

# Frequence\_TMJA

2025-12-15

```
#Quelques packages qu'on a utilisé ci-dessous  
library(dplyr)  # Pour manipuler les données
```

```
##  
## Attachement du package : 'dplyr'  
  
## Les objets suivants sont masqués depuis 'package:stats':  
##  
##      filter, lag  
  
## Les objets suivants sont masqués depuis 'package:base':  
##  
##      intersect, setdiff, setequal, union
```

```
library(MASS)    # Pour la loi Binomiale Négative (glm.nb)
```

```
##  
## Attachement du package : 'MASS'  
  
## L'objet suivant est masqué depuis 'package:dplyr':  
##  
##      select
```

```
library(ggplot2) # Pour les graphes  
library(readxl)  # pour lire un fichier .xls
```

## A) CSV “nettoyé”

Pour déterminer la fréquences des accidents avec TMJA, notre méthode est la suivante : on part des accidents individuels (données de l'ONISR) puis on les regroupe par ville (df\_communes) ensuite, on calcule TMJA moyen par département (df\_tmja\_dep) et ensuite on met tout dans df\_final

```
df_accidents <- read.csv("../data/ONISR-2021.csv",  
                          sep = ",",  
                          stringsAsFactors = FALSE)  
  
print(head(names(df_accidents)))
```

```
## [1] "Num_Acc" "jour"    "mois"    "an"      "hrmn"    "lum"
```

```
# --- 2. FONCTION DE NETTOYAGE ---
clean_dep <- function(d) {
  d <- as.character(d)
  # Si le code a 3 chiffres et finit par 0 (ex: 380), on garde les 2 premiers
  ifelse(nchar(d) == 3 & substr(d, 3, 3) == "0", substr(d, 1, 2), d)
}

df_clean <- df_accidents %>%
  mutate(dep_clean = clean_dep(dep))

# Vérification
print(unique(head(df_clean$dep_clean)))
```

```
## [1] "30" "51" "85" "93" "76" "68"
```

Ici on

## B) Filtre région Auvergne-Rhône-Alpes et dataset final

```
# 1. Liste des codes départements ARA
# On met "1" et "01" pour être sûr de tout attraper selon le formatage
lista_ARA <- c("1", "01", "3", "03", "7", "07", "15", "26", "38", "42", "43", "63", "69", "73", "74")

df_ARA <- df_clean %>%
  filter(as.character(dep_clean) %in% lista_ARA)
```

df\_ARA est df\_accidents et on a gardé uniquement le département Auvergne Rhône-Alpes

Ici, 1 ligne = 1 accident

```
library(dplyr)

df_communes <- df_ARA %>%
  group_by(dep_clean, com) %>%
  summarise(
    Nb_Accidents = n(),

    Part_Pluie = mean(grepl("pluie", atm, ignore.case = TRUE), na.rm = TRUE),

    Part_Neige = mean(grepl("neige", atm, ignore.case = TRUE), na.rm = TRUE),

    Part_Virage = mean(!grepl("rectiligne", plan, ignore.case = TRUE), na.rm = TRUE),

    Part_Nuit = mean(!grepl("plein jour", lum, ignore.case = TRUE), na.rm = TRUE),

    # Coordonnées pour la carte
    lat_center = mean(lat, na.rm = TRUE),
    long_center = mean(long, na.rm = TRUE),
```

```

    .groups = 'drop'
)

```

df\_communes (L'agrégation) ON change l'échelle, donc on regroupe tout par communes et on veut prédire le "risque par ville" (ex : avant 10 lignes pour 10 accidents à Grenoble mtn uniquement 1 ligne par accident à Grenoble )

