Brucellosis

2024-04-17

Contents

Test of Association and Time Series Linear Regression

The following section presents the data cleaning steps, the trend of brucellosis cases, the test for stationarity, and the spatial analysis. The section also includes the models run for test of association for individual cases, combined cases, (with and without NA).

```
# Importing packages
if (require(pacman))
{
  p_load(
    tidyverse,
    tseries,
    data.table,
    scales,
    zoo,
    forecast,
    sf,
    patchwork,
    grid,
    fable,
    patchwork,
    xts,
    feasts,
    cowplot,
    broom,
    kableExtra,
    readxl,
    stringi,
    stringr,
    rKenyaCensus,
    knitr,
    purrr,
    RColorBrewer
  )
}


# Importing Data
p09 <- read_excel("Projections.xlsx",
                  sheet = "2009")
p19 <- read_excel("Projections.xlsx",
                  sheet = "2019")
df_incidence <- fread("all_bruc_incidence.csv")
df_numbers <- fread("all_bruc_pop.csv")
shp <- st_read("shapefiles/County.shp", quiet = T)
eco_zones <- fread("eco_zones_county.csv")
shp <- st_read("shapefiles/County.shp", quiet = T)
```

```
# 2009
colnames(p09) <- p09[1,] #Make the 1st row the header
p09 <- p09[-1, ]  #Remove the 1st row

p09_clean <- p09 |>
  filter(
```

```
    !County %in% c(
      "Kenya",
      "Central",
      "Coast",
      "Eastern",
      "Western",
      "N. Eastern",
      "Nyanza",
      "R. Valley"
    )
  ) |>
  distinct() |>
  select(County, "2014", "2015")

# 2019
p19_clean <- p19 |>
  select(County, "2020", "2021", "2022", "2023", "2024", "2025") |>
  setNames(c("County", "pop2020", "pop2021", "pop2022", "pop2023", "pop2024", "pop2025")) |>
  na.omit() |>
  mutate(across(2:7, ~ as.numeric(str_remove(., ","))),
         across(2:7, ~ . * 1000))

p19_clean2 <- p19_clean |>
  filter(County != "Kenya") |>
  pivot_longer(
    cols = -County,
    names_to = "Year",
    values_to = "pop"
  ) |>
  arrange()


p19_clean2_kenya <- p19_clean |>
  filter(County == "Kenya") |>
  pivot_longer(
    cols = -County,
    names_to = "Year",
    values_to = "pop"
  ) |>
  arrange() |>
  select(County,Year, pop) |>
  mutate(Year = str_replace(Year, "pop", "") |> str_squish() |> as.numeric())

write.csv(p19_clean2_kenya, "kenya_pop_2020_2025.csv")
# Growth Rate
p19_growth <- p19_clean2 |>
  group_by(County) |>
  arrange(County, Year) |>
  mutate(growth_rate = (pop - lag(pop)) / lag(pop) * 100) |>
  group_by(County) |>
  summarise(growth_rate = mean(growth_rate, na.rm = T) |> round(2))
write.csv(p19_growth, "growthrate.csv")
```

```r
# 2019 Population
pop <- rKenyaCensus::V1_T2.2 |>
  select(County, pop2019 = Total) |>
  filter(!County %in% c("Total")) |>
  arrange(County) |>
  mutate(County = case_when(
    County == "Elgeyo/Marakwet" ~ "Elgeyo Marakwet",
    County == "Nairobi City" ~ "Nairobi",
    County == "Taita/Taveta" ~ "Taita Taveta",
    County == "Tharaka-Nithi" ~ "Tharaka Nithi",
    TRUE ~ County
  )) |>
  merge(p19_growth, by = "County") |>
  mutate(
    pop2018 = round(pop2019 * (1 - (growth_rate / 100))),
    pop2017 = round(pop2018 * (1 - (growth_rate / 100))),
    pop2016 = round(pop2017 * (1 - (growth_rate / 100))),
    pop2015 = round(pop2016 * (1 - (growth_rate / 100))),
    pop2014 = round(pop2015 * (1 - (growth_rate / 100)))
  ) |>
  merge(p19_clean |> select(1:4), by = "County") |>
  select(County,
         pop2014,
         pop2015,
         pop2016,
         pop2017,
         pop2018,
         pop2019,
         pop2020,
         pop2021
  ) |>
  pivot_longer(
    cols = -County,
    values_to = "pop"
  ) |>
  mutate(year = case_when(
    name == "pop2014" ~ 2014,
    name == "pop2015" ~ 2015,
    name == "pop2016" ~ 2016,
    name == "pop2017" ~ 2017,
    name == "pop2018" ~ 2018,
    name == "pop2019" ~ 2019,
    name == "pop2020" ~ 2020,
    name == "pop2021" ~ 2021
  )) |>
  select(county = County, year, pop)
write.csv(pop, "population_county_2014_2021.csv")
```

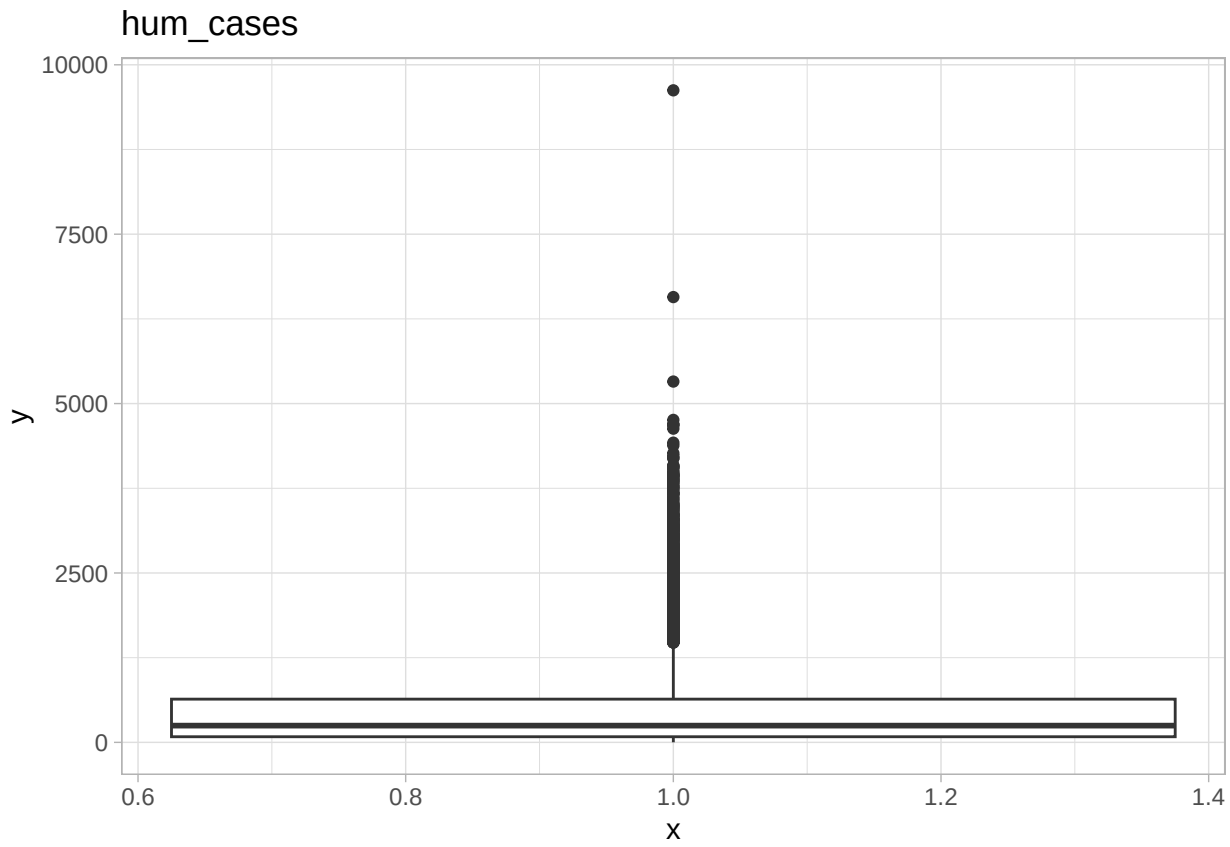
```r
# Cases per year per county
df_incidence2.1 <- df_incidence |>
  select(date,
         county,
         diseases,
```

```
        diagnosis,
        contains("cases"),
        catt_pop,
        goat_pop,
        sheep_pop,
        cam_pop) |>
  mutate(year = lubridate::year(date)) |>
  merge(pop, by = c("county", "year"), all = T) |>
  filter(!is.na(year))

# Identifying the outliers in the number of cases
numeric_columns <- df_incidence2.1 %>%
  select(contains('cases')) |>
  keep(is.numeric)

numeric_columns %>%
  map(~summary(.))
```

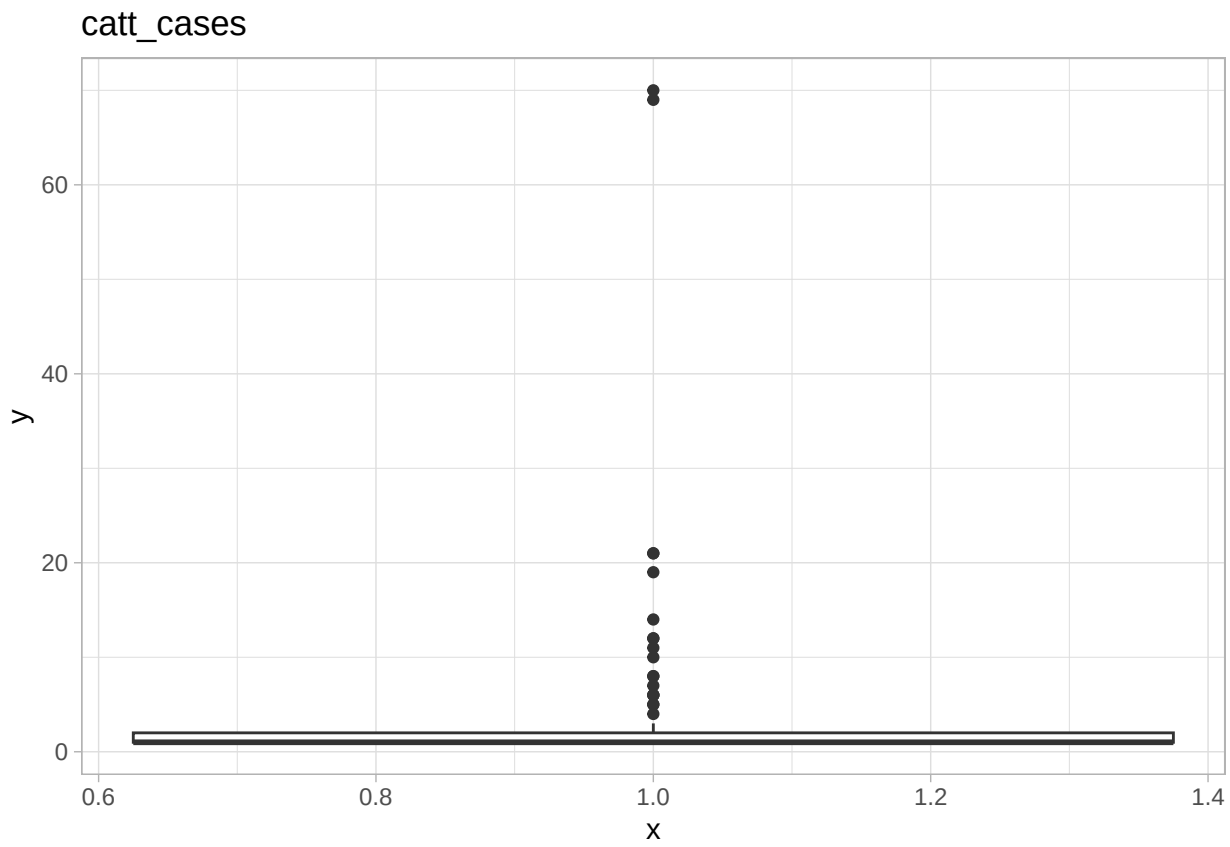
```
## $hum_cases
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     1.0    83.0   247.0   503.5   638.0  9623.0      10
##
## $catt_cases
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   1.000   1.000   4.175   2.000  70.000    8152
##
## $cam_cases
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##       1       1       1       4       4      13    8245
##
## $goat_cases
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1.00    1.00    2.00   22.56   10.00  528.00    8208
##
## $shp_cases
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     1.0     1.0     2.5     4.5     8.0    12.0    8237
```

```
numeric_columns %>%
  imap(
    ~ ggplot(data = data.frame(y = .x)) +
      geom_boxplot(aes(x = 1, y = y)) +
      labs(title = .y) +
      theme_light()
  )
```

