GEANES-USP-BR / chikungunya-INLAspacetime-spde

Code Issues Pull requests Actions Projects Security Insights

chikungunya-INLAspacetime-spde / Chik_SPDE_nbi_model.R

Rpalasio Update Chik_SPDE_nbi_model.R fb55229 · now

805 lines (675 loc) · 33.4 KB

Code Blame Raw

```
####  Spatio-temporal modeling of chikungunya in Brazil ####
#  Author: Raquel Gardini Sanches Palasio
#  Article: "Spatio-temporal Modeling of chikungunya in Brazil:
#            Interplay of Climate, Mobility, and Socioeconomic Factors"
#  Description: Implementation of a DEMF(1,2,1) SPDE model using INLAspacetime
# R version 4.3.3 (2024-02-29) -- "Angel Food Cake"

# ---- 1. Load Packages --------------------------------------------------------
# Core libraries
library(tidyverse)
library(dplyr)
library(ggplot2)
library(sf)
library(cowplot)
library(gridExtra)

# Spatial and Bayesian modeling
library(INLA)            # version 24.3.29
library(INLAspacetime)   # version 0.1.8
library(inlabru)         # version 2.10.1
library(fmesher)         # version 0.1.7
library(INLAOutputs)

# Mapping and shapefiles
library(geobr)           # for Brazilian municipalities and regions
library(ggspatial)       # to add north arrow and scale bar
library(ggsn)            # extra map tools


# ---- 2. Load and Prepare Data ------------------------------------------------

# The dataset 'data' should contain chikungunya incidence and covariates
# Variables include: time (month), space (municipality), Tmin, mobility index, etc.
# Dataset and dictionary available at [**Figshare (link to be inserted)**]
# Please download the dataset from Figshare and place it in the data/ folder before
# running the script.
```

```
data <- read_delim("data/df_ChiK.csv", delim = ";", escape_double = FALSE, trim_ws = TRUE)

# ---- 2.1 Load Shapefile of Brazilian Municipalities --------------------------

# Load 2022 municipal boundaries (from geobr)
MN_shp <- read_municipality(year = 2022, showProgress = FALSE)

# Adjust municipality code (remove last digit to match your dataset ID)
MN_shp$COD_MUN6 <- substr(MN_shp$code_muni, 1, nchar(MN_shp$code_muni) - 1)

# Remove Fernando de Noronha (not included in study area)
MN_shp <- MN_shp %>% filter(code_muni != 2605459)

# ---- 2.2 Prepare Spatial Data Frame ------------------------------------------

# Convert shapefile to dataframe (drop geometry)
BR.df <- MN_shp %>% st_drop_geometry()
N <- nrow(BR.df)

# Order by municipality code
BR.df <- BR.df[order(BR.df$COD_MUN6, decreasing = FALSE), ]

# Create spatial IDs for model indexing
BR.df$ID.space.1 <- seq(1, N)
shp.df <- as.data.frame(cbind(my.ID=1:N, ID.space.1=BR.df$ID.space.1, ID_MN_RESI=BR.df$COD
shp.df$ID_MN_RESI <- as.numeric(shp.df$ID_MN_RESI)

# ---- 2.3 Merge Spatial Identifiers with Main Data ----------------------------

# Add municipality and time identifiers to dataset
df <- left_join(data, shp.df[, c("my.ID", "ID_MN_RESI")], by = "ID_MN_RESI")

# Order dataset by time and municipality to match shapefile order
df <- arrange(df, ID.time.1, my.ID)

# ---- 2.4 Coordinate Transformation -------------------------------------------

# Convert to sf object using latitude and longitude
df <- st_as_sf(df, coords = c("LONG", "LAT"), crs = 4326, remove = FALSE)

# Define Albers Equal-Area projection for South America (ESRI:102033)
albers_km <- "+proj=aea +lat_0=-32 +lon_0=-60 +lat_1=-5 +lat_2=-42 +x_0=0 +y_0=0 +ellps=au

# Transform coordinates to projected CRS (in kilometers)
df <- st_transform(df, crs = albers_km)

# Extract UTM coordinates (for mesh creation)
df$xUTM <- st_coordinates(df)[, 1]
df$yUTM <- st_coordinates(df)[, 2]
```

```r
# ---- 2.5 Prepare Coordinates for Mesh ----------------------------------------

# Create a dataframe of unique coordinates for mesh construction
coords.cv <- data.frame(unique(st_coordinates(df)))


# ---- 3. Mesh Construction ----------------------------------------------------
# Create SPDE mesh using the actual study border

# Load border shapefile (projected CRS: Albers South America)
border <- read_sf("data/borda_102033.shp")

