spatiales et comment fusionner les données spatiales et les données attributives par niveaux administratifs en utilisant le package `dplyr` de R, ainsi que comment vérifier et valider les données jointes.

Création d'une belle carte choroplète avec `ggplot2`

Utilisation de la variable de remplissage (c'est-à-dire la variable attributaire)

Pour afficher le nombre de cas, par exemple pour l'année 2021, nous devons remplir les polygones dessinés avec `geom_sf()` en utilisant la variable `fill`. Cela est très simple car il suit la même logique que la syntaxe du package `ggplot2`.

```
ggplot(data=malaria2) +  
  geom_sf(aes(fill=cases_2021)) +  # définir le remplissage en fonction de la  
  # variable de comptage des cas  
  labs(title = "Cas de Paludisme Distribués au Nigéria")
```



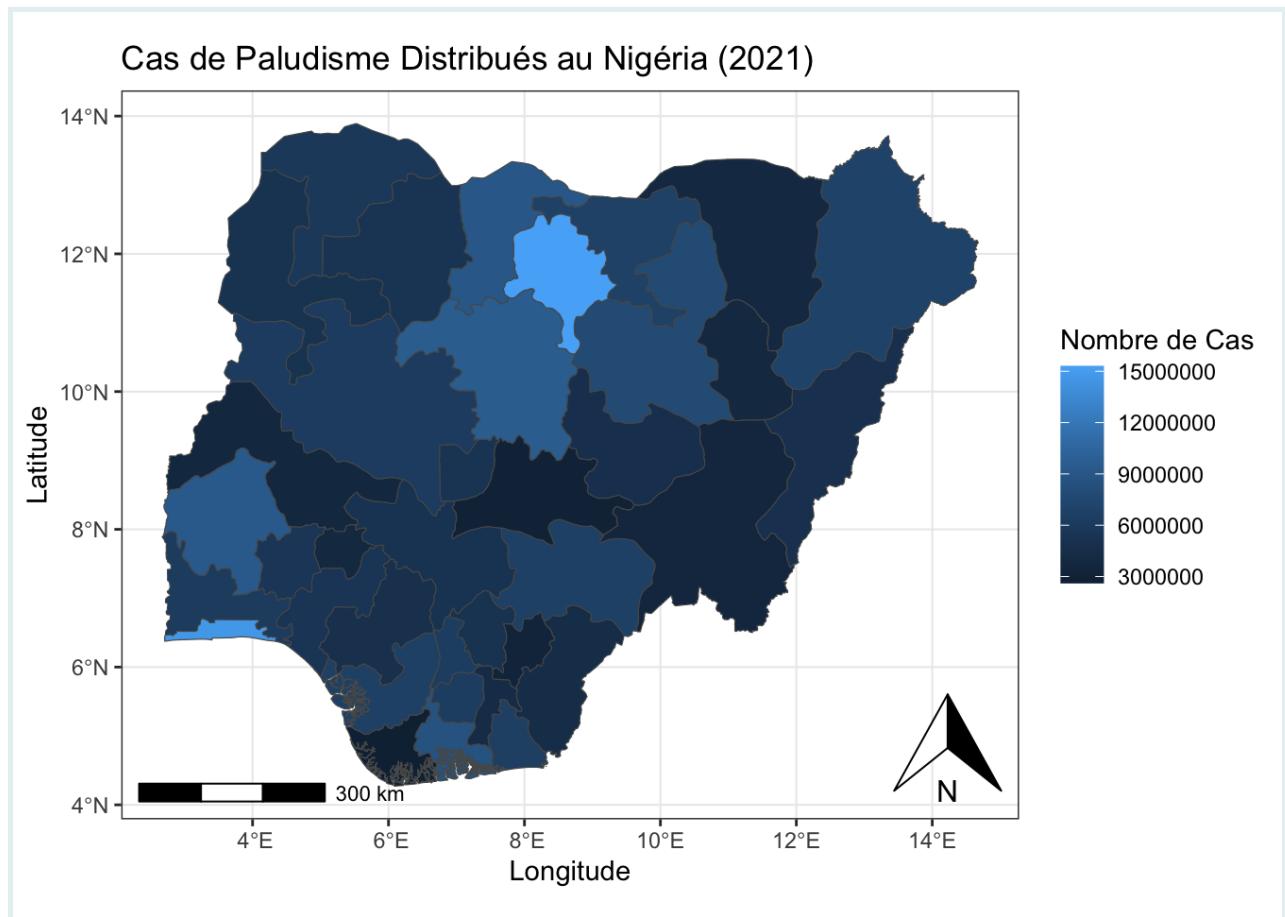
Nous créons une carte choroplète de base à l'aide de `ggplot2`. L'esthétique `fill` est définie pour varier en fonction de la variable `cases`, ce qui colore les régions en fonction du nombre de cas de paludisme.

