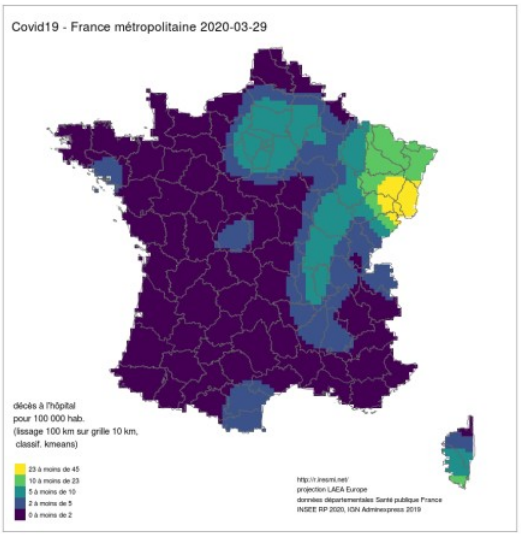
....




```
# Carto décès COVID 19 France
# avec lissage

# sources ------------------------------------------------------------------

fichier_covid <- "donnees/covid.csv"
url_donnees_covid <- "https://www.data.gouv.fr/fr/datasets/r/63352e38-d353-4b54-bfd1-f1b3ee1cabd7"

fichier_pop <- "donnees/pop.xls"
# https://www.insee.fr/fr/statistiques/2012713#tableau-TCRD_004_tab1_departements
url_donnees_pop <- "https://www.insee.fr/fr/statistiques/fichier/2012713/TCRD_004.xls"

# Adminexpress :
# https://geoservices.ign.fr/documentation/diffusion/telechargement-donnees-libres.html#admin-express


# config ------------------------------------------------------------------
library(tidyverse)
library(httr)
library(fs)
library(sf)
library(readxl)
library(janitor)
library(tmap)
# + btb, raster, fasterize, plyr

#' Kernel weighted smoothing with arbitrary bounding area
#'
#' @param df sf object (points)
#' @param field weight field in the df
#' @param bandwidth kernel bandwidth (map units)
#' @param resolution output grid resolution (map units)
#' @param zone sf study zone (polygon)
#' @param out_crs EPSG (should be an equal-area projection)
#'
#' @return a raster object
#' @import btb, raster, fasterize, dplyr, plyr, sf
lissage <- function(df, field, bandwidth, resolution, zone, out_crs = 3035) {

  if (st_crs(zone)$epsg != out_crs) {
    message("reprojecting data...")
    zone <- st_transform(zone, out_crs)
  }

  if (st_crs(df)$epsg != out_crs) {
    message("reprojecting study zone...")
    df <- st_transform(df, out_crs)
  }

  zone_bbox <- st_bbox(zone)

  # grid generation
  message("generating reference grid...")
  zone_xy <- zone %>%
    dplyr::select(geometry) %>%
    st_make_grid(cellsize = resolution,
                 offset = c(plyr::round_any(zone_bbox[1] - bandwidth, resolution, f = floor),
                            plyr::round_any(zone_bbox[2] - bandwidth, resolution, f = floor)),
                 what = "centers") %>%
```

```
    st_sf() %>%
    st_join(zone, join = st_intersects, left = FALSE) %>%
    st_coordinates() %>%
    as_tibble() %>%
    dplyr::select(x = X, y = Y)

  # kernel
  message("computing kernel...")
  kernel <- df %>%
    cbind(., st_coordinates(.)) %>%
    st_set_geometry(NULL) %>%
    dplyr::select(x = X, y = Y, field) %>%
    btb::kernelSmoothing(dfObservations = .,
                         sEPSG = out_crs,
                         iCellSize = resolution,
                         iBandwidth = bandwidth,
                         vQuantiles = NULL,
                         dfCentroids = zone_xy)

  # rasterization
  message("\nrasterizing...")
  raster::raster(xmn = plyr::round_any(zone_bbox[1] - bandwidth, resolution, f = floor),
                 ymn = plyr::round_any(zone_bbox[2] - bandwidth, resolution, f = floor),
                 xmx = plyr::round_any(zone_bbox[3] + bandwidth, resolution, f = ceiling),
                 ymx = plyr::round_any(zone_bbox[4] + bandwidth, resolution, f = ceiling),
                 resolution = resolution) %>%
    fasterize::fasterize(kernel, ., field = field)
}


# téléchargement-----------------------------------------------------------------

if (!file_exists(fichier_covid) |
    file_info(fichier_covid)$modification_time < Sys.Date()) {
  GET(url_donnees_covid,
      progress(),
      write_disk(fichier_covid, overwrite = TRUE))
}

if (!file_exists(fichier_pop)) {
  GET(url_donnees_pop,
      progress(),
      write_disk(fichier_pop))
}


# données ------------------------------------------------------------------

covid <- read_csv2(fichier_covid)

# adminexpress prétéléchargé
dep <- read_sf("donnees/ADE_2-0_SHP_LAMB93_FR/DEPARTEMENT.shp") %>%
  clean_names()

pop <- read_xls(fichier_pop, skip = 2) %>%
  clean_names()


# prétraitement ------------------------------------------------------------

fr <- dep %>%
  st_union() %>%
  st_sf() %>%
  st_set_crs(2154)

deces <- dep %>%
  left_join(pop, by = c("insee_dep" = "x1")) %>%
  left_join(covid %>%
              filter(jour == max(jour),
                     sexe == 0) %>%
              group_by(dep) %>%
              summarise(deces = sum(dc, na.rm = TRUE)),
            by = c("insee_dep" = "dep")) %>%
  st_point_on_surface() %>%
  st_set_crs(2154)


# lissage ------------------------------------------------------------------

d <- deces %>%
  lissage("deces", 100000, 10000, fr, 3035)

p <- deces %>%
  lissage("x2020_p", 100000, 10000, fr, 3035)

d100k <- d * 100000 / p
```

```
# carto ------------------------------------------------------------------

tm_layout(title = paste("Covid19 - France métropolitaine", max(covid$jour)),
          legend.position = c("left", "bottom")) +
  tm_shape(d100k) +
  tm_raster(title = "décès pour 100 000 hab.\n(lissage 100 km sur grille 10 km,\n classif.
kmeans)",
            style = "kmeans",
            palette = "viridis",
            legend.format = list(text.separator = "à moins de",
                                 digits = 0),
            legend.reverse = TRUE) +
  tm_shape(dep) +
  tm_borders() +
  tm_credits("http://r.iresmi.net/\nprojection LAEA Europe\ndonnées départementales Santé publique
France\nINSEE RP 2020, IGN Adminexpress 2019")
```