# Transform CRS to match UTM coordinates of centroids
border <- st_transform(border, crs = albers_km)

# Remove unnecessary columns (e.g., FID)
border <- border[, !(names(border) %in% c("FID"))]

# Create a spatial mesh with small scale
smesh <- inla.mesh.2d(
  loc = cbind(coords.cv[, 1], coords.cv[, 2]),  # spatial points
  boundary = st_as_sfc(border),                  # study area boundary
  max.edge = c(10, 350),
  offset = c(100, 350),
  cutoff = 35
)

# Check mesh summary and manifold structure
print(fm_manifold(smesh, c("S", "R")))

# Number of mesh vertices
smesh$n

# Plot mesh with border and data points
mesh_plot <- ggplot() +
  gg(smesh) +
  geom_point(
    data = data.frame(coords.cv),
    aes(x = X, y = Y),
    size = 0.2,
    colour = "red"
  ) +
  theme_minimal()

# Save figure (high-resolution)
ggsave(
  filename = "FigS9_mesh.jpeg",
  plot = mesh_plot,
  dpi = 1000,
  width = 8,
  height = 6,
  units = "in"
```

```r
)

# ---- 4. Prepare Climate and Covariates for Modeling -------------------------

# ---- 4.1 Create Non-linear Climate Groups ----------------------------------

# Group continuous climate variables using quantiles
df$Tmin.grp  <- inla.group(df$Tmin, n = 25, method = "quantile")        # minimum temperat
df$Prec.grp  <- inla.group(df$Prec / 1000, n = 25, method = "quantile") # precipitation (m
df$Rtemp.grp <- inla.group(df$Rtemp, n = 25, method = "quantile")       # relative tempera

# Note: Prec.grp and Rtemp.grp tested but not used in final model

# ---- 4.2 Select Covariates -------------------------------------------------

xnames <- c(
  "sewerage", "piped", "Tmin", "Rtemp", "Forest", "NDVI",
  "Infra", "logKatz", "Tmin.grp", "Prec.grp", "Rtemp.grp"
)

# ---- 4.3 Create Modeling Data Frame ----------------------------------------

dataf <- data.frame(
  df[c(
    "id_key_time", "ID_MN_RESI", "xUTM", "yUTM",
    "ID.time.1", "ID.space.time", "ID.space.1",
    xnames
  )],
  pop10      = df$pop / 10000,        # population in 10k units
  Prec       = df$Prec / 1000,        # precipitation in meters
  ALT.fc     = factor(df$ALT.fc, levels = c(1, 2, 3, 4)),
  BIO.fc     = factor(df$BIO.fc, levels = c(1, 2, 3, 4, 5, 6)),
  Enso.fc2   = factor(df$Enso.fc2, levels = c(1, 2, 3, 4, 5, 6)),
  Reg.fc     = factor(df$Reg.fc, levels = c(1, 2, 3, 4, 5)),
  sewerage.fc = factor(df$sewerage.fc, levels = c(1, 2, 3)),
  piped.fc    = factor(df$piped.fc, levels = c(1, 2, 3)),
  y = as.numeric(df$cases.chik)
)

# Remove geometry column if present
dataf <- dataf[, !(names(dataf) %in% c("geometry"))]

# Quick summary including missing values
summary(dataf, useNA = "always")



# ---- 5. Define Space-Time Model ---------------------------------------------

# ---- 5.1 Temporal Mesh ------------------------------------------------------

nt <- max(dataf$ID.time.1)  # total number of time points (10 year * 4 season)
h  <- 1                      # temporal resolution (1 season)
```

```r

# Create 1D temporal mesh
tmesh <- inla.mesh.1d(
  loc    = seq(1, nt + h/2, h),  # sequence of time points
  degree = 1                      # linear basis
)

# Check number of nodes in temporal mesh
tmesh$n


# ---- 5.2 Space-Time Model Definition --------------------------------------

stmodel <- stModel.define(
  smesh = smesh,          # spatial mesh (from section 3)
  tmesh = tmesh,          # temporal mesh
  model = '121',          # model type (see reference paper, Lindgren et al. 2024)
  control.priors = list(
    prs    = c(5, 0.1),   # P(spatial range < 1) = 0.1
    prt    = c(2, 0),     # temporal range fixed to 5
    psigma = c(1, 0.1)    # P(sigma > 1) = 0.1
  )
)

# ---- 5.3 Define Model Formula ------------------------------------------------

# Model formula: includes fixed effects, nonlinear effects, categorical factors,
# offset for population, and spatiotemporal random field
M <- ~ -1 + Intercept(1) + myoffset(log(pop10), model = "offset") +