Personnalisation de la carte

Nous pouvons personnaliser la carte en ajoutant des titres aux axes et à la légende, en ajoutant une flèche indiquant le nord et une échelle à l'aide de

`ggspatial::annotation_north_arrow()` et `ggspatial::annotation_scale()`, et en changeant le thème en `theme_bw()`.

```
ggplot(data=malaria2) +  
  geom_sf(aes(fill=cases_2021)) +  # définir le remplissage en fonction de la  
  # variable de comptage des cas  
  labs(title = "Cas de Paludisme Distribués au Nigéria (2021)",  
       fill = "Nombre de Cas") +  
  xlab("Longitude") +  
  ylab("Latitude") +  
  ggspatial::annotation_north_arrow(location = "br") +  
  ggspatial::annotation_scale(location = "bl") +  
  theme_bw()
```



Dans cette section, nous avons d'abord créé une carte choroplète de base à l'aide de `ggplot2` pour visualiser les cas de paludisme régionaux au Nigéria pour 2021. Ensuite, nous avons amélioré la carte en ajoutant des titres, des étiquettes d'axes, une flèche indiquant le nord, une échelle, et en appliquant un thème monochrome.

1. Construisez une carte choroplète pour afficher la répartition des cas de paludisme en 2019, en utilisant la colonne `cases_2019` de l'ensemble de données `malaria2`.

Vous pouvez améliorer la conception et la clarté de votre carte en incorporant des titres, des étiquettes d'axes et d'autres étiquettes pertinentes.