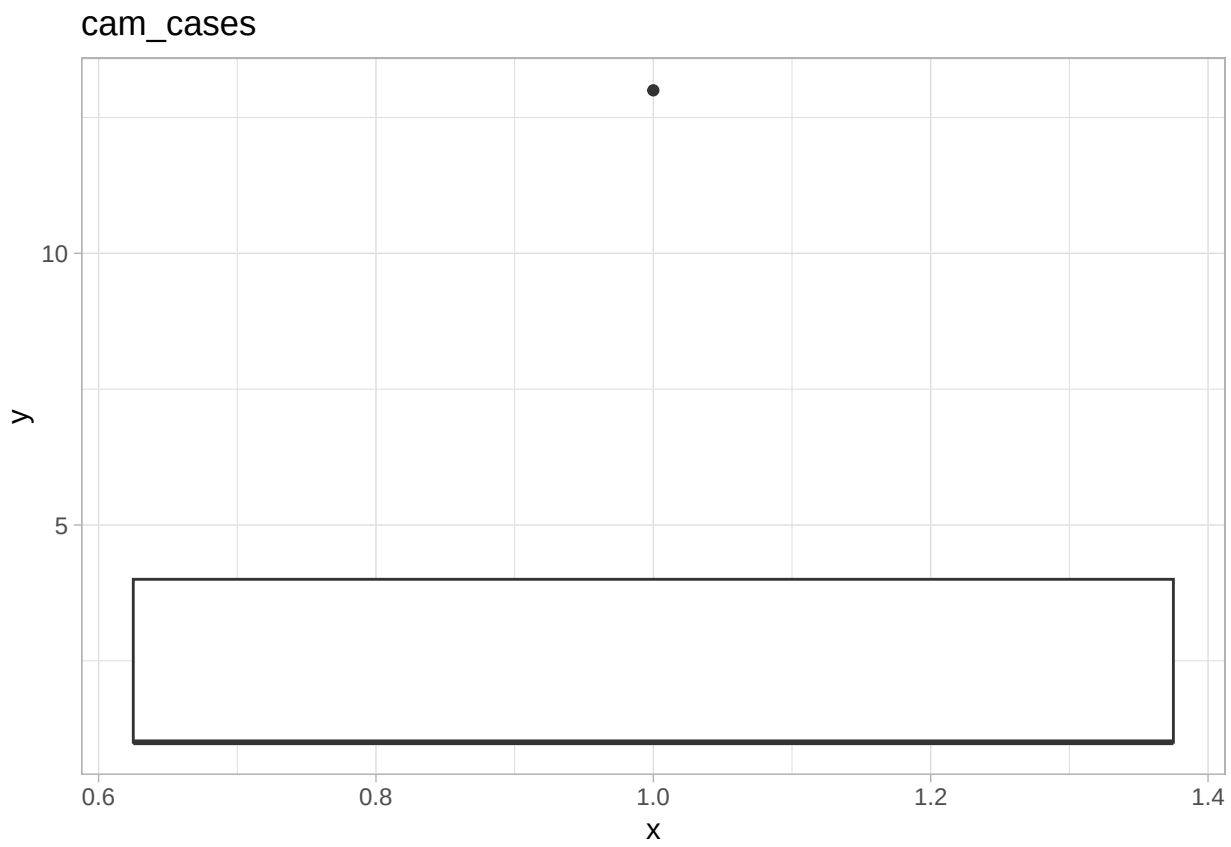
```
## $hum_cases
```



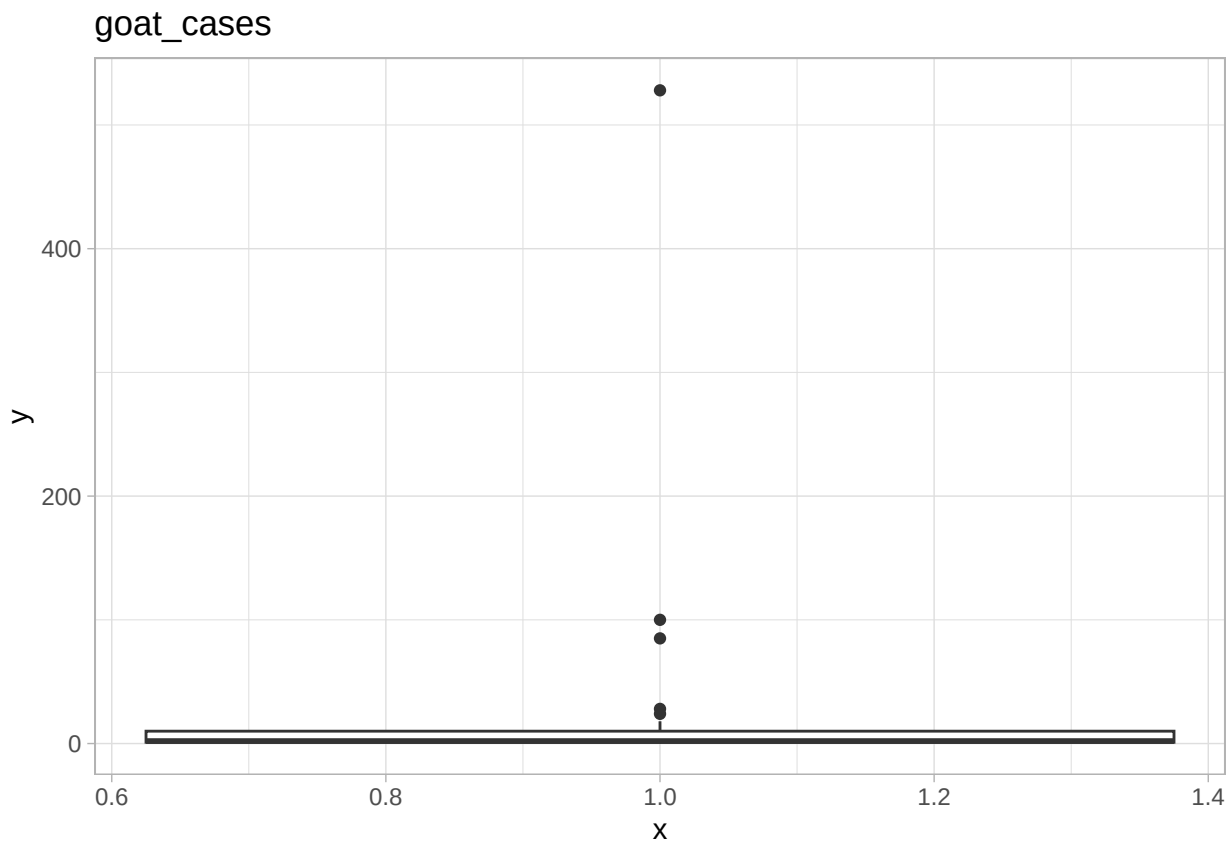

```
##
## $catt_cases
```



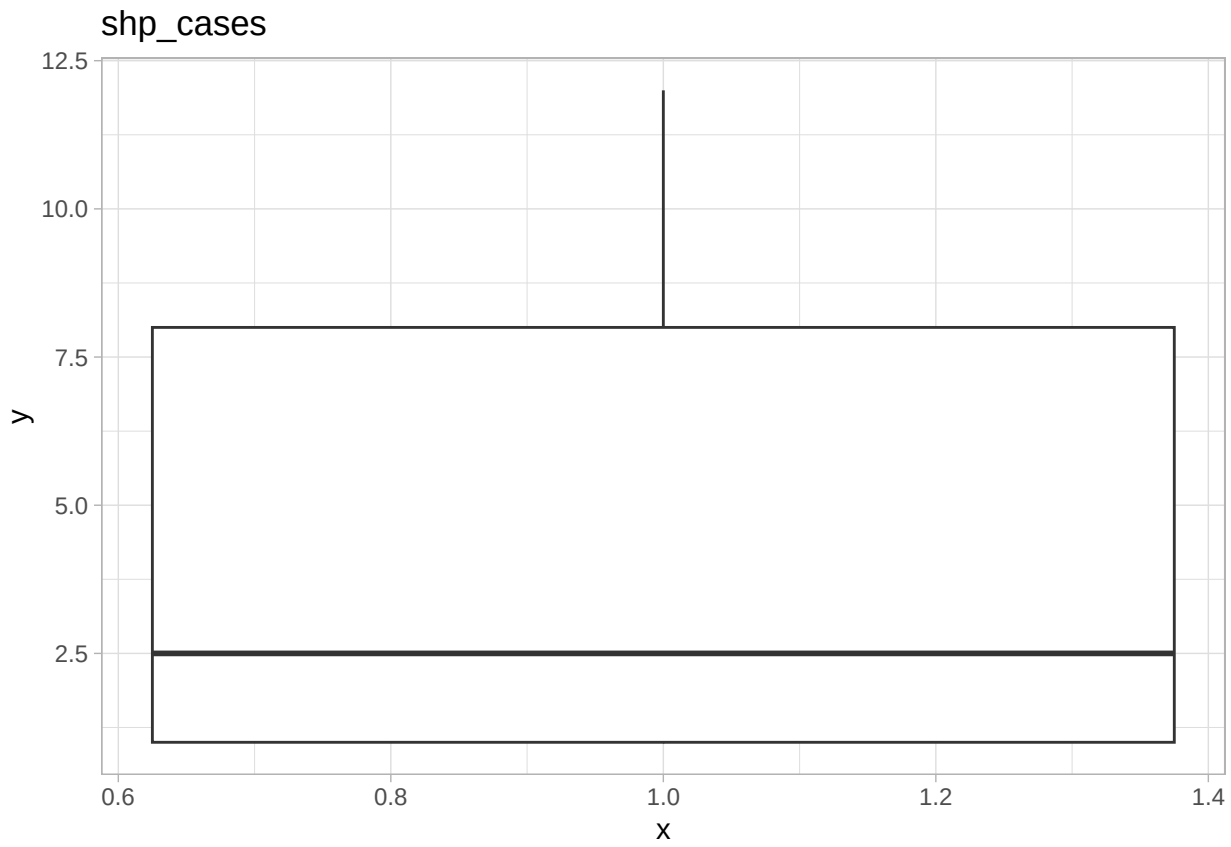

```
##
## $cam_cases
```




```
##
## $goat_cases
```




```
##
## $shp_cases
```