  sew(sewerage.fc, model ="factor_contrast") +  # Socioeconomic: sewerage
  pip(piped.fc, model ="factor_contrast") +     # Socioeconomic: piped water
  Tmin.grp(Tmin.grp, model= 'rw2') +            # Climate: min temperature (nonlinear)
  Prec +                                        # Climate: precipitation
  logKatz +                                     # Mobility: Katz index (Log)
  Forest +                                      # Environmental: forest cover
  Alt(ALT.fc, model = "factor_contrast") +      # Elevation factor
  Biome(BIO.fc, model ="factor_contrast") +     # Biome categorical factor
  field(list(
    space = cbind(xUTM, yUTM),              # Spatial coordinates
    time = ID.time.1),                      # Temporal index
    model = stmodel)                        # Spatiotemporal model defined earlier


# ---- 6. Define Likelihood ---------------------------------------------------
lhood <- like(
  formula = y ~ .,          # Response variable is count of cases (y)
  family = "nbinomial",     # Negative binomial model for overdispersed counts
  data = dataf
)

# ---- 7. Define Computation Options ------------------------------------------
ctrc <- list(
```

```r
    waic = TRUE,  # Compute WAIC
    dic = TRUE,   # Compute DIC
    cpo = TRUE    # Compute CPO for model checking
  )

  # ---- 8. Fit the Model -------------------------------------------------------
  fit121 <- bru(
    M,       # Model formula
    lhood,   # Likelihood
    options = list(
      control.compute = ctrc,  # Specify computation options
      verbose = FALSE          # Suppress detailed output during fitting
    )
  )

  #Full model (not uploaded due to file size)
  #save(fit121, file = "data/Results_models_chik.Rdata")

  #To make it accessible on GitHub, the model was split and reduced into two files:
  load("data/fit121_reduced.RData")
  load("data/fit121_reduced_map.RData")

  fit121 <- fit121_reduced
  fit121$summary.fitted.values <- fit121_reduced2$summary.fitted.values

  # ---- 9. Summarize Model Results --------------------------------------------
  # Summary of posterior marginal distributions for the fixed effects
  summary(fit121)

  # Compute model fit statistics: WAIC, DIC, and mean CPO
  round(c(fit121$waic$waic, fit121$dic$dic, -mean(log(fit121$cpo$cpo), na.rm=T)), 3)

  # Posterior Distributions of Hyperparameters
  # Avoid scientific notation for easier reading
  options(scipen = 999)

  # Posterior summaries for hyperparameters (e.g., spatial/temporal variance)
  round(fit121$summary.hyperpar, 2)

  # Posterior Summaries of Fixed Effects
  # Columns: mean, sd, 0.025 quantile, 0.975 quantile
  round(cbind(fit121$summary.fixed[, c(1, 2, 3, 5)]), 3)

  # ---- 10. Prepare Table for Plot --------------------------------------------
  # Fixed effects table
  a <- data.frame(FixedEffects(fit121), fix.empty.names = FALSE)

  var <- data.frame(
    ID = 1:4,
    mean = a$mean,
    X0.025quant = a$qt0.025,
    X0.975quant = a$qt0.975,
```

```
    type = c("-", "Climate", "Mobility", "Environmental"),
    Covariates = c("Intercept", "Precipitation total (m)", "Katz index", "Forest")
  )  %>% rename("0.025quant" = X0.025quant, "0.975quant" = X0.975quant)


  # Elevation
  Elevation <- cbind(
    fit121$summary.random$Alt$ID,
    round(exp(fit121$summary.random$Alt[, c("mean", "0.025quant", "0.975quant")]), 2)
  ) %>%  rename( 'ID' = `fit121$summary.random$Alt$ID`)
  linha <- data.frame(ID=1, mean=1, `0.025quant`=1, `0.975quant`=1)
  linha <-rename (linha, `0.025quant` = X0.025quant ,  `0.975quant` = X0.975quant)

  Elevation <- rbind(linha, Elevation)
  Elevation$type <- c("Ref.", "Elevation", "Elevation", "Elevation")
  Elevation$Covariates <- c("900-1601 (Ref.)", "600-899 (m)", "250-599 (m)", "3-249 (m)")

  # Sewerage
  sew <-  cbind(fit121$summary.random$sew$ID,
         round(exp(fit121$summary.random$sew [,c( "mean","0.025quant","0.975quant")]),2))