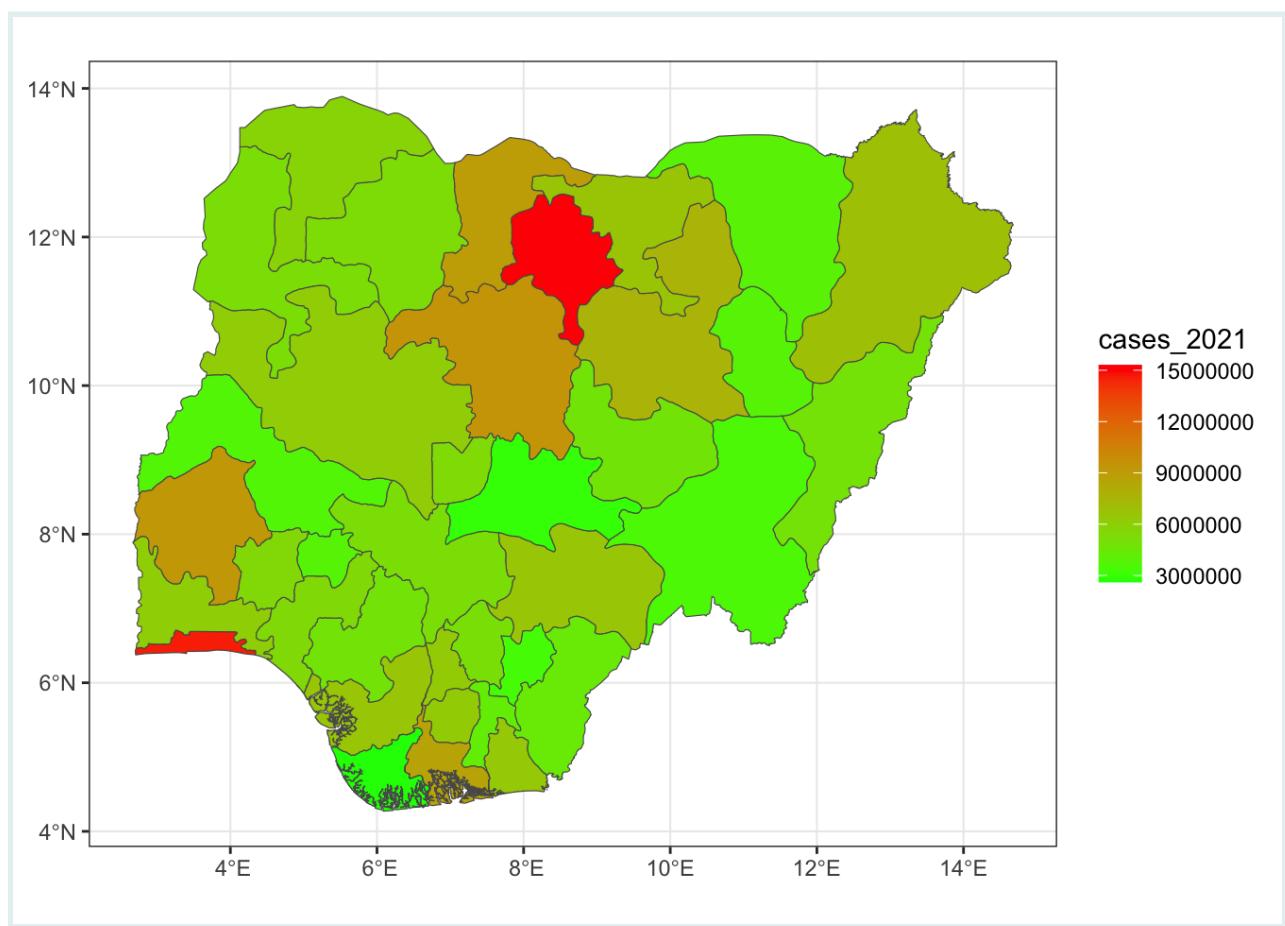
Échelle des couleurs

Échelle des couleurs pour les attributs continus

La fonction `scale_fill_continuous()` du package `ggplot2` en R est utilisée pour appliquer une échelle de couleurs continue à une carte choroplèthe.

Nous pouvons personnaliser la palette de couleurs utilisée pour l'échelle de couleurs continues en spécifiant les paramètres `low` et `high` dans la fonction `scale_fill_continuous()`.

```
# Créer un objet ggplot
ggplot(data = malaria2) +
  geom_sf(aes(fill = cases_2021)) +
  scale_fill_continuous(low = "green", high = "red") # appliquer une échelle
  # de couleurs continue
  theme_bw()
```



RECAP

RECAP

Dans cette section, nous avons appris comment appliquer une échelle de couleurs continue à une carte choroplète à l'aide de la fonction `scale_fill_continuous()` du package `ggplot2` en R et comment personnaliser la palette de couleurs.

Échelle des couleurs pour les attributs discrets

REMINDER

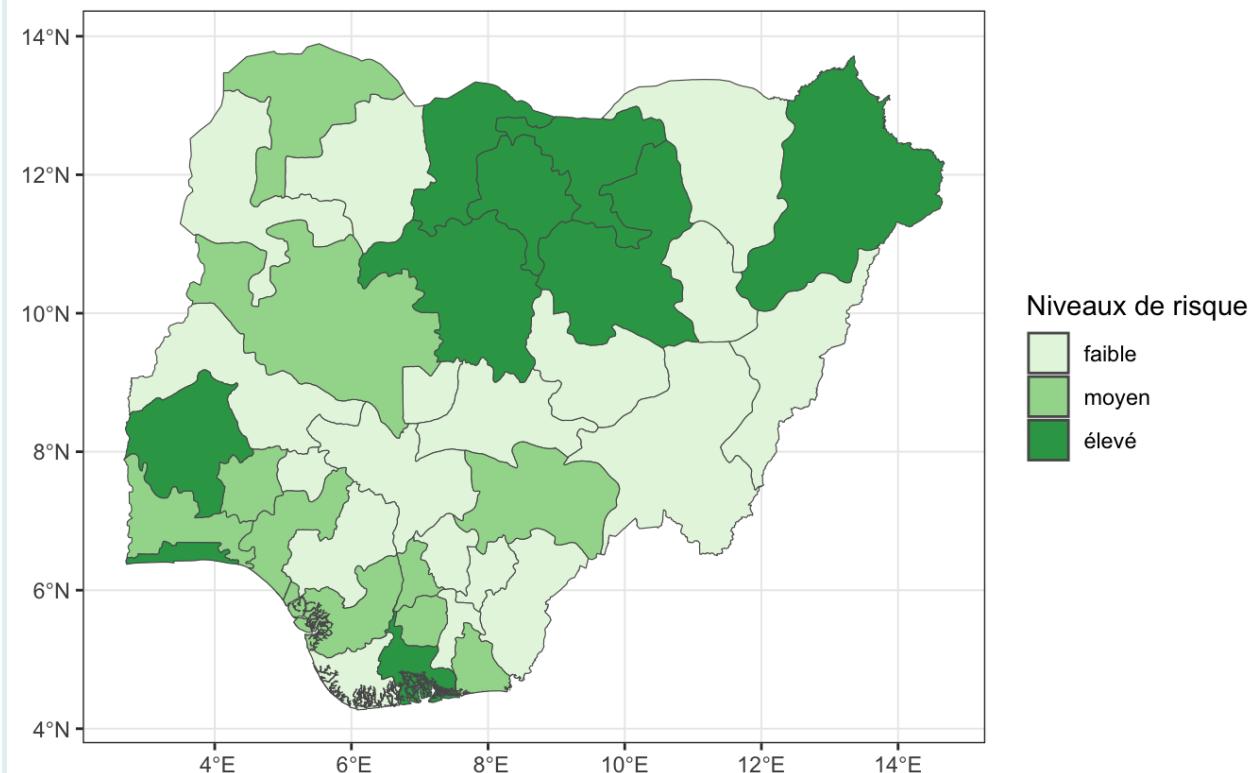
Les données discrètes sont un type de données quantitatives qui ne peuvent prendre que des valeurs spécifiques et distinctes. Elles sont souvent le résultat du décompte d'objets ou d'événements. Les données discrètes sont importantes dans les cartes choroplèthes car elles nous permettent de représenter le décompte ou la quantité d'objets ou d'événements dans différentes régions.