Cela correspond à la colonne Nb\_Accidents dans l'entête

```
df_tmja_brut <- read_excel("tmja_2021_intranet.xls", sheet = 2)
```

```

## New names:
## * 'Sens' -> 'Sens...11'
## * 'Sens' -> 'Sens...16'
## * 'Année' -> 'Année...18'
## * 'TMJA TV' -> 'TMJA TV...19'
## * 'TMJA PL' -> 'TMJA PL...20'
## * '%PL' -> '%PL...21'
## * 'janvier TMJA TV' -> 'janvier TMJA TV...22'
## * 'février TMJA TV' -> 'février TMJA TV...23'
## * 'mars TMJA TV' -> 'mars TMJA TV...24'
## * 'avril TMJA TV' -> 'avril TMJA TV...25'
## * 'mai TMJA TV' -> 'mai TMJA TV...26'
## * 'juin TMJA TV' -> 'juin TMJA TV...27'
## * 'juillet TMJA TV' -> 'juillet TMJA TV...28'
## * 'août TMJA TV' -> 'août TMJA TV...29'
## * 'septembre TMJA TV' -> 'septembre TMJA TV...30'
## * 'octobre TMJA TV' -> 'octobre TMJA TV...31'
## * 'novembre TMJA TV' -> 'novembre TMJA TV...32'
## * 'décembre TMJA TV' -> 'décembre TMJA TV...33'
## * 'Année' -> 'Année...34'
## * 'TMJA TV' -> 'TMJA TV...35'
## * 'TMJA PL' -> 'TMJA PL...36'
## * '%PL' -> '%PL...37'
## * 'janvier TMJA TV' -> 'janvier TMJA TV...38'
## * 'février TMJA TV' -> 'février TMJA TV...39'
## * 'mars TMJA TV' -> 'mars TMJA TV...40'
## * 'avril TMJA TV' -> 'avril TMJA TV...41'
## * 'mai TMJA TV' -> 'mai TMJA TV...42'
## * 'juin TMJA TV' -> 'juin TMJA TV...43'
## * 'juillet TMJA TV' -> 'juillet TMJA TV...44'
## * 'août TMJA TV' -> 'août TMJA TV...45'
## * 'septembre TMJA TV' -> 'septembre TMJA TV...46'
## * 'octobre TMJA TV' -> 'octobre TMJA TV...47'
## * 'novembre TMJA TV' -> 'novembre TMJA TV...48'
## * 'décembre TMJA TV' -> 'décembre TMJA TV...49'
## * 'Année' -> 'Année...50'
## * 'TMJA TV' -> 'TMJA TV...51'
## * 'TMJA PL' -> 'TMJA PL...52'
## * '%PL' -> '%PL...53'
## * 'janvier TMJA TV' -> 'janvier TMJA TV...54'
## * 'février TMJA TV' -> 'février TMJA TV...55'
## * 'mars TMJA TV' -> 'mars TMJA TV...56'