  sew <-rename (sew, 'ID' = `fit121$summary.random$sew$ID`    )
  sew <- rbind(linha, sew)

  sew$type <- c("Ref.",'Socioeconomic', 'Socioeconomic')
  sew$Covariates <- c ("sewerage <=0.3 (Ref.)","sewerage>0.3 & <=0.6", "sewerage >0.6")

  # Piped water
  pip <- cbind(fit121$summary.random$pip$ID,
        round(exp(fit121$summary.random$pip [,c( "mean","0.025quant","0.975quant")]),2))
  pip <-rename (pip, 'ID' = `fit121$summary.random$pip$ID`    )
  pip <- rbind(linha, pip)
  pip$type <- c("Ref.",'Socioeconomic', 'Socioeconomic')
  pip$Covariates <- c ("piped <=0.3 (Ref.)","piped >0.3 & <=0.6", "piped >0.6")

  # Biome
  Biome <-  cbind(fit121$summary.random$Biome$ID,
         round(exp(fit121$summary.random$Biome [,c( "mean","0.025quant","0.975quant")]),2))
  Biome <-rename (Biome, 'ID' = `fit121$summary.random$Biome$ID`    )
  Biome <- rbind(linha, Biome)

  Biome$type <- c("Ref.",'Biome', 'Biome','Biome', 'Biome','Biome')
  Biome$Covariates <- c ("Mata Atlantica (Ref.)","Pantanal", "Pampa", "Amazonia", "Cerrado",

  # Combine all tables
  tab <- rbind(pip, sew, Elevation, Biome, var)
  tab <-rename (tab, ll = '0.025quant'  )
  tab <-rename (tab, ul = '0.975quant'  )

  tab$type <- factor(tab$type,
           levels = c("-", "Socioeconomic", "Mobility", "Environmental", "Climate", "Elevati
  tab <- tab %>% arrange(type)

```

```r
tab$Covariates <- factor(tab$Covariates, levels = rev(c(
  "Intercept",
  "sewerage <=0.3 (Ref.)",
  "sewerage>0.3 & <=0.6",
  "sewerage >0.6",
  "piped <=0.3 (Ref.)",
  "piped >0.3 & <=0.6",
  "piped >0.6",
  "Katz index",
  "Forest",
  "Precipitation total (m)",
  "900-1601 (Ref.)",
  "600-899 (m)",
  "250-599 (m)",
  "3-249 (m)",
  "Mata Atlantica (Ref.)",
  "Pantanal",
  "Pampa",
  "Amazonia",
  "Cerrado",
  "Caatinga"
)))

tab <- tab %>% filter(Covariates != "Intercept")

# ---- 11. Plot IRR with 95% CI ---------------------------------------------

Figplot <- ggplot(tab, aes(x = mean, y = Covariates, color = type, shape = type)) +
  geom_point(size = 3) +
  geom_errorbarh(aes(xmin = ll, xmax = ul), height = 0.2) +
  geom_vline(xintercept = 1, linetype = "dashed", color = "gray") +
  labs(
    x = "IRR (95% CI)",
    y = "Covariates") +
  scale_color_manual( "",
    values = c( "orange", "black","green3","blue","red", "brown",  "grey")
      ) +
  scale_shape_manual( "",
    values = c(16,16,16, 16, 16,16,17)) +
  theme_minimal() +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    axis.line.x = element_line(color = "black"),
    axis.line.y = element_line(color = "black"),
    axis.text.y = element_text(size = 8, color = "black"),
    legend.position =  'bottom',
    legend.box = "horizontal",
    legend.spacing.y = unit(0.0, "cm"),
    legend.spacing.x = unit(0.5, "cm"),
    legend.title = element_blank(),
    legend.text = element_text(size = 10),
```

```r
      legend.key.width = unit(0.5, "cm"),
      legend.margin = margin(0, 0, 0, 0)
  ) +
    guides(
      color = guide_legend(nrow = 1),
      shape = guide_legend(nrow = 1)
    )

  # Extract legend separately
  legend = cowplot::get_plot_component(Figplot, 'guide-box-bottom', return_all = TRUE)
  cowplot::ggdraw(legend)

  # Plot without legend
  Figplot <- ggplot(tab, aes(x = mean, y = Covariates, color = type, shape = type)) +
    geom_point(size = 3) +  # Define point size
    geom_errorbarh(aes(xmin = ll, xmax = ul), height = 0.2) +  # Horizontal error bars
    geom_vline(xintercept = 1, linetype = "dashed", color = "gray") +  # Reference line at R
    labs(
      x = "RR (95% CI)",
      y = "Covariates"
    ) +
    scale_color_manual(
      "",  # Remove legend title
      values = c("orange", "black", "green3", "blue", "red", "brown", "grey")  # Manual colo
    ) +
    scale_shape_manual(
      "",
      values = c(16, 16, 16, 16, 16, 16, 17)  # Manual shapes
    ) +
    theme_minimal() +
    theme(
      panel.grid.major = element_blank(),  # Remove major grid lines
      panel.grid.minor = element_blank(),  # Remove minor grid lines