La fonction `scale_fill_brewer()` du package `ggplot2` en R est utilisée pour appliquer une échelle de couleurs discrètes à une carte choroplète.

Avant d'appliquer l'échelle de couleurs discrètes, nous devrons créer une nouvelle colonne discrète, qui pourrait être un niveau de risque basé sur le nombre de cas. Pour cela, nous pouvons utiliser `mutate()` combinée avec `case_when()`.

```
# Transformation des données : création d'une nouvelle colonne 'risk' basée
# sur le nombre de cas en 2021
malaria2 %>%
  mutate(risk = case_when(cases_2021 < quantile(cases_2021, 0.5) ~ 'faible',
                          cases_2021 > quantile(cases_2021, 0.75) ~ 'élevé',
                          TRUE ~ 'moyen')) -> malaria3

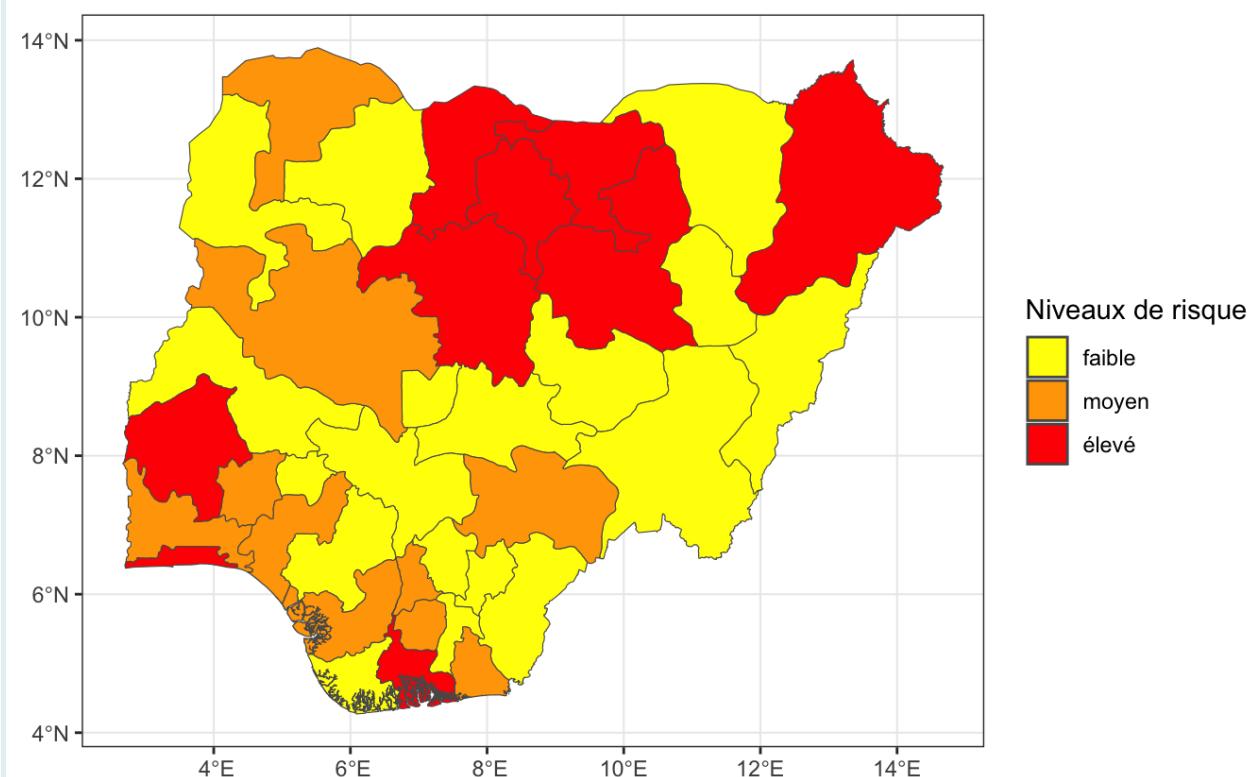
# Visualisation des données
ggplot(data = malaria3) +
  geom_sf(aes(fill = fct_reorder(risk, cases_2021))) + # Les niveaux de risque
  # sont réorganisés en fonction du nombre de cas
  scale_fill_brewer(palette = "Set4", "Niveaux de risque") +
  theme_bw()
```