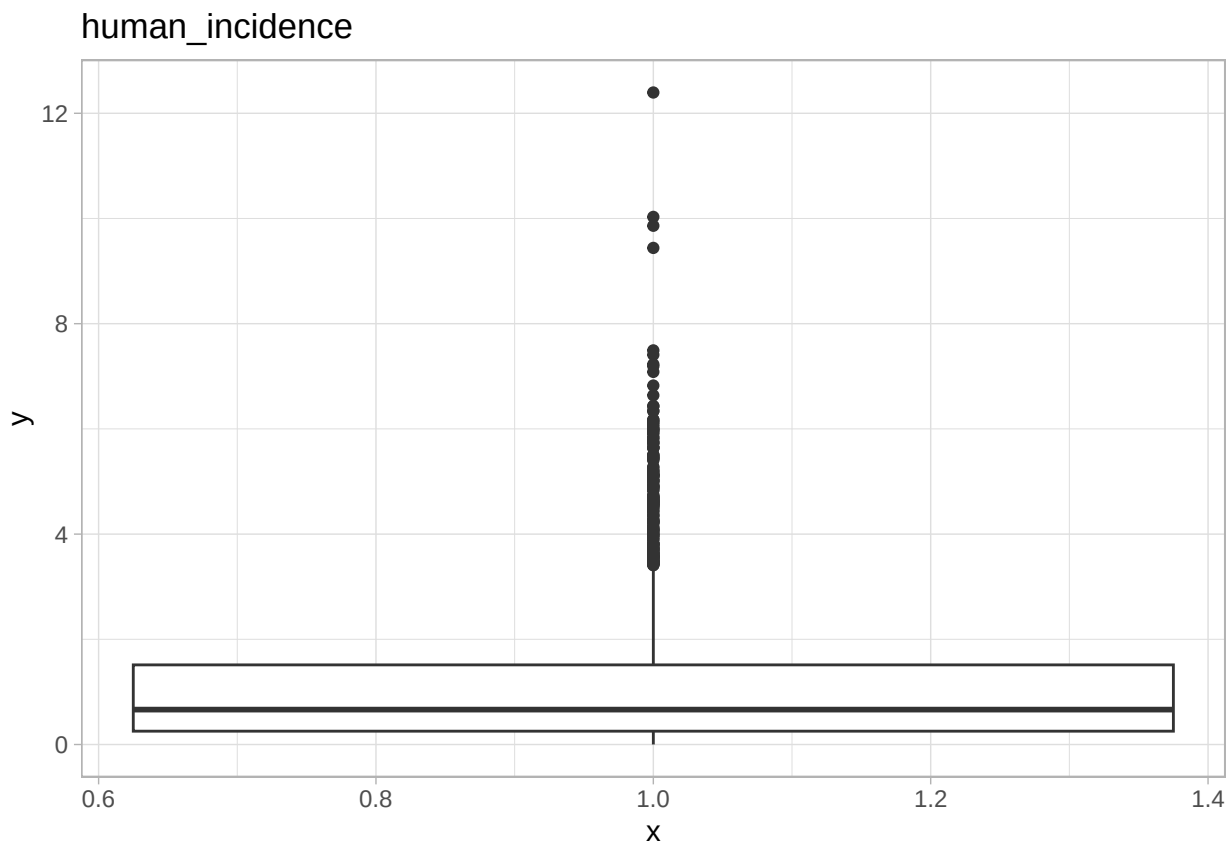

```
# Replacing the outliers with the mean
df_incidence2 <- df_incidence2.1 |>
  mutate(
    catt_cases = ifelse(catt_cases >= 69, round(mean(catt_cases, na.rm = T)), catt_cases),
    goat_cases = ifelse(goat_cases > 28, round(mean(goat_cases, na.rm = T)), goat_cases)
  ) |>
  mutate(date = as.Date(date))

df_tot_cases <- df_incidence2 |>
  group_by(date, county) |>
  summarise(across(contains("cases"), ~ sum(., na.rm = T)))

# Population per year, per county
df_pop <- df_incidence2 |>
  select(date, county, contains("pop")) %>%
  distinct(.) |>
  as_tibble() |>
  group_by(date, county) %>%
  summarise(across(where(is.numeric), ~unique(.)))

df_1 <- df_tot_cases |>
  merge(df_pop, by = c("date", "county")) |>
  filter(!is.na(date)) |>
  mutate(
    human_incidence = round((hum_cases / pop) * 1000, 4),
    catt_incidence = round((catt_cases / catt_pop) * 1000000, 4),
```

```
    cam_incidence = round((cam_cases / cam_pop) * 1000000, 4),
    goat_incidence = round((goat_cases / goat_pop) * 1000000, 4),
    shp_incidence = round((shp_cases / sheep_pop) * 1000000, 4)
  ) |>
  select(date, county, contains(c("incidence", "cases"))) |>
  as_tibble() |>
  arrange(county)

df_1_pop <- df_tot_cases |>
  merge(df_pop, by = c("date", "county")) |>
  select(date, county, pop)
write.csv(df_1_pop, 'county_humanpop.csv', row.names = F)

df_1_complete <- df_1 |>
  select(date, county, human_incidence, catt_incidence, goat_incidence, hum_cases, catt_cases, goat_cas
  filter(catt_incidence > 0)
write.csv(df_1_complete, "df_1_complete.csv", row.names = F)

# Outliers
numeric_columns <- df_1 %>%
  select(contains('incidence')) |>
  keep(is.numeric)

numeric_columns %>%
  map(~summary(.))
```

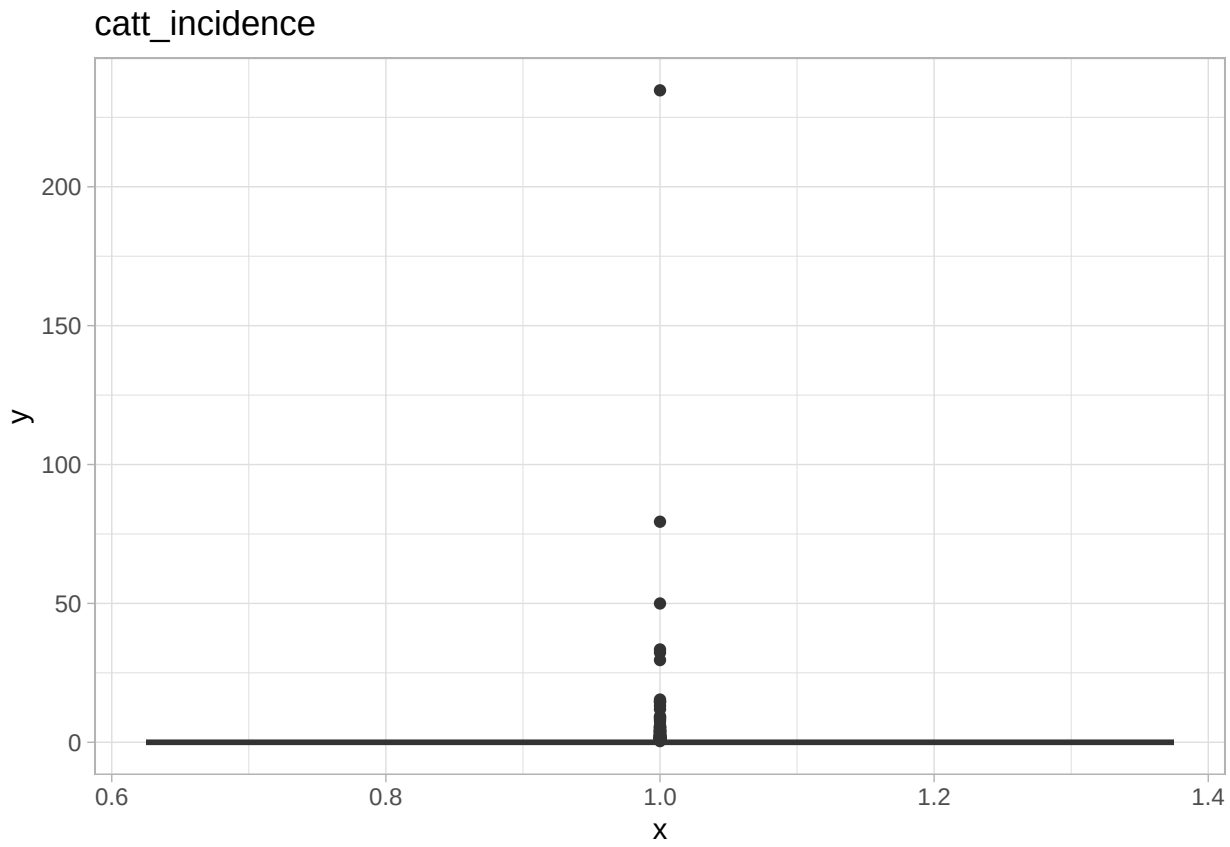
```
## $human_incidence
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0007  0.2535  0.6634  1.0457  1.5137 12.3950
##
## $catt_incidence
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##   0.0000   0.0000   0.0000   0.1736   0.0000 234.7197
##
## $cam_incidence
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
##   0.0000   0.0000   0.0000   0.0506   0.0000 121.9558     1911
##
## $goat_incidence
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.00000  0.00000  0.03474  0.00000 30.07240
##
## $shp_incidence
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00000  0.00000  0.00000  0.01545  0.00000 45.07380
```

```
numeric_columns %>%
  imap(
    ~ ggplot(data = data.frame(y = .x)) +
      geom_boxplot(aes(x = 1, y = y)) +
      labs(title = .y) +
      theme_light()
  )
```