      axis.line.x = element_line(color = "black"),  # X-axis line
      axis.line.y = element_line(color = "black"),  # Y-axis line
      axis.text.y = element_text(size = 8, color = "black"),
      legend.position = ""  # Remove legend
    )

  # ---- 12. Plot Tmin effect ------------------------------------------------------
  data <- fit121$summary.random$Tmin.grp[,c( "ID", "mean","0.025quant","0.975quant")]

  # Extract Tmin random effect summary
  Tmin_max <- fit121$summary.random$Tmin.grp %>% filter(mean==max(mean))
  Tmin_max_value <- round(Tmin_max$ID, 0)

  FigTmin <-ggplot(data, aes(x = ID)) +
    geom_ribbon(aes(ymin = exp(`0.025quant`), ymax = exp(`0.975quant`)), fill = "lightblue",
    geom_line(aes(y = exp(`0.025quant`)), color = "blue", linetype = "dashed") +
    geom_line(aes(y = exp(`0.975quant`)), color = "blue", linetype = "dashed") +
    geom_line(aes(y = exp(mean)), color = "blue", linetype = "solid", size = 0.7) +
    geom_hline(yintercept = 1, color = "grey", linetype = "dashed") +
```

```r
    geom_vline(xintercept = Tmin_max_value, color = "red", linetype = "dashed") +
    scale_x_continuous(
      breaks = seq(9, 25, by = 2),
      labels = seq(9, 25, by = 2)  ) +
    labs(
      x = "Minimum temperature (°Celsius)",
      y = "RR (95% CI)"
    ) +
    theme_classic() +
    theme(
      axis.text = element_text(size = 9),
      axis.title = element_text(size = 9),
      axis.line.x = element_line(color = "black"),  # Adiciona linha no eixo X
      axis.line.y = element_line(color = "black"),  # Adiciona linha no eixo Y
      panel.grid.major = element_blank(),  # Remove linhas de grade principais
      panel.grid.minor = element_blank(),  # Remove linhas de grade secundárias
      )


  # Combine plots with layout
  layout.matrix <- matrix(c(1, 3, 2, 3), nrow = 2, ncol = 2)
  layout(mat = layout.matrix,
         heights = c(4, 1), # Heights of the two rows
         widths = c(1.5, 2)) # Widths of the two columns

  # Add label "A" to Figplot
  Figplot2 <- Figplot +
    annotate("text", x = 1.8, y = 18.5, label = "A",
             hjust = 0, vjust = 0, size = 6, fontface = "bold", color = "black")
  # Add label "B" to FigTmin
  FigTmin2 <- FigTmin +
    annotate("text", x = max(data$ID), y = max(exp(data$mean)), label = "B",
             hjust = 1, vjust = -1.3, size = 6, fontface = "bold", color = "black")
  # Arrange plots and legend using gridExtra
  layout <-grid.arrange(
    Figplot2, FigTmin2, legend,
    widths = c(1.5, 2), heights = c(13, 1),
    layout_matrix = rbind(c(1, 2),
                          c(3, 3))
  )

  # Save combined figure as PNG
  ggsave("Fig5_grafico_RR_CI.png",  layout, width = 19, height = 11,  unit="cm", dpi = 1000)

  # ---- 13. Prepare dataset for mapping ------------------------------
  rnams <- c( paste0(
      "summary.", c("fixed", "hyperpar", "random", "fitted.values")
    ), "internal.marginals.hyperpar", "mlik", "mode", "cpu.used")

  ndata <- nrow(dataf)
  sres <- fit121[rnams]
  sres$summary.fitted.values <- fit121$summary.fitted.values[1:ndata, c(1, 2)]
```

```r

# Compute posterior mean incidence per 10,000 population
dataf$pred_mean = fit121$summary.fitted.values[1:ndata, "mean"] / dataf$pop10
# Compute lower limit of 95% credible interval
dataf$pred_ll = fit121$summary.fitted.values[1:ndata, "0.025quant"] / dataf$pop10
# Compute upper limits of the 95% credible interval
dataf$pred_ul = fit121$summary.fitted.values[1:ndata, "0.975quant"] / dataf$pop10

# Export results table for QGIS mapping (Figure 6 in the paper)
#write.table(dataf, "Result_model.csv", sep=";", dec = ".", row.names = FALSE, na="")

# ---- 14. Prepare region shapefile ---------------------------------------------
# Load Brazilian regions (year 2020)
reg <- read_region(year = 2020,showProgress = FALSE)

# Convert multipolygons to individual polygons
reg_single <- reg %>%
  st_cast("POLYGON") %>%
  mutate(area = st_area(st_geometry(.)))