Vous pouvez également créer une palette de couleurs personnalisée pour les variables discrètes.

```
palette_personnalisee <- c("yellow", "orange", "red") # créer manuellement une
                                                     palette de couleurs personnalisée

# Appliquer la palette de couleurs personnalisée
ggplot(data = malaria3) + # Créer un objet ggplot
  geom_sf(aes(fill = fct_reorder(risk, cases_2021)))+ # réorganiser les
  étiquettes de risque en fonction du nombre de cas
  scale_fill_manual(values = palette_personnalisee, "Niveaux de risque") +
  coord_sf(expand = TRUE) +
  theme_bw()
```



RECAP



Dans cette section, nous avons appris comment appliquer une échelle de couleurs discrètes à une carte choroplète à l'aide de la fonction `scale_fill_brewer()` du package `ggplot2` et comment personnaliser la palette de couleurs.

PRACTICE



2. Créez votre propre palette de couleurs distincte de celle initiale fournie ci-dessous, et affichez les cas de paludisme au Nigéria pour l'année 2000 en utilisant cette palette de couleurs personnalisée. N'oubliez pas d'incorporer des améliorations esthétiques supplémentaires.

Facet Wrap vs. Grid

- `facet_wrap()` : Cette fonction enveloppe une séquence 1D de panneaux en 2D. C'est utile lorsque vous avez une seule variable avec de nombreux niveaux et que

vous souhaitez disposer les graphiques de manière plus économique en termes d'espace.

- `facet_grid()` : Cette fonction crée une matrice de panneaux définie par des variables de facettement en lignes et en colonnes. Elle est particulièrement utile lorsque vous avez deux variables discrètes, et que toutes les combinaisons des variables existent dans les données.

PRO TIP



- `facet_wrap()` est utilisé pour le facettement d'une seule variable, tandis que `facet_grid()` est utilisé pour le facettement de deux variables.
- `facet_wrap()` dispose les panneaux dans une grille 2D, tandis que `facet_grid()` les dispose dans une matrice.

Conclusion

Félicitations, vous avez terminé cette leçon !

Dans cette leçon, vous avez appris :

- Ce qu'est une carte choroplète et ses composantes.
- Les avantages et les limites des cartes choroplèthes.
- Comment préparer les données pour créer une carte choroplète.
- Comment créer une carte choroplète à l'aide de `ggplot2`.
- Comment appliquer des échelles de couleurs continues et discrètes.
- Comment créer des multiples avec `facet_wrap()` et `facet_grid()`.

Avec ces connaissances, vous êtes maintenant prêt à créer vos propres cartes choroplèthes à l'aide de `ggplot2` en R.

Dernières réflexions

La création de cartes choroplèthes est une compétence essentielle pour toute personne travaillant avec des données géographiques. Avec les outils et les techniques que vous avez appris dans cette leçon, vous pouvez maintenant créer vos propres cartes choroplèthes en utilisant `ggplot2` en R. N'hésitez pas à expérimenter avec différentes ensembles de données, palettes de couleurs et variables de facettement pour créer des cartes informatives et visuellement attrayantes. Bonne cartographie !

Références

1. Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis* (3e). Available at: <https://ggplot2-book.org/>
2. Hadley Wickham and Garrett Grolemund. *R for Data Science*. Available at: <https://r4ds.had.co.nz/>

3. Robin Lovelace, Jakub Nowosad, and Jannes Muenchow. *Geocomputation with R*. Available at: <https://r.geocompx.org/>

Récapitulatif

Bravo d'être arrivé à la conclusion de la leçon d'aujourd'hui !