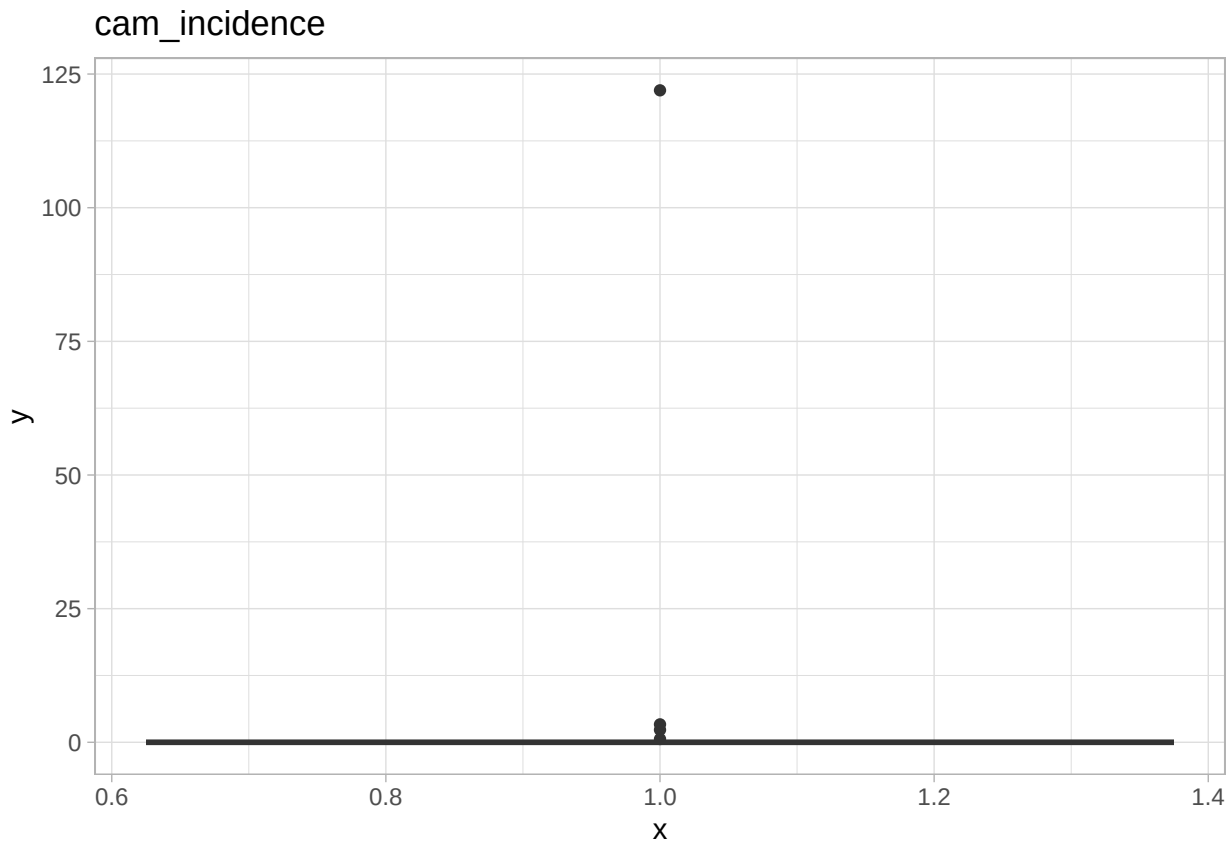
```
## $human_incidence
```



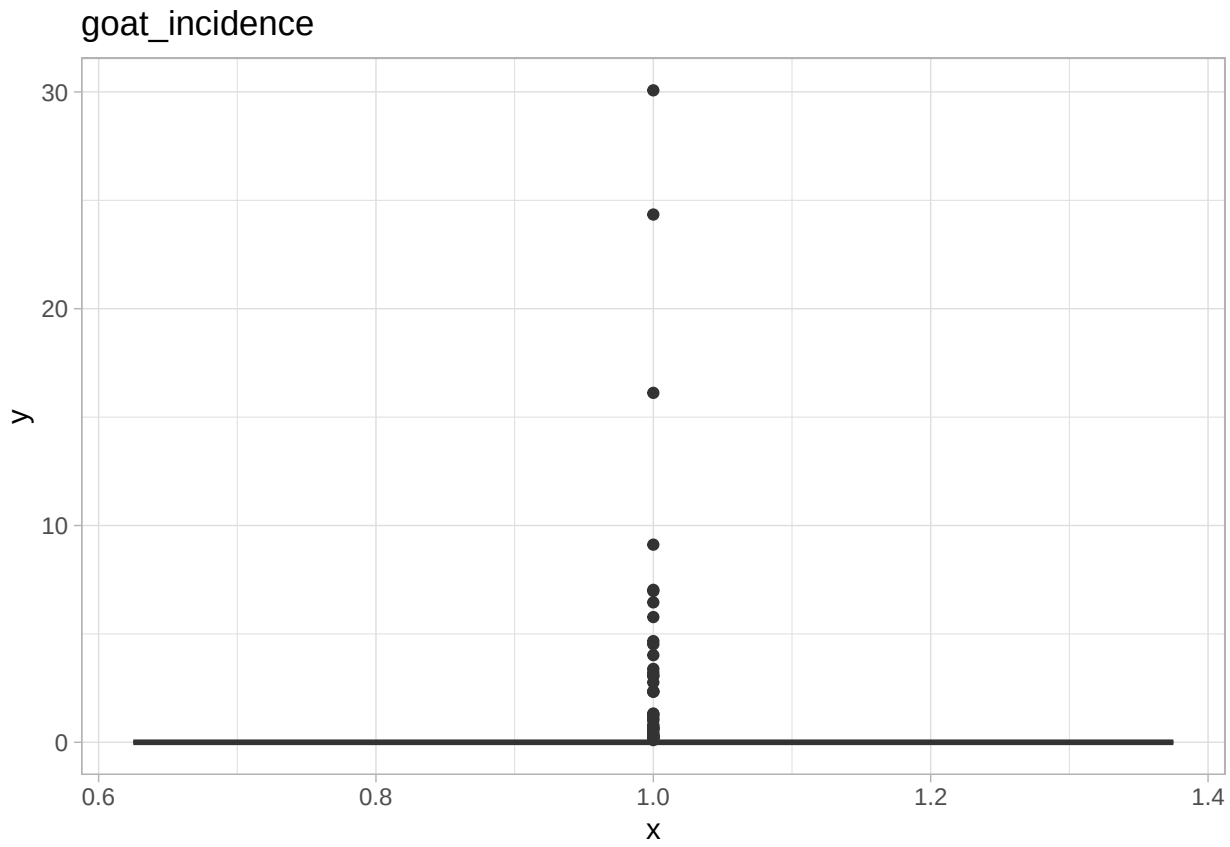

```
##
## $catt_incidence
```



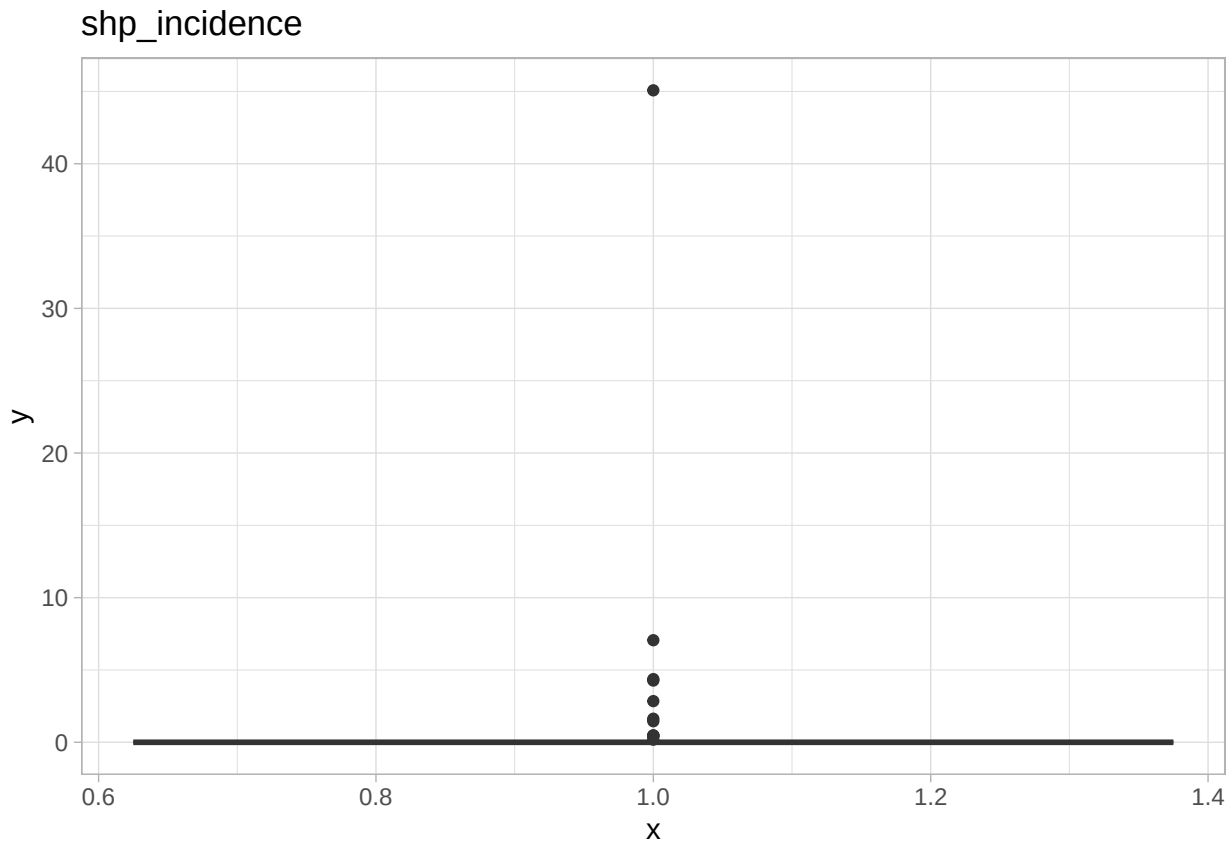

```
##
## $cam_incidence
```




```
##
## $goat_incidence
```




```
##
## $shp_incidence
```



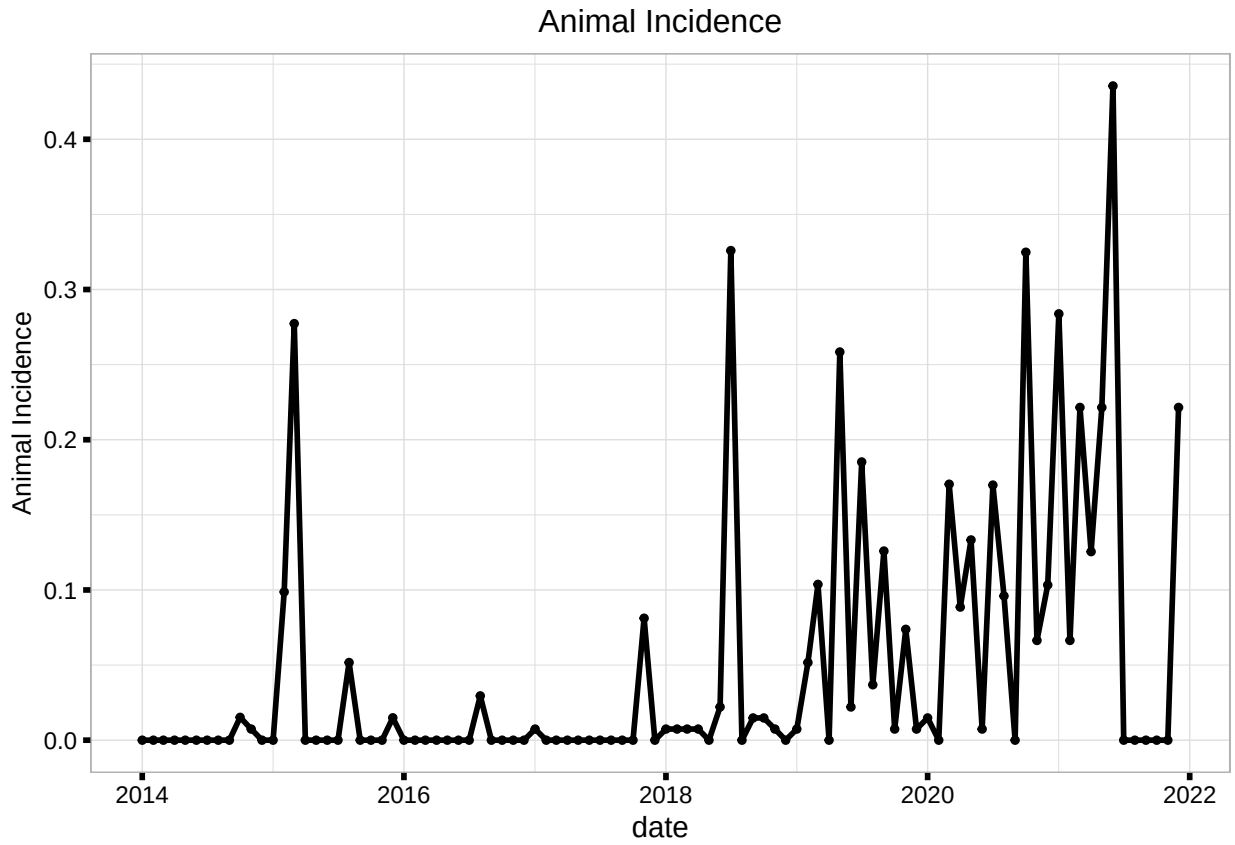

```
df_cum <- df_tot_cases |>
  merge(df_pop, by = c("date", "county")) |>
  filter(!is.na(date)) |>
  rowwise() |>
  mutate(
    animal_cases = sum(catt_cases, goat_cases, shp_cases, cam_cases, na.rm = T),
    animal_pop = sum(catt_pop, goat_pop, sheep_pop, cam_pop, na.rm = T),
    animal_cases = ifelse(animal_cases == 0, NA, animal_cases),
    animal_incidence = round((animal_cases / animal_pop) * 1000000, 4),
    human_incidence = round((hum_cases / pop) * 1000, 4)
  ) |>
  select(date, county, contains(c("incidence", 'cases'))) |>
  as_tibble() |>
  mutate()

df_cum_complete <- df_cum |>
  select(date, county, human_incidence, animal_incidence, animal_cases, hum_cases) |>
  filter(animal_incidence > 0)
write.csv(df_cum_complete, "df_cum_complete.csv", row.names = F)

## County
df_county <- df_incidence |>
  select(county, contains("cases")) |>
  mutate(
    catt_cases = ifelse(catt_cases >= 69, round(mean(catt_cases, na.rm = T)), catt_cases),
    goat_cases = ifelse(goat_cases > 28, round(mean(goat_cases, na.rm = T)), goat_cases)
```

```
  ) |>
  group_by(county) |>
  summarise(across(where(is.numeric), ~round(sum(., na.rm = T))))
```