# Keep only the largest polygon for each region
reg_main <- reg_single %>%
  group_by(code_region) %>%   # change 'code_region' to 'code_muni' if working with munici
  slice_max(order_by = area, n = 1) %>%
  ungroup() %>%
  select(-area)

# ---- 15. Map: Posterior Mean of Predicted Cases --------------------------------
dw <-dataf %>% select (id_key_time, ID_MN_RESI, pred_mean) %>%
  reshape(timevar = "id_key_time",
          idvar = "ID_MN_RESI", direction = "wide")

map <- merge(MN_shp[,c("COD_MUN6", "name_muni", "abbrev_state", "name_region")], dw, by.x
map_sf <- st_as_sf(map)

map_sf <- gather(map_sf, id_key_time, pred_mean, paste0("pred_mean.", 1:4, "_", rep(2015:2
map_sf$id <- gsub("pred_mean.", "", map_sf$id_key_time)
map_sf <- map_sf  %>% separate(id , c("season", "year"), sep="_")

map_sf$season [map_sf$season =="1"] <- "Summer"
map_sf$season [map_sf$season =="2"] <- "Autumn"
map_sf$season [map_sf$season =="3"] <- "Winter"
map_sf$season [map_sf$season =="4"] <- "Spring"
map_sf$season <- factor(map_sf$season , levels = c("Summer", "Autumn", "Winter","Spring" )

map_sf$id <- paste(map_sf$season, map_sf$year)
map_sf$id <- factor(map_sf$id,
                    levels =  c("Summer 2015","Summer 2016","Summer 2017","Summer 2018","S
                                "Autumn 2015","Autumn 2016","Autumn 2017","Autumn 2018","A
                                "Winter 2015","Winter 2016","Winter 2017","Winter 2018","W
                                "Spring 2015","Spring 2016","Spring 2017","Spring 2018","S
                    ))
```

```r

# Categorize predicted mean values by quantiles
map_sf$pred_mean <- as.numeric(map_sf$pred_mean)
quantile(map_sf$pred_mean)
summary(map_sf$pred_mean)
map_sf$quantil <- NA

map_sf$quantil[map_sf$pred_mean > 0 & map_sf$pred_mean < 0.2] <- "0.01-0.19"
map_sf$quantil[map_sf$pred_mean >= 0.2 & map_sf$pred_mean < 1] <- "0.2-0.9"
map_sf$quantil[map_sf$pred_mean >= 1 & map_sf$pred_mean < 3] <- "1.0-2.9"
map_sf$quantil[map_sf$pred_mean >= 3 & map_sf$pred_mean < 20] <- "3.0-19.9"
map_sf$quantil[map_sf$pred_mean >= 20] <-">20"

map_sf$quantil <- factor(map_sf$quantil,
                         levels =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20"))
table(map_sf$quantil)

# Create ggplot map
ggp <- ggplot() +
  geom_sf(data = map_sf, aes(fill = quantil), color = "transparent") +
  geom_sf(data = reg_main, fill = "transparent", aes(color = "Brazil´s regions"), linewidt
  theme_classic() +
  scale_fill_manual(name = "", values= c("#0277BD",'#abdda4', '#ffffbf', '#fdae61', "red")
                    limits =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20")) +
  scale_color_manual( name = "",  values = c("Brazil´s regions" = "black") ) +
  guides(
   fill = guide_legend(order = 1,
      label.theme = element_text(size = 8),# Tamanho do texto menor
      keywidth = unit(0.3, "cm"),# Tamanho do ícone menor
      keyheight = unit(0.3, "cm"),# Tamanho do ícone menor
      nrow = 1),# Divide a legenda em duas linhas
   color = guide_legend( order = 2,
      override.aes = list(fill = NA, linetype = 1, size = 0.1),
      label.theme = element_text(size = 8),
      keywidth = unit(0.28, "cm"),
      keyheight = unit(0.28, "cm"),
      nrow = 1)) +
    theme(legend.text = element_text(size = 10, face = "bold"),
        legend.direction = "horizontal",
        legend.position = "top", #"bottom",
        legend.box = "horizontal",         # ensures one block
        legend.margin = margin(0, 0, 0, 0),
        text = element_text(size = 10),
        axis.line = element_blank(),  # Remove as linhas dos eixos
        axis.text = element_blank(),  # Remove os rótulos dos eixos
        axis.ticks = element_blank(),
        strip.text = element_text(size = 4), #color = "dark green"),
        panel.spacing = unit(0.01, "lines"),# Change spacing between facets on both axis
        panel.spacing.x = unit(0.01, "lines"),# Change horizontal spacing between facets
        panel.spacing.y = unit(0.01, "lines")) +# Change vertical spacing between facets
  facet_wrap(~ id, ncol = 10 ) +
  labs(title = "",
```

```r
        x = "",
        y = "") +
  blank() +
  scalebar(map_sf, dist = 1000, dist_unit = "km", st.size = 1.1,