Tout au long de notre parcours aujourd'hui, nous avons exploré :

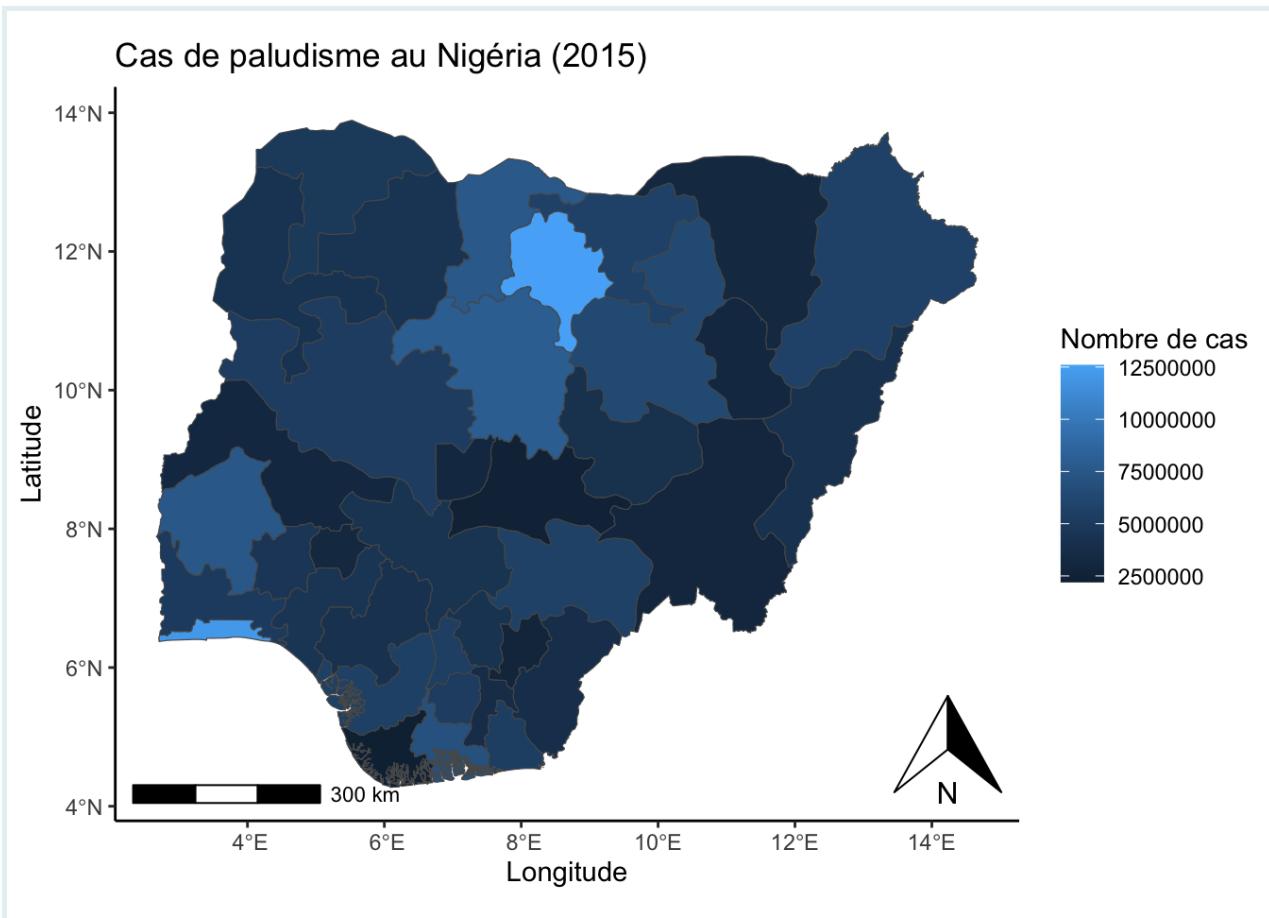
- Les concepts fondamentaux d'une carte choroplèthe et ses éléments essentiels.
- L'évaluation des avantages et des défis liés à l'utilisation des cartes choroplèthes.
- Les étapes pour organiser vos données de manière appropriée pour une carte choroplèthe.
- La maîtrise de l'art de créer une carte choroplèthe à l'aide de ggplot2.
- Les techniques pour appliquer des schémas de couleurs à la fois continus et discrets.
- Les méthodes pour produire de petits multiples en utilisant `facet_wrap()` et `facet_grid()`.

Vous êtes désormais armé de la compétence nécessaire pour créer des cartes choroplèthes distinctes en R à l'aide de ggplot2.