```

```
## * 'avril TMJA TV' -> 'avril TMJA TV...57'
## * 'mai TMJA TV' -> 'mai TMJA TV...58'
## * 'juin TMJA TV' -> 'juin TMJA TV...59'
## * 'juillet TMJA TV' -> 'juillet TMJA TV...60'
## * 'août TMJA TV' -> 'août TMJA TV...61'
## * 'septembre TMJA TV' -> 'septembre TMJA TV...62'
## * 'octobre TMJA TV' -> 'octobre TMJA TV...63'
## * 'novembre TMJA TV' -> 'novembre TMJA TV...64'
## * 'décembre TMJA TV' -> 'décembre TMJA TV...65'
## * 'Année' -> 'Année...66'
## * 'TMJA TV' -> 'TMJA TV...67'
## * 'TMJA PL' -> 'TMJA PL...68'
## * '%PL' -> '%PL...69'
## * 'janvier TMJA TV' -> 'janvier TMJA TV...70'
## * 'février TMJA TV' -> 'février TMJA TV...71'
## * 'mars TMJA TV' -> 'mars TMJA TV...72'
## * 'avril TMJA TV' -> 'avril TMJA TV...73'
## * 'mai TMJA TV' -> 'mai TMJA TV...74'
## * 'juin TMJA TV' -> 'juin TMJA TV...75'
## * 'juillet TMJA TV' -> 'juillet TMJA TV...76'
## * 'août TMJA TV' -> 'août TMJA TV...77'
## * 'septembre TMJA TV' -> 'septembre TMJA TV...78'
## * 'octobre TMJA TV' -> 'octobre TMJA TV...79'
## * 'novembre TMJA TV' -> 'novembre TMJA TV...80'
## * 'décembre TMJA TV' -> 'décembre TMJA TV...81'
## * 'Année' -> 'Année...82'
## * 'TMJA TV' -> 'TMJA TV...83'
## * 'TMJA PL' -> 'TMJA PL...84'
## * '%PL' -> '%PL...85'
## * 'janvier TMJA TV' -> 'janvier TMJA TV...86'
## * 'février TMJA TV' -> 'février TMJA TV...87'
## * 'mars TMJA TV' -> 'mars TMJA TV...88'
## * 'avril TMJA TV' -> 'avril TMJA TV...89'
## * 'mai TMJA TV' -> 'mai TMJA TV...90'
## * 'juin TMJA TV' -> 'juin TMJA TV...91'
## * 'juillet TMJA TV' -> 'juillet TMJA TV...92'
## * 'août TMJA TV' -> 'août TMJA TV...93'
## * 'septembre TMJA TV' -> 'septembre TMJA TV...94'
## * 'octobre TMJA TV' -> 'octobre TMJA TV...95'
## * 'novembre TMJA TV' -> 'novembre TMJA TV...96'
## * 'décembre TMJA TV' -> 'décembre TMJA TV...97'
```

```
df_tmja_dep <- df_tmja_brut %>%
  dplyr::select(dep_source = 13, Valeur_TMJA = 19) %>%

  mutate(dep_clean = as.character(as.integer(dep_source))) %>%
  filter(dep_clean %in% lista_ARA) %>%
  group_by(dep_clean) %>%
  summarise(
    Trafic_Moyen = mean(Valeur_TMJA, na.rm = TRUE)
  )
#On fusionne
df_communes <- df_communes %>%
  left_join(df_tmja_dep, by = "dep_clean")
```

```
print("TMJA ajouté avec succès via les indices de colonnes !")
```

```
## [1] "TMJA ajouté avec succès via les indices de colonnes !"
```

```
#Df_final qu'on va utiliser  
df_final <- df_communes %>%  
  filter(!is.na(Trafic_Moyen))
```

df\_tmja\_brut correspond au fichier Excel téléchargés sur le site du TMJA

df\_tmja\_dep (le résumé) : comme on ne peut pas coller le trafic sur chaque rue, on a calculé la moyenne.

Ainsi, on a transformé l'Excel en un petit tableau simple :

- Département 38 (Isère) -> Trafic Moyen = X véhicules/jour.
- Département 15 (Cantal) -> Trafic Moyen = Y véhicules/jour.