           height=0.08, border.size = 0.1,
           st.dist = 0.1,  # Ajuste a distância vertical do texto da escala
           anchor = c(x = -35, y = -38),
           transform = TRUE, model = "WGS84",
           facet.var = c("id"),  #faz ficar soh no ultimo
           facet.lev = c("Spring 2024"))

# Save map as JPEG
jpeg("FigS6_map_pred_mean_all.jpeg", width = 17, height = 10,
     units = "cm", res = 2000, type = "cairo"
    )
ggp
north2(ggp = ggp, scale = 0.065, x = 0.88, y = 0.04, symbol = 10)
dev.off()

# ---- 16. Lower Limit of Predicted Cases (95% CI) -----------------------------
dw_ll <-dataf %>% select (id_key_time, ID_MN_RESI, pred_ll) %>%
  reshape(timevar = "id_key_time",
          idvar = "ID_MN_RESI", direction = "wide")

map <- merge(MN_shp[,c("COD_MUN6", "name_muni", "abbrev_state", "name_region")], dw_ll, by
map_sf <- st_as_sf(map)

map_sf <- gather(map_sf, id_key_time, pred_ll, paste0("pred_ll.", 1:4, "_", rep(2015:2024,
map_sf$id <- gsub("pred_ll.", "", map_sf$id_key_time)
map_sf <- map_sf  %>% separate(id , c("season", "year"), sep="_")

map_sf$season [map_sf$season =="1"] <- "Summer"
map_sf$season [map_sf$season =="2"] <- "Autumn"
map_sf$season [map_sf$season =="3"] <- "Winter"
map_sf$season [map_sf$season =="4"] <- "Spring"
map_sf$season <- factor(map_sf$season , levels = c("Summer", "Autumn", "Winter","Spring" )

map_sf$id <- paste(map_sf$season, map_sf$year)
map_sf$id <- factor(map_sf$id,
                    levels =  c("Summer 2015","Summer 2016","Summer 2017","Summer 2018","S
                                "Autumn 2015","Autumn 2016","Autumn 2017","Autumn 2018","A
                                "Winter 2015","Winter 2016","Winter 2017","Winter 2018","W
                                "Spring 2015","Spring 2016","Spring 2017","Spring 2018","S
                    ))
map_sf$pred_ll <- as.numeric(map_sf$pred_ll)
map_sf$quantil <- NA
map_sf$quantil[map_sf$pred_ll > 0 & map_sf$pred_ll < 0.2] <- "0.01-0.19"
map_sf$quantil[map_sf$pred_ll >= 0.2 & map_sf$pred_ll < 1] <- "0.2-0.9"
map_sf$quantil[map_sf$pred_ll >= 1 & map_sf$pred_ll < 3] <- "1.0-2.9"
map_sf$quantil[map_sf$pred_ll >= 3 & map_sf$pred_ll < 20] <- "3.0-19.9"
map_sf$quantil[map_sf$pred_ll >= 20] <-">20"
map_sf$quantil <- factor(map_sf$quantil,
```

```r
                              levels =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20"))

ggp <- ggplot() +
  geom_sf(data = map_sf, aes(fill = quantil), color = "transparent") +
  geom_sf(data = reg_main, fill = "transparent", aes(color = "Brazil´s regions"), linewidt
  theme_classic() +
  scale_fill_manual(name = "", values= c("#0277BD",'#abdda4', '#ffffbf', '#fdae61', "red")
                    limits =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20")) +
  scale_color_manual( name = "",  values = c("Brazil´s regions" = "black") ) +
  guides(
    fill = guide_legend(order = 1,
                        label.theme = element_text(size = 8),
                        keywidth = unit(0.3, "cm"),
                        keyheight = unit(0.3, "cm"),
                        nrow = 1),
    color = guide_legend( order = 2,
                          override.aes = list(fill = NA, linetype = 1, size = 0.2),
                          label.theme = element_text(size = 8),
                          keywidth = unit(0.28, "cm"),
                          keyheight = unit(0.28, "cm"),
                          nrow = 1)) +
  theme(legend.text = element_text(size = 10, face = "bold"),
        legend.direction = "horizontal",
        legend.position = "top",
        legend.box = "horizontal",
        legend.margin = margin(0, 0, 0, 0),
        text = element_text(size = 10),
        axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        strip.text = element_text(size = 4),
        panel.spacing = unit(0.01, "lines"),
        panel.spacing.x = unit(0.01, "lines"),
        panel.spacing.y = unit(0.01, "lines")) +
  facet_wrap(~ id, ncol = 10 ) +
  labs(title = "",
       x = "",
       y = "") +
  blank() +
  scalebar(map_sf, dist = 1000, dist_unit = "km", st.size = 1.1,
           height=0.08, border.size = 0.1,
           st.dist = 0.1,
           anchor = c(x = -35, y = -38),
           transform = TRUE, model = "WGS84",