Solution

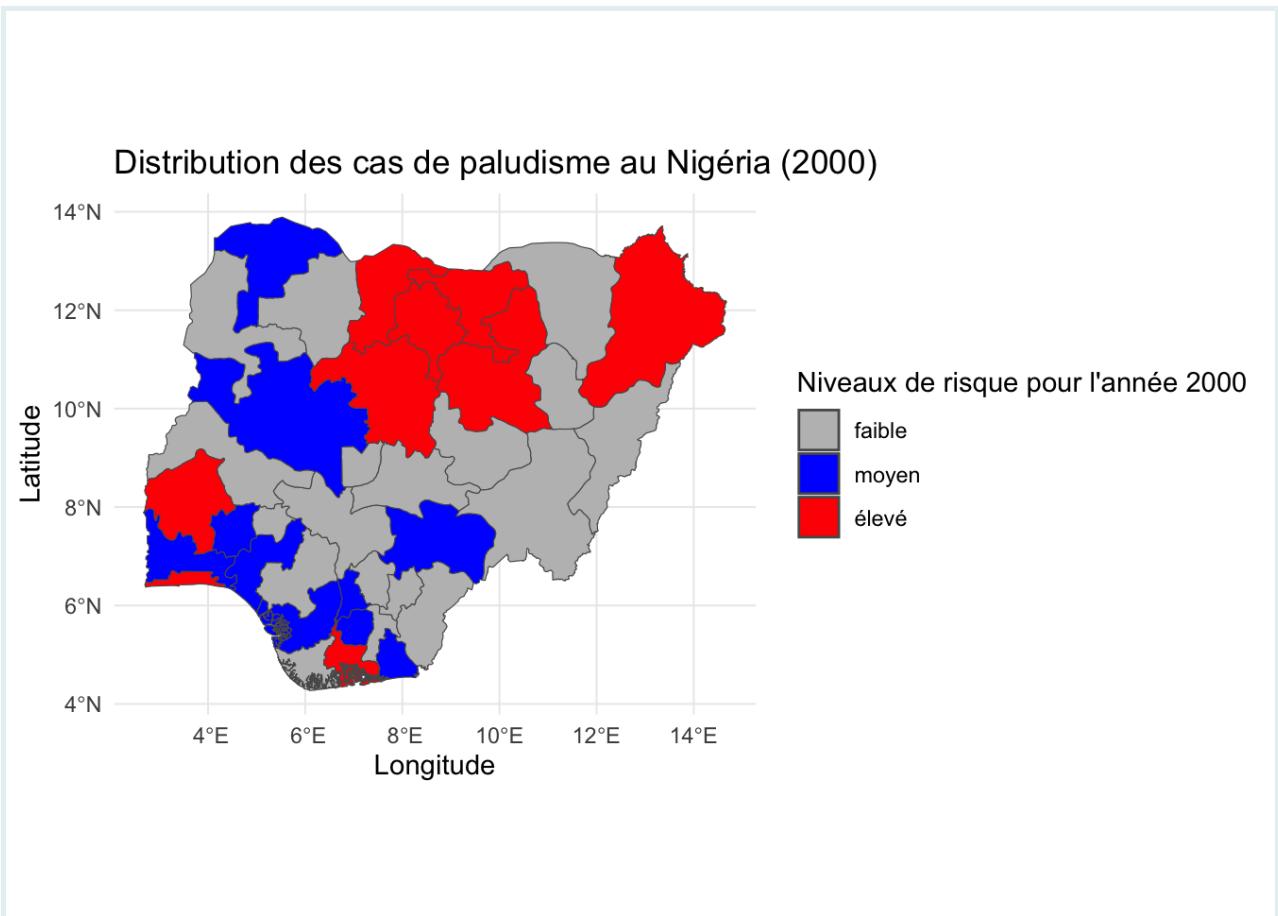
1.

```
ggplot(data=malaria2) +  
  geom_sf(aes(fill=cases_2015)) +  
  labs(title = "Cas de paludisme au Nigéria (2015)",  
       fill = "Nombre de cas",  
       x = "Longitude",  
       y = "Latitude") +  
  ggspatial::annotation_north_arrow(location = "br") +  
  ggspatial::annotation_scale(location = "bl") +  
  theme_classic()
```



2.

```
new_palette <- c("grey", "blue", "red") # Définissez un ensemble de couleurs
# différent
ggplot(data=malaria3) +
  geom_sf(aes(fill = fct_reorder(risk, cases_2000))) + # Réorganisez les
  # étiquettes de risque en fonction des cas de l'année 2000
  scale_fill_manual(values = new_palette, "Niveaux de risque pour l'année
  2000") +
  labs(title = "Distribution des cas de paludisme au Nigéria (2000)",
       fill = "Nombre de cas",
       x = "Longitude",
       y = "Latitude") +
  coord_sf(expand = TRUE) +
  theme_minimal()
```