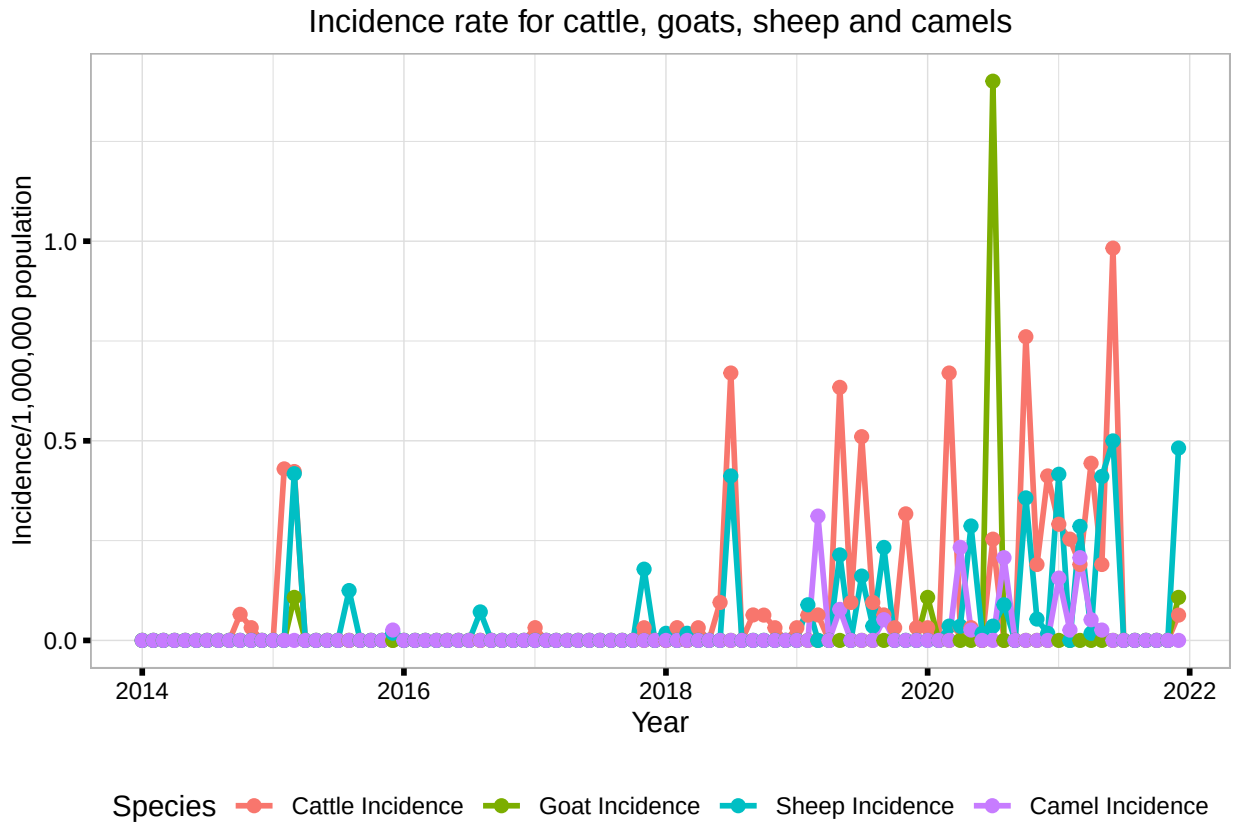
```
df_incidence2_trend <- df_incidence2 |>
  select(date,
         county,
         diseases,
         diagnosis,
         contains("cases"),
         catt_pop,
         goat_pop,
         sheep_pop,
         cam_pop) |>
  mutate(year = year(as.Date(date))) |>
  merge(pop, by = c("county", "year"))

df_tot_cases_trend <- df_incidence2_trend |>
  group_by(date) |>
  summarise(across(contains("cases"), ~ sum(., na.rm = T))) |>
  mutate(across(contains('cases'), ~ifelse(. == 0, NA, .)))

df_tot_pop_trend <- df_incidence2_trend |>
  select(date, county, contains("pop")) %>%
  distinct(.) |>
  as_tibble() |>
  group_by(date) %>%
  summarise(
    sheep_pop = sum(sheep_pop),
    goat_pop = sum(goat_pop),
    cam_pop = sum(cam_pop),
    catt_pop = sum(catt_pop),
    hum_pop = sum(pop)
  ) %>%
  as_tibble()


df_1_trend <- df_tot_cases_trend |>
  merge(df_tot_pop_trend, by = "date") |>
  filter(!is.na(date)) |>
  mutate(
    human_incidence = round((hum_cases / hum_pop) * 1000, 4),
    catt_incidence = round((catt_cases / catt_pop) * 1000000, 4),
    cam_incidence = round((cam_cases / cam_pop) * 1000000, 4),
    goat_incidence = round((goat_cases / goat_pop) * 1000000, 4),
    shp_incidence = round((shp_cases / sheep_pop) * 1000000, 4),
    date = as.Date(date)
  ) |>
  select(date, contains("incidence")) |>
  as_tibble() |>
  mutate(

  )
```

```r
write.csv(df_1_trend, "individual_incidence.csv")

df_1_trend_complete <- df_1_trend |>
  select(date, human_incidence, catt_incidence, goat_incidence) |>
  filter(!is.na(catt_incidence)) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))

# Cases
df_1_trend_cases <- df_tot_cases_trend |>
  merge(df_tot_pop_trend, by = "date") |>
  filter(!is.na(date)) |>
  mutate(
    human_incidence = round((hum_cases / hum_pop) * 1000, 4),
    catt_incidence = round((catt_cases / catt_pop) * 1000000, 4),
    cam_incidence = round((cam_cases / cam_pop) * 1000000, 4),
    goat_incidence = round((goat_cases / goat_pop) * 1000000, 4),
    shp_incidence = round((shp_cases / sheep_pop) * 1000000, 4),
    date = as.Date(date)
  ) |>
  select(date, contains(c("incidence", "cases"))) |>
  as_tibble() |>
  mutate(

  )
write.csv(df_1_trend_cases, "individual_incidence_cases.csv")


## The differenced one
date <- df_1_trend$date[-1]

df_1_trend_diff <- df_1_trend |>
  reframe(across(contains("incidence"), ~ diff(., na.rm = T))) |>
  mutate(date = as.Date(date))


df_cum_trend <- df_tot_cases_trend |>
  merge(df_tot_pop_trend, by = "date") |>
  filter(!is.na(date)) |>
  rowwise() |>
  mutate(animal_cases = sum(catt_cases, goat_cases, shp_cases, cam_cases, na.rm = T),
         animal_pop = sum(catt_pop, goat_pop, sheep_pop, cam_pop, na.rm = T),
        # animal_cases = ifelse(animal_cases == 0, NA, animal_cases),
         human_incidence = round((hum_cases / hum_pop) * 1000, 4),
         animal_incidence = round((animal_cases / animal_pop) * 1000000, 4)
  ) |>
  select(date, contains("incidence"))
write.csv(df_cum_trend, "combined_incidence.csv")

# Cases
df_cum_trend_cases <- df_tot_cases_trend |>
  merge(df_tot_pop_trend, by = "date") |>
  filter(!is.na(date)) |>
  rowwise() |>
```