## C) Test

A présent, que le dataset est bien fiable, organisé et bien propre, on va réaliser tous nos test sur df\_final

```
modele_poisson <- glm(Nb_Accidents ~ Part_Pluie + Part_Neige + Part_Virage + Part_Nuit +  
  offset(log(Trafic_Moyen)),  
  family = poisson,  
  data = df_final)  
  
summary(modele_poisson)
```

```
##  
## Call:  
## glm(formula = Nb_Accidents ~ Part_Pluie + Part_Neige + Part_Virage +  
##   Part_Nuit + offset(log(Trafic_Moyen)), family = poisson,  
##   data = df_final)  
##  
## Coefficients:  
##             Estimate Std. Error  z value Pr(>|z|)  
## (Intercept) -8.44337    0.02336 -361.417  < 2e-16 ***  
## Part_Pluie   0.07203    0.06252   1.152   0.2492  
## Part_Neige  -0.68477    0.28124  -2.435   0.0149 *  
## Part_Virage -1.23959    0.04738 -26.161  < 2e-16 ***  
## Part_Nuit    0.30797    0.04261   7.228 4.91e-13 ***  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## (Dispersion parameter for poisson family taken to be 1)  
##  
##    Null deviance: 10732.4  on 775  degrees of freedom  
## Residual deviance:  9810.8  on 771  degrees of freedom  
## AIC: 11887  
##  
## Number of Fisher Scoring iterations: 7
```

```
modele_binomiale_négative <- glm.nb(Nb_Accidents ~ Part_Pluie + Part_Neige + Part_Virage + Part_Nuit +
                                     offset(log(Trafic_Moyen)),
                                     data = df_final)
summary(modele_binomiale_négative)
```

```
##
## Call:
## glm.nb(formula = Nb_Accidents ~ Part_Pluie + Part_Neige + Part_Virage +
##       Part_Nuit + offset(log(Trafic_Moyen)), data = df_final, init.theta = 0.8156901657,
##       link = log)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.46881    0.06902 -122.699 < 2e-16 ***
## Part_Pluie   0.01178    0.17946   0.066   0.948
## Part_Neige  -0.49342    0.55006  -0.897   0.370
## Part_Virage -1.45946    0.10962 -13.314 < 2e-16 ***
## Part_Nuit    0.49355    0.12375   3.988 6.65e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(0.8157) family taken to be 1)
##
## Null deviance: 909.55  on 775  degrees of freedom
## Residual deviance: 758.25  on 771  degrees of freedom
## AIC: 3969.7
##
## Number of Fisher Scoring iterations: 1
##
##
##             Theta: 0.8157
##             Std. Err.: 0.0419
##
## 2 x log-likelihood: -3957.7290
```

```
aic_poisson <- AIC(modele_poisson)
aic_nb <- AIC(modele_binomiale_négative)
cat("AIC Poisson          :", round(aic_poisson, 2), "\n")
```

```
## AIC Poisson          : 11886.51
```

```
cat("AIC Binomiale Négative :", round(aic_nb, 2), "\n")
```

```
## AIC Binomiale Négative : 3969.73
```

A nouveau même à une échelle régionale le modèle Binomiale négative est plus intéressant que le modèle de Poisson

```
ratios <- exp(coef(modele_binomiale_négative))
print(ratios)
```

```
## (Intercept)    Part_Pluie    Part_Neige    Part_Virage    Part_Nuit
## 0.0002099136  1.0118516339  0.6105373790  0.2323609958  1.6381240249
```

D'une part, **La nuit** (coef 1.64) apparait comme un facteur aggravant majeur qui augmente le risque de 64% à trafic constant, en raison de la visibilité réduite et de la fatigue. D'autre part, **La neige**(0.61) et **les virages** (0.23) présentent des coefficients inférieurs à 1, ce qui suggère un effet "protecteur" contre-intuitif. Ce paradoxe apparent s'explique par une forte adaptation comportementale. Face à des conditions visiblement difficiles (routes de montagne, intempéries), les conducteurs augmentent leur vigilance et réduisent leur vitesse, alors que les environnements jugés "sûrs" favorisent le relâchement de l'attention.

## D) Aperçu visuel sur la carte de France

```
library(ggplot2)
library(dplyr)
library(stringr)
df_map_source <- df_ARA

if("latitude" %in% names(df_map_source)) df_map_source <- rename(df_map_source, lat = latitude)
if("longitude" %in% names(df_map_source)) df_map_source <- rename(df_map_source, long = longitude)

df_map_source$lat <- as.numeric(gsub(",", ".", as.character(df_map_source$lat)))
df_map_source$long <- as.numeric(gsub(",", ".", as.character(df_map_source$long)))

# Calcul des centres par communes
coords_communes <- df_map_source %>%
  filter(!is.na(lat) & !is.na(long)) %>%
  group_by(dep_clean, com) %>%
  summarise(
    lat_center = mean(lat, na.rm = TRUE),
    long_center = mean(long, na.rm = TRUE),
    .groups = 'drop'
  )
df_final_clean <- df_final %>%
  dplyr::select(-any_of(c("lat_center", "long_center")))

# On colle les nouvelles coordonnées propres
df_map <- df_final_clean %>%
  inner_join(coords_communes, by = c("dep_clean", "com"))

print(paste("Nombre de communes prêtes à être affichées :", nrow(df_map)))
```

```
## [1] "Nombre de communes prêtes à être affichées : 776"
```

```
# --- 4. LA CARTE (VIRAGES) ---
# Calcul du zoom automatique
min_x <- min(df_map$long_center, na.rm = TRUE) - 0.2
max_x <- max(df_map$long_center, na.rm = TRUE) + 0.2
min_y <- min(df_map$lat_center, na.rm = TRUE) - 0.2
max_y <- max(df_map$lat_center, na.rm = TRUE) + 0.2

ggplot(df_map, aes(x = long_center, y = lat_center)) +
```

```

# Fond de carte
borders("france", colour = "grey90", fill = "white") +

# Les points (Part_Virage)
geom_point(aes(size = Nb_Accidents, color = Part_Virage), alpha = 0.7) +

# Couleurs : Bleu (Plat) -> Rouge (Virage)
scale_color_gradient(low = "grey85", high = "#e31a1c", name = "% Virages",
                     limits = c(0, 1)) +

# Taille
scale_size_continuous(range = c(0.5, 6), name = "Vol. Accidents") +

# Zoom
coord_quickmap(xlim = c(min_x, max_x), ylim = c(min_y, max_y)) +

# Titres
labs(title = "Géographie du Risque : Virages vs Lignes Droites",
     subtitle = "Bleu = Plaine | Rouge = Montagne",
     x = "Longitude", y = "Latitude") +
theme_minimal()