           facet.var = c("id"),
           facet.lev = c("Spring 2024"))

jpeg("FigS7_map_pred_ll.jpeg", width = 17, height = 10,
     units = "cm", res = 2000, type = "cairo"
)
ggp
north2(ggp = ggp, scale = 0.065, x = 0.88, y = 0.04, symbol = 10)
```

```
dev.off()


# ---- 17. Map: Upper  Limit of Predicted Cases (95% CI) ----------------------
dw_ul <-dataf %>% select (id_key_time, ID_MN_RESI, pred_ul) %>%
  reshape(timevar = "id_key_time",
          idvar = "ID_MN_RESI", direction = "wide")
map <- merge(MN_shp[,c("COD_MUN6", "name_muni", "abbrev_state", "name_region")], dw_ul, by
map_sf <- st_as_sf(map)

map_sf <- gather(map_sf, id_key_time, pred_ul, paste0("pred_ul.", 1:4, "_", rep(2015:2024,
map_sf$id <- gsub("pred_ul.", "", map_sf$id_key_time)
map_sf <- map_sf  %>% separate(id , c("season", "year"), sep="_")

map_sf$season [map_sf$season =="1"] <- "Summer"
map_sf$season [map_sf$season =="2"] <- "Autumn"
map_sf$season [map_sf$season =="3"] <- "Winter"
map_sf$season [map_sf$season =="4"] <- "Spring"
map_sf$season <- factor(map_sf$season , levels = c("Summer", "Autumn", "Winter","Spring" )

map_sf$id <- paste(map_sf$season, map_sf$year)
map_sf$id <- factor(map_sf$id,
                    levels =  c("Summer 2015","Summer 2016","Summer 2017","Summer 2018","S
                                "Autumn 2015","Autumn 2016","Autumn 2017","Autumn 2018","A
                                "Winter 2015","Winter 2016","Winter 2017","Winter 2018","W
                                "Spring 2015","Spring 2016","Spring 2017","Spring 2018","S
                    ))

map_sf$pred_ul <- as.numeric(map_sf$pred_ul)
map_sf$quantil <- NA

map_sf$quantil[map_sf$pred_ul > 0 & map_sf$pred_ul < 0.2] <- "0.01-0.19"
map_sf$quantil[map_sf$pred_ul >= 0.2 & map_sf$pred_ul < 1] <- "0.2-0.9"
map_sf$quantil[map_sf$pred_ul >= 1 & map_sf$pred_ul < 3] <- "1.0-2.9"
map_sf$quantil[map_sf$pred_ul >= 3 & map_sf$pred_ul < 20] <- "3.0-19.9"
map_sf$quantil[map_sf$pred_ul >= 20] <-">20"
map_sf$quantil <- factor(map_sf$quantil,
                         levels =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20"))

ggp <- ggplot() +
  geom_sf(data = map_sf, aes(fill = quantil), color = "transparent") +
  geom_sf(data = reg_main, fill = "transparent", aes(color = "Brazil´s regions"), linewidt
  theme_classic() +
  scale_fill_manual(name = "", values= c("#0277BD",'#abdda4', '#ffffbf', '#fdae61', "red")
                    limits =  c("0.01-0.19", "0.2-0.9","1.0-2.9", "3.0-19.9", ">20")) +
  scale_color_manual( name = "",  values = c("Brazil´s regions" = "black") ) +
  guides(
    fill = guide_legend(order = 1,
                        label.theme = element_text(size = 8),
                        keywidth = unit(0.3, "cm"),
                        keyheight = unit(0.3, "cm"),
                        nrow = 1),
```

```r
    color = guide_legend( order = 2,
                          override.aes = list(fill = NA, linetype = 1, size = 0.2),
                          label.theme = element_text(size = 8),
                          keywidth = unit(0.28, "cm"),
                          keyheight = unit(0.28, "cm"),
                          nrow = 1)) +
  theme(legend.text = element_text(size = 10, face = "bold"),
        legend.direction = "horizontal",
        legend.position = "top",
        legend.box = "horizontal",
        legend.margin = margin(0, 0, 0, 0),
        text = element_text(size = 10),
        axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        strip.text = element_text(size = 4),
        panel.spacing = unit(0.01, "lines"),
        panel.spacing.x = unit(0.01, "lines"),
        panel.spacing.y = unit(0.01, "lines")) +
  facet_wrap(~ id, ncol = 10 ) +
  labs(title = "",
       x = "",
       y = "") +
  blank() +
  scalebar(map_sf, dist = 1000, dist_unit = "km", st.size = 1.1,
           height=0.08, border.size = 0.1,
           st.dist = 0.1,
           anchor = c(x = -35, y = -38),
           transform = TRUE, model = "WGS84",
           facet.var = c("id"),
           facet.lev = c("Spring 2024"))


jpeg("FigS8_map_pred_ul.jpeg", width = 17, height = 10,
     units = "cm", res = 2000, type = "cairo"
)
ggp
north2(ggp = ggp, scale = 0.065, x = 0.88, y = 0.04, symbol = 10)
dev.off()


```