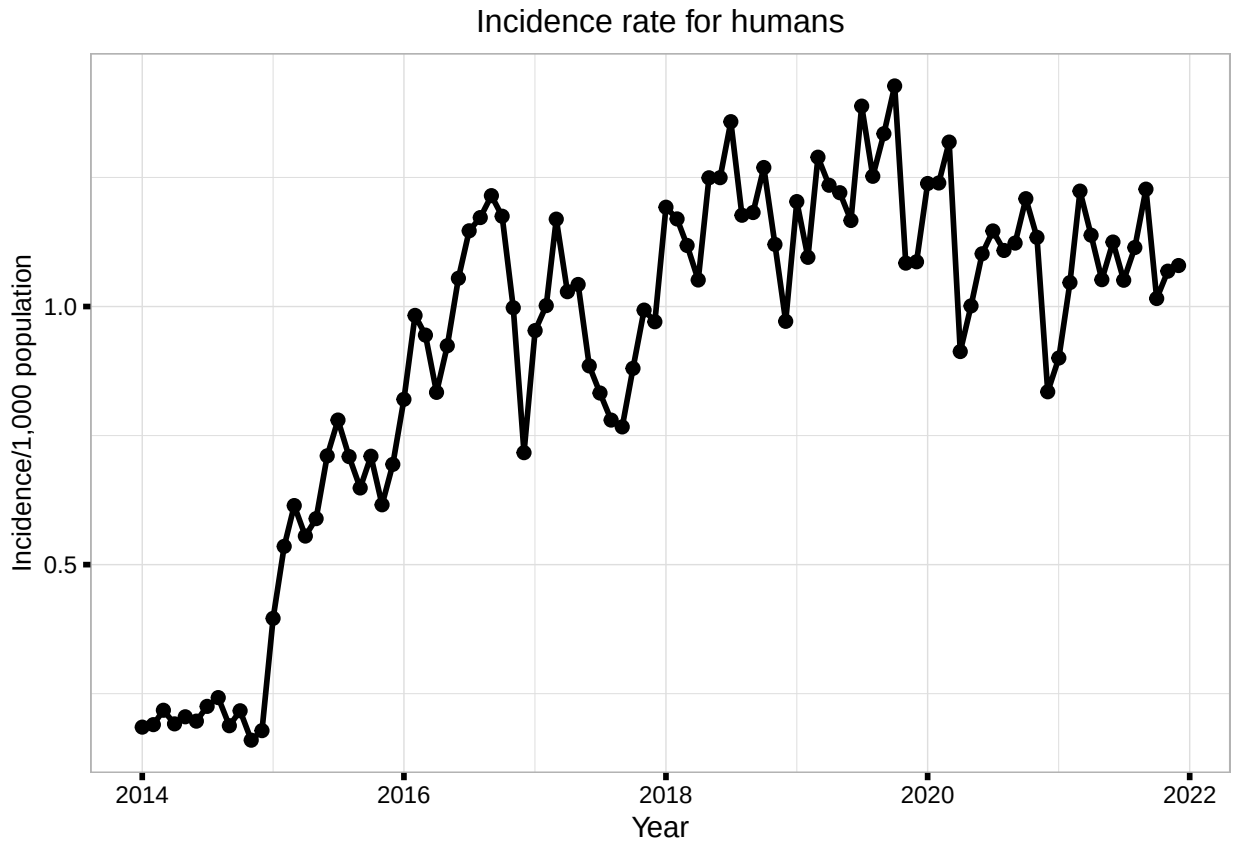
```r
  mutate(animal_cases = sum(catt_cases, goat_cases, shp_cases, cam_cases, na.rm = T),
         animal_pop = sum(catt_pop, goat_pop, sheep_pop, cam_pop, na.rm = T),
        # animal_cases = ifelse(animal_cases == 0, NA, animal_cases),
         human_incidence = round((hum_cases / hum_pop) * 1000, 4),
         animal_incidence = round((animal_cases / animal_pop) * 1000000, 4)
  ) |>
  select(date, contains(c("incidence", "cases")))
write.csv(df_cum_trend_cases, "combined_incidence_cases.csv")

# The differenced one
df_cum_trend_diff <- df_cum_trend |>
  arrange(date) |>
  as.data.frame() |>
  reframe(across(c(human_incidence, animal_incidence), ~ diff(., 1, na.rm = T))) |>
  mutate(date = as.Date(date))

trend_data <- df_1_trend %>%
  pivot_longer(cols = -date) %>%
  mutate(
    name = factor(name, levels = unique(name)),
    name = factor(name, labels = c(
      "Human Incidence", "Cattle Incidence", "Goat Incidence",
      "Sheep Incidence", "Camel Incidence"
    ))
  )

all_plus_hum <- df_cum_trend |>
  #filter(!is.na(animal_incidence)) |>
  mutate(animal_incidence = ifelse(is.na(animal_incidence), 0, animal_incidence)) |>
  ggplot(aes(date)) +
  geom_point(aes(y = animal_incidence), col = "black", size = 1) +
  geom_line(aes(y = animal_incidence), col = "black", linewidth = 1) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Animal Incidence") +
  ggtitle('Animal Incidence')
all_plus_hum
```




```
# All except humans
species_plt <- trend_data %>%
  filter(name != "Human Incidence") |>
  mutate(value = ifelse(is.na(value), 0, value)) |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = value, col = name), linewidth = 1) +
  geom_point(aes(y = value, col = name), size = 2) +
  theme_light() +
  #facet_wrap(~name, scales = "free", ncol = 3) +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Incidence rate for cattle, goats, sheep and camels")

species_plt
```




```
# Humans
humans_plt <- trend_data %>%
  filter(name == "Human Incidence") |>
  mutate(value = ifelse(is.na(value), 0, value)) |>

  ggplot(aes(x = date)) +
  geom_line(aes(y = value), linewidth = 1) +
  geom_point(aes(y = value), size = 2) +
  theme_light() +
  #facet_wrap(~name, scales = "free", ncol = 3) +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Incidence rate for humans")
humans_plt
```




```
# At county Level
adf1 <- adf.test(na.omit(df_1$human_incidence))
adf2 <- adf.test(na.omit(df_1$catt_incidence))
adf3 <- adf.test(na.omit(df_1$cam_incidence))
adf4 <- adf.test(na.omit(df_1$goat_incidence))
adf5 <- adf.test(na.omit(df_1$shp_incidence))

adf_res <- data.frame(
  variable = c(
    "Human Incidence",
    "Cattle Incidence",
    "Camel Incidence",
    "Goat Incidence",
    "Sheep Incidence"
  ),
  statistic = c(
    adf1$statistic,
    adf2$statistic,
    adf3$statistic,
    adf4$statistic,
    adf5$statistic
  ),

  `P Value` = c(
    adf1$p.value,
    adf2$p.value,
```

```
    adf3$p.value,
    adf4$p.value,
    adf5$p.value
  )


) %>%
  mutate(across(where(is.numeric), ~round(., 3))) |>
knitr::kable(
    align = "l",
    caption = "Results of Augmented Dickey-Fuller Test for Stationarity at County Level",
    format = "pipe",
    latex_options = "hold_position"
  )

# At National Level
adf_trend1 <- adf.test(na.omit(df_1_trend$human_incidence))
adf_trend2 <- adf.test(na.omit(df_1_trend$catt_incidence))
adf_trend3 <- adf.test(na.omit(df_1_trend$cam_incidence))
adf_trend4 <- adf.test(na.omit(df_1_trend$goat_incidence))
adf_trend5 <- adf.test(na.omit(df_1_trend$shp_incidence))

adf_trend_res <- data.frame(
  Variable = c(
    "Human Incidence",
    "Cattle Incidence",
    "Camel Incidence",
    "Goat Incidence",
    "Sheep Incidence"
  ),
  Statistic = c(
    adf_trend1$statistic,
    adf_trend2$statistic,
    adf_trend3$statistic,
    adf_trend4$statistic,
    adf_trend5$statistic
  ),

  `P Value` = c(
    adf_trend1$p.value,
    adf_trend2$p.value,
    adf_trend3$p.value,
    adf_trend4$p.value,
    adf_trend5$p.value
  )


) |>
  mutate(across(where(is.numeric), ~round(., 3))) |>
  knitr::kable(
    align = "l",
    caption = "Results of Augmented Dickey-Fuller Test for Stationarity at National Level",
    format = "pipe",
```

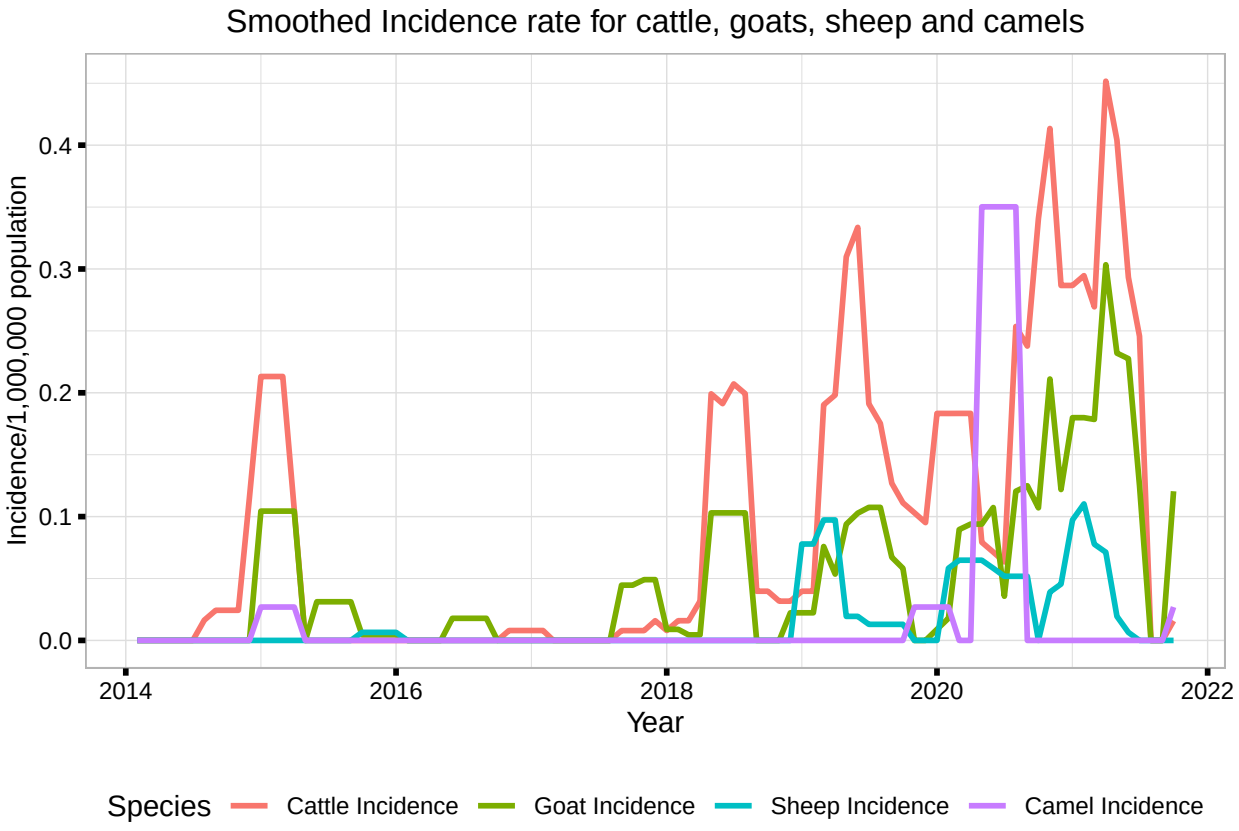
```
    latex_options = "hold_position"
  )

## All incidences combined

adf_combined <- adf.test(na.omit(df_cum$animal_incidence))
adf_combined_trend <- adf.test(df_cum_trend$animal_incidence)


adf_combined_results <- data.frame(
  "Level" = c("County", "National"),
  "Statistic" = c(adf_combined$statistic, adf_combined_trend$statistic),
  "P Value" = c(adf_combined$p.value, adf_combined_trend$p.value)
)
```

```
# Select columns containing "incidence"
incidence_cols <- grep("incidence", names(df_1_trend), value = TRUE)

# Apply moving average smoothing to selected columns
smoothed_df <- df_1_trend %>%
  mutate(across(incidence_cols, ~ifelse(is.na(.), 0, .))) |>
  mutate(across(all_of(incidence_cols), ~zoo::rollmean(., k = 4, fill = NA), .names = "smoothed_{.col}")
  na.omit()

# Print the first few rows of the smoothed data
head(smoothed_df)
```

```
## # A tibble: 6 x 11
##   date       human_incidence catt_incidence cam_incidence goat_incidence
##   <date>               <dbl>          <dbl>         <dbl>          <dbl>
## 1 2014-02-01           0.19               0             0              0
## 2 2014-03-01           0.218              0             0              0
## 3 2014-04-01           0.192              0             0              0
## 4 2014-05-01           0.205              0             0              0
## 5 2014-06-01           0.197              0             0              0
## 6 2014-07-01           0.225              0             0              0
## # i 6 more variables: shp_incidence <dbl>, smoothed_human_incidence <dbl>,
## #   smoothed_catt_incidence <dbl>, smoothed_cam_incidence <dbl>,
## #   smoothed_goat_incidence <dbl>, smoothed_shp_incidence <dbl>
```