```

```

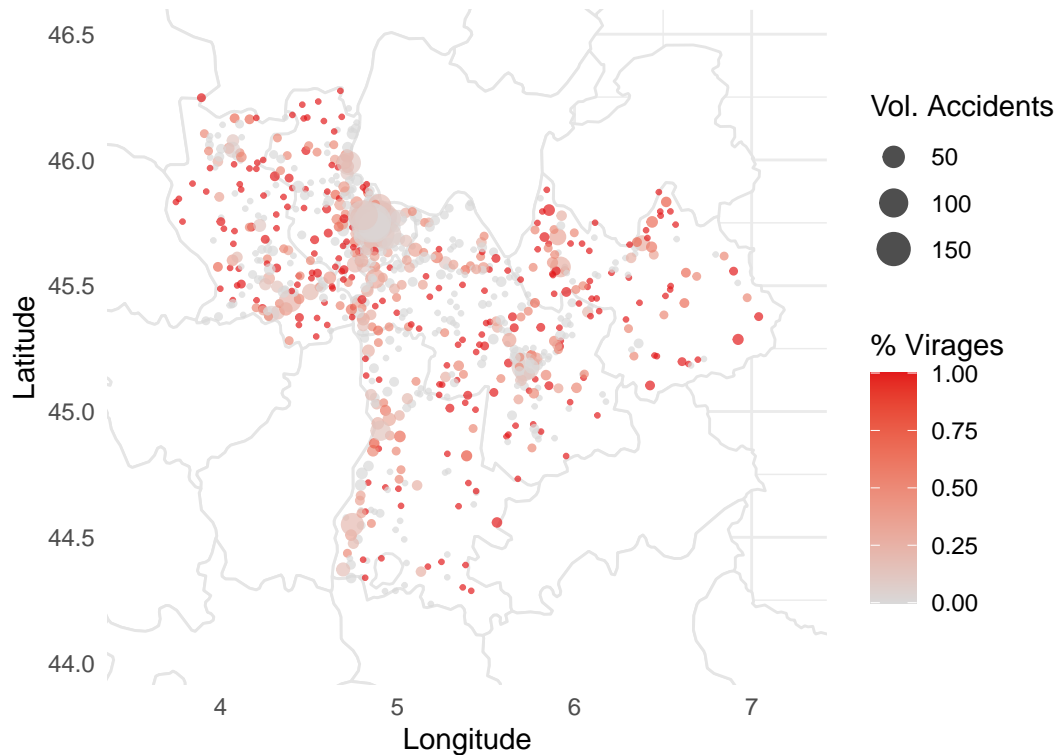
## Warning: 'borders()' was deprecated in ggplot2 4.0.0.
## i Please use 'annotation_borders()' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