3. Pour catégoriser les données en fonction de la médiane au lieu des quantiles, nous devons apporter une petite modification à la fonction `mutate()` où se produit la catégorisation du risque. Plus précisément, au lieu d'utiliser la fonction des quantiles, nous utiliserons la fonction de la médiane pour calculer la médiane des cas et diviser les données en risques "faibles" et "élevés" en fonction de cette valeur.

```

# Pivoter les données
malaria3_longer <- malaria2 %>%
  pivot_longer(cols = `cases_2000`:`cases_2021`, names_to = "year", values_to
  = "cases")

# Catégoriser le risque en fonction de la médiane
malaria3_longer %>%
  mutate(risk = case_when(
    cases <= median(cases) ~ 'faible',
    cases > median(cases) ~ 'élevé'
  )) -> malaria3_longer2

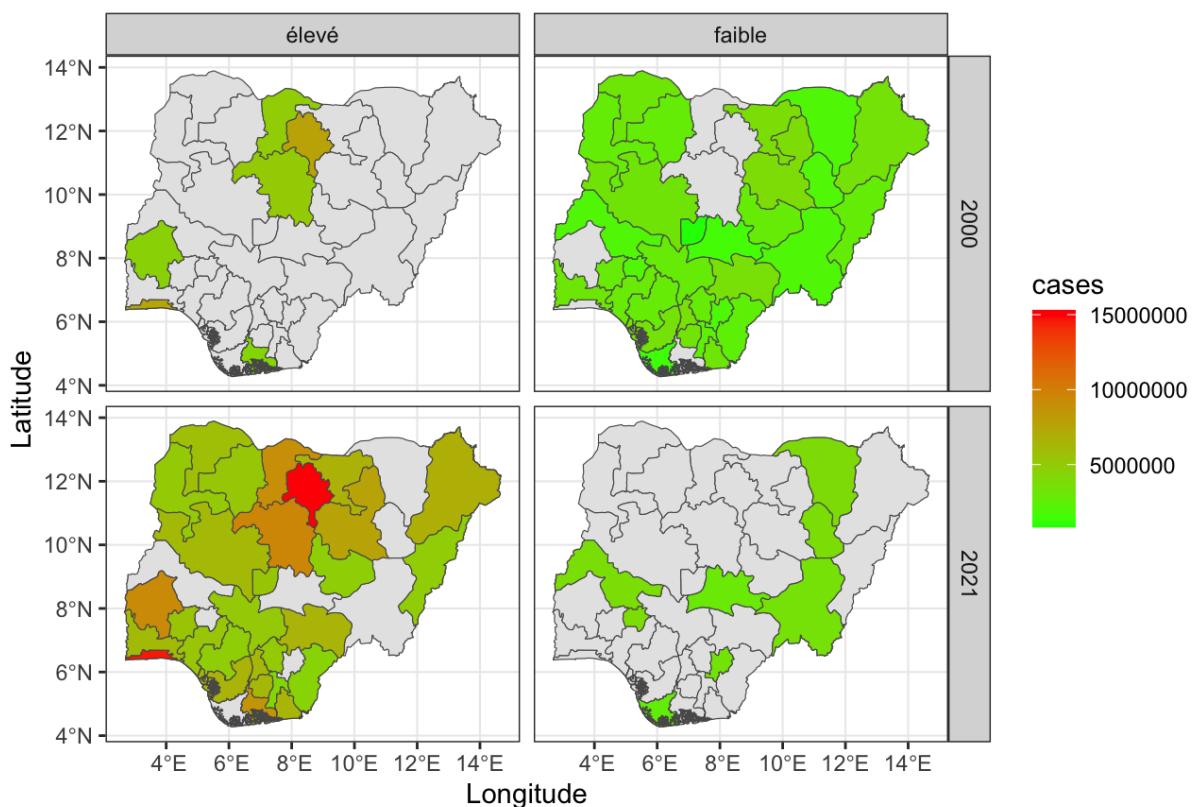
# Nettoyer les valeurs de l'année
malaria3_longer2$year <- str_replace(malaria3_longer2$year, "cases_", "")

# Tracer les données
ggplot() +
  geom_sf(data = nga_adm1) +
  geom_sf(aes(fill = cases), data = filter(malaria3_longer2, year %in%
    c("2000", "2021"))) +
  facet_grid(year ~ risk) +
  coord_sf(expand = TRUE) +
  scale_fill_continuous(low = "green", high = "red") +
  labs(title = "Distribution of Malaria cases and risk levels in Nigeria",
       subtitle = "2000 & 2021") +
  xlab("Longitude") +
  ylab("Latitude") +
  theme_bw()

```

Distribution of Malaria cases and risk levels in Nigeria

2000 & 2021



Contributeurs

Les membres de l'équipe suivants ont contribué à cette leçon :



IMAD EL BADISY

Data Science Education Officer
Deeply interested in health data



JOY VAZ

R Developer and Instructor, the GRAPH Network
Loves doing science and teaching science