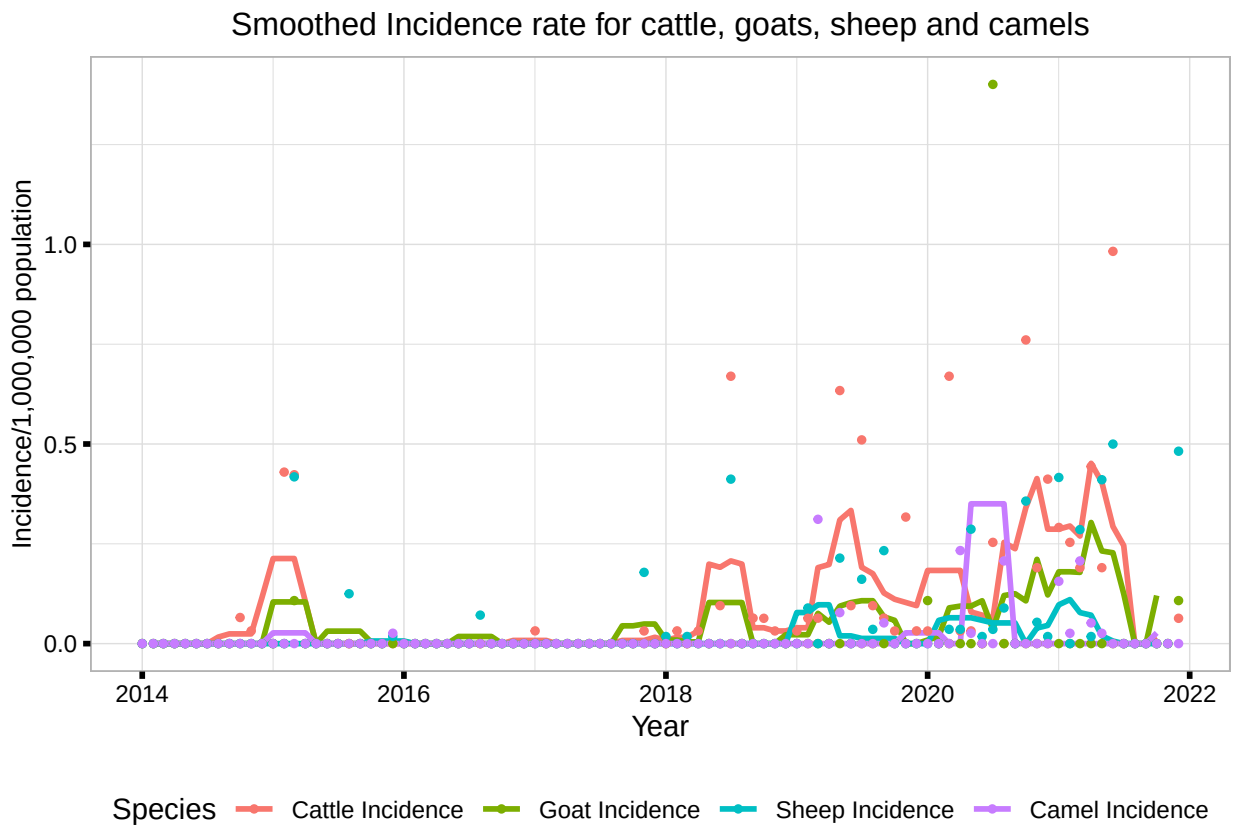
```
order <- c(
  "smoothed_human_incidence",
  "smoothed_catt_incidence",
  "smoothed_goat_incidence",
  "smoothed_shp_incidence",
  "smoothed_cam_incidence"
)


order2 <- c("Human Incidence", "Cattle Incidence", "Camel Incidence", "Goat Incidence",
"Sheep Incidence")

trend_data_smoothed <- smoothed_df %>%
```

```r
  select(date, contains("smoothed")) %>%
  pivot_longer(cols = -date) %>%
  mutate(
    name = factor(name, levels = order),
    name = factor(name, labels = c(
      "Human Incidence", "Cattle Incidence", "Goat Incidence",
      "Sheep Incidence", "Camel Incidence"
    ))
  )

df_long <- df_1_trend %>%
  mutate(across(incidence_cols, ~ifelse(is.na(.), 0, .))) |>
  select(date, contains("incidence")) %>%
  pivot_longer(cols = -date) %>%
  mutate(
    name = case_when(
    name == "human_incidence" ~ "Human Incidence",
    name == "catt_incidence" ~ "Cattle Incidence",
    name == "goat_incidence" ~ "Goat Incidence",
    name == "shp_incidence" ~ "Sheep Incidence",
    TRUE ~ "Camel Incidence"
  )) |>
  mutate(
    name = factor(name, levels = order2),
    name = factor(name, labels = c(
      "Human Incidence", "Cattle Incidence", "Goat Incidence",
      "Sheep Incidence", "Camel Incidence"
    ))
  )

species_sm_plt <- trend_data_smoothed |>
  filter(name != "Human Incidence") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = value, col = name), linewidth = 1) +
  #geom_point(aes(y = value, col = name)) +
  # facet_wrap(~name, scales = "free", ncol = 3) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Smoothed Incidence rate for cattle, goats, sheep and camels")

species_sm_plt
```

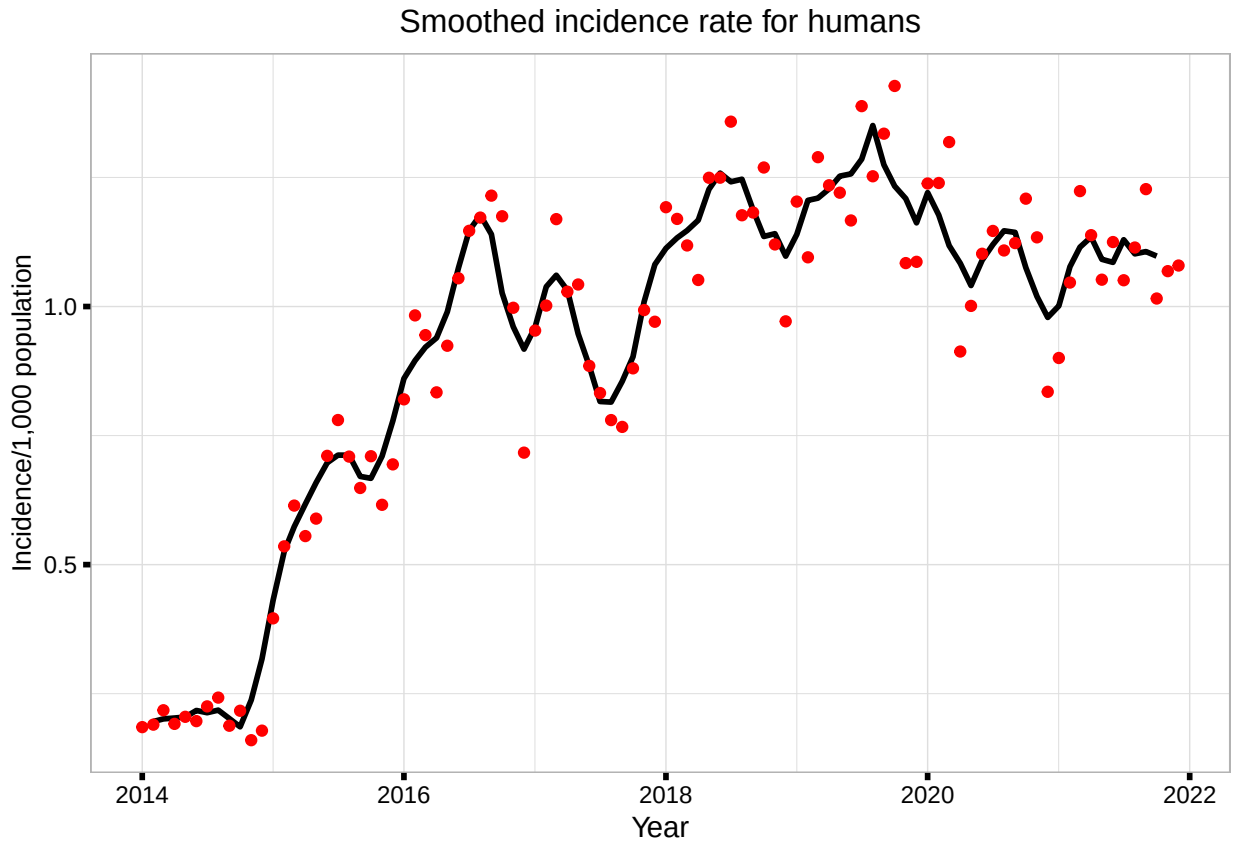



```
# With points
species_sm_plt_points <- trend_data_smoothed |>
  filter(name != "Human Incidence") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = value, col = name), linewidth = 1) +
  geom_point(data = trend_data |>
               mutate(value = ifelse(is.na(value), 0, value)) |>
               filter(name != "Human Incidence"),

             aes(y = value, col = name), size = 1) +
  # facet_wrap(~name, scales = "free", ncol = 3) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Smoothed Incidence rate for cattle, goats, sheep and camels")
```

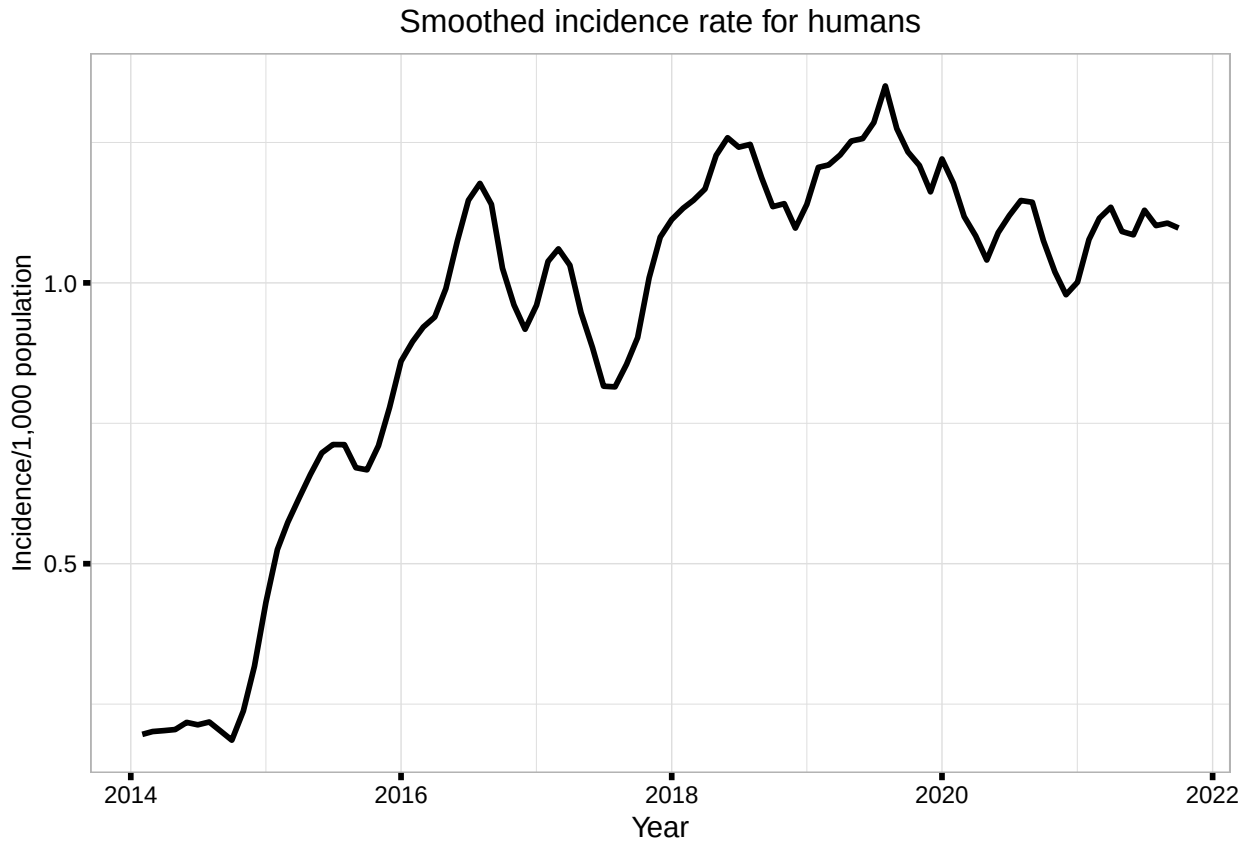
```
species_sm_plt_points
```




```
humans_sm_plt <- trend_data_smoothed |>
  filter(name == "Human Incidence") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = value), linewidth = 1) +
  geom_point(data = df_1_trend, aes(y = human_incidence), col = "red") +
  # facet_wrap(~name, scales = "free", ncol = 3) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Smoothed incidence rate for humans")
humans_sm_plt
```