```



## Géographie du Risque : Virages vs Lignes Droites

Bleu = Plaine | Rouge = Montagne



CARTE PART\_NUIT

```
ggplot(df_map, aes(x = long_center, y = lat_center)) +
  # Fond
  borders("france", colour = "grey90", fill = "white") +

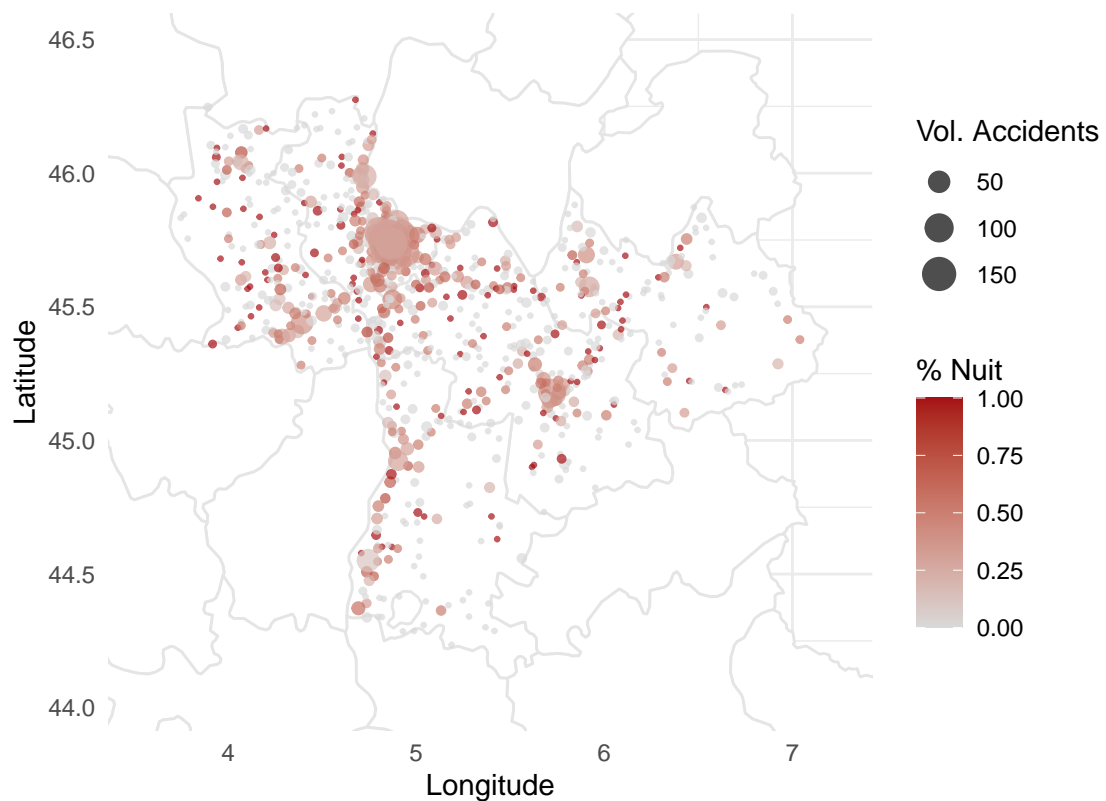
  geom_point(aes(size = Nb_Accidents, color = Part_Nuit), alpha = 0.7) +

  scale_color_gradient(low = "grey85", high = "#a50f15", name = "% Nuit",
    limits = c(0, 1)) +

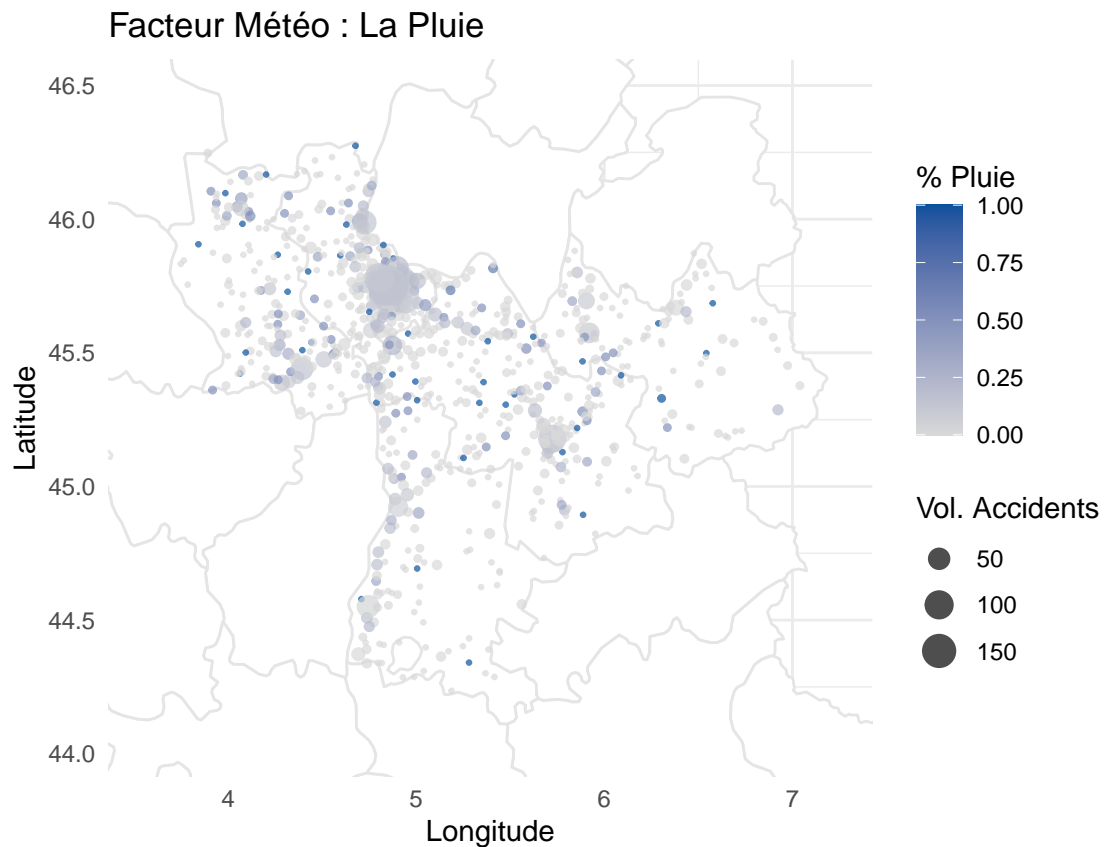
  scale_size_continuous(range = c(0.5, 6), name = "Vol. Accidents") +
  coord_quickmap(xlim = c(min_x, max_x), ylim = c(min_y, max_y)) +

  labs(title = "Géographie des horaires : Jour vs Nuit",
    x = "Longitude", y = "Latitude") +
  theme_minimal()
```