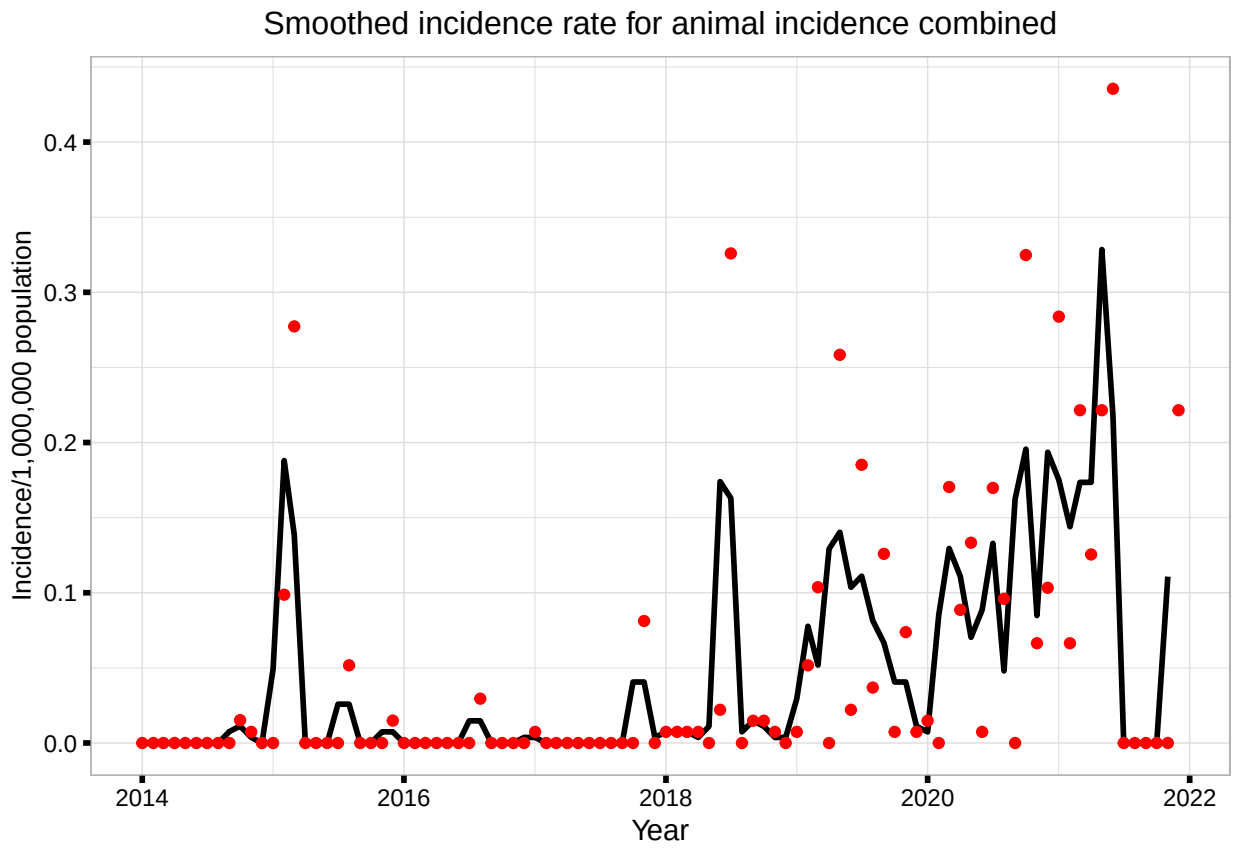



```
humans_sm_plt_nopoints <- trend_data_smoothed |>
  filter(name == "Human Incidence") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = value), linewidth = 1) +
  #geom_point(data = df_1_trend, aes(y = human_incidence), col = "red") +
  # facet_wrap(~name, scales = "free", ncol = 3) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Smoothed incidence rate for humans")
humans_sm_plt_nopoints
```




```
## For all animal incidence combined
df_cum_trend <- df_cum_trend |>
  mutate(date = as.Date(date))

incidence_cols_cum <- grep("incidence", names(df_cum_trend), value = TRUE)

# Apply moving average smoothing to selected columns
# Select columns containing "incidence"
incidence_cols <- grep("incidence", names(df_cum_trend), value = TRUE)

# Apply moving average smoothing to selected columns
smoothed_df_combined <- df_cum_trend %>%
  mutate(animal_incidence = ifelse(is.na(animal_incidence), 0, animal_incidence)) |>
  as_tibble() |>
  mutate(across(
    all_of(incidence_cols),
    ~ zoo::rollmean(., k = 2, fill = NA),
    .names = "smoothed_{.col}"
  )) |>
  na.omit()

animal_sm_plt <- smoothed_df_combined |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = smoothed_animal_incidence), linewidth = 1) +
  geom_point(data = df_cum_trend |>
               mutate(animal_incidence = ifelse(is.na(animal_incidence), 0, animal_incidence))
```

```
              , aes(y = animal_incidence), col = "red") +
  # facet_wrap(~name, scales = "free", ncol = 3) +
  theme_light() +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(color = "black", hjust = 0.5, size = 12),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "bottom",
    legend.text = element_text(color = "black")
  ) +
  ylab("Incidence/1,000,000 population") +
  xlab("Year") +
  labs(col = "Species", title = "Smoothed incidence rate for animal incidence combined")
animal_sm_plt
```




```
table1.1 <- df_incidence |>
  select(county, diagnosis, contains("cases")) |>
  mutate(
    catt_cases = ifelse(catt_cases >= 69, round(mean(catt_cases, na.rm = T)), catt_cases),
    goat_cases = ifelse(goat_cases > 28, round(mean(goat_cases, na.rm = T)), goat_cases)
  ) |>
```

```
  pivot_longer(cols = -c(county, diagnosis)) %>%
  group_by(name) |>
  group_by(name, Diagnosis = diagnosis) |>
  summarise(Cases = sum(value, na.rm = T)) |>
  mutate(
    Species = recode(
      name,
      "cam_cases" = 'Camels',
      "hum_cases" = 'Humans',
      "goat_cases" = 'Goats',
      "shp_cases" = "Sheep",
      "catt_cases" = "Cattle"
    )
  ) |>
  ungroup() |>
  select(-name) |>
  group_by(Species, Diagnosis) |>
  group_by(Species) |>
  mutate(`Percent(%)` = round((Cases/sum(Cases)) * 100, 2)) |>
  select(3, 1,2,4) |>
  knitr::kable(align = "l",
               caption = "Number of cases according to the type of Diagnosis",
               format = "pipe",
               latex_options = "hold_position")
table1.1
```

Table 1: Number of cases according to the type of Diagnosis

Species	Diagnosis	Cases	Percent(%)
Camels	Clinically confirmed	21.0	91.30
Camels	Lab confirmed	2.0	8.70
Camels	Post Mortem	0.0	0.00
Cattle	Clinically confirmed	162.0	57.65
Cattle	Lab confirmed	105.0	37.37
Cattle	Post Mortem	14.0	4.98
Goats	Clinically confirmed	191.0	62.42
Goats	Lab confirmed	115.0	37.58
Goats	Post Mortem	0.0	0.00
Humans	Clinically confirmed	945046.3	22.78
Humans	Lab confirmed	3203162.0	77.22
Humans	Post Mortem	0.0	0.00
Sheep	Clinically confirmed	65.0	79.27
Sheep	Lab confirmed	17.0	20.73
Sheep	Post Mortem	0.0	0.00

```
write.csv(table1.1 <- df_incidence |>
  select(county, diagnosis, contains("cases")) |>
  pivot_longer(cols = -c(county, diagnosis)) %>%
  group_by(name) |>
  group_by(name, Diagnosis = diagnosis) |>
  summarise(Cases = sum(value, na.rm = T)) |>
```

```
  mutate(
    Species = recode(
      name,
      "cam_cases" = 'Camels',
      "hum_cases" = 'Humans',
      "goat_cases" = 'Goats',
      "shp_cases" = "Sheep",
      "catt_cases" = "Cattle"
    )
  ) |>
  ungroup() |>
  select(-name) |>
  group_by(Species, Diagnosis) |>
  group_by(Species) |>
  mutate(`Percent(%)` = round((Cases/sum(Cases)) * 100, 2)) |>
  select(3, 1,2,4),

  "cases_table.csv")

# The descriptive statistics are for the Incidence Rate National Wide
table2 <- df_1 %>%
  select(county, contains("incidence")) |>
  pivot_longer(cols = -1) %>%
  group_by(name) %>%
  summarise(
    `Mean Incidence Rate` = mean(value, na.rm = TRUE),
    minimum = min(value, na.rm = TRUE),
    median = median(value, na.rm = TRUE),
    max = max(value, na.rm = TRUE),
    sd = sd(value, na.rm = TRUE)
  ) %>%
  arrange(desc(`Mean Incidence Rate`)) %>%
  mutate(
    name = case_when(
     name ==  "human_incidence" ~   "Human",
      name =="catt_incidence"  ~  "Cattle",
      name == "goat_incidence" ~  "Goat",
      name == "cam_incidence"  ~ "Camel",
      name == "shp_incidence" ~  "Sheep"
     ),
    Cases = comma(`Mean Incidence Rate`),
    Minimum = comma(minimum),
    Median = comma(median),
    Maximum = comma(max),
    `Standard Deviation` = comma(sd)
  ) |>
  select(Species = name,
         `Mean Incidence Rate`,
         Minimum,
         Median,
         Maximum,
         `Standard Deviation`) |>
  knitr::kable(
```

```
    align = "l",
    caption = "Descriptive Statistics for Incidence Rate",
    format = "pipe",
    latex_options = "hold_position"
  )

table2
```

Table 2: Descriptive Statistics for Incidence Rate

Species	Mean Incidence Rate	Minimum	Median	Maximum	Standard Deviation
Human	1.0457179	0.0007	0.66	12	1.1243
Cattle	0.1736142	0.0000	0.00	235	3.9378
Camel	0.0505520	0.0000	0.00	122	2.4226
Goat	0.0347421	0.0000	0.00	30	0.6939
Sheep	0.0154518	0.0000	0.00	45	0.6923

```
write.csv(df_1 %>%
  select(county, contains("incidence")) |>
  pivot_longer(cols = -1) %>%
  group_by(name) %>%
  summarise(
    `Mean Incidence Rate` = mean(value, na.rm = TRUE),
    minimum = min(value, na.rm = TRUE),
    median = median(value, na.rm = TRUE),
    max = max(value, na.rm = TRUE),
    sd = sd(value, na.rm = TRUE)
  ) %>%
  arrange(desc(`Mean Incidence Rate`)) %>%
  mutate(
    name = case_when(
     name ==  "human_incidence" ~   "Human",
      name =="catt_incidence"  ~  "Cattle",
      name == "goat_incidence" ~  "Goat",
      name == "cam_incidence"  ~ "Camel",
      name == "shp_incidence" ~  "Sheep"
     ),
    Cases = comma(`Mean Incidence Rate`),
    Minimum = comma(minimum),
    Median = comma(median),
    Maximum = comma(max),
    `Standard Deviation` = comma(sd)
  ) |>
  select(Species = name,
         `Mean Incidence Rate`,
         Minimum,
         Median,
         Maximum,
         `Standard Deviation`),
  "descriptive_table.csv")
```

```r
# Cases per year per county
df_tot_cases_spatial <- df_incidence2 |>
  group_by(year = year(as.Date(date)), county) |>
  summarise(across(contains("cases"), ~ sum(., na.rm = T))) |>
  mutate(across(contains('cases'), ~ifelse(. == 0, NA, .)))

# Population per year, per county
df_pop_spatial <- df_incidence2 |>
  select(date, county, contains("pop")) %>%
  distinct(.) |>
  as_tibble() |>
  group_by(year = year(as.Date(date