## Géographie des horaires : Jour vs Nuit



```
ggplot(df_map, aes(x = long_center, y = lat_center)) +  
  borders("france", colour = "grey90", fill = "white") +  
  
  geom_point(aes(size = Nb_Accidents, color = Part_Pluie), alpha = 0.7) +  
  
  scale_color_gradient(low = "grey85", high = "#08519c", name = "% Pluie",  
    limits = c(0, 1)) +  
  
  scale_size_continuous(range = c(0.5, 6), name = "Vol. Accidents") +  
  coord_quickmap(xlim = c(min_x, max_x), ylim = c(min_y, max_y)) +  
  
  labs(title = "Facteur Météo : La Pluie",  
    x = "Longitude", y = "Latitude") +  
  theme_minimal()
```



```
ggplot(df_map, aes(x = long_center, y = lat_center)) +
  borders("france", colour = "grey90", fill = "white") +

  geom_point(aes(size = Nb_Accidents, color = Part_Neige), alpha = 0.7) +

  scale_color_gradient(low = "grey85", high = "#00BFFF", name = "% Neige",
    limits = c(0, 1)) +

  scale_size_continuous(range = c(0.5, 6), name = "Vol. Accidents") +
  coord_quickmap(xlim = c(min_x, max_x), ylim = c(min_y, max_y)) +

  labs(title = "Facteur Météo : La Neige (Événement rare)",
    x = "Longitude", y = "Latitude") +
  theme_minimal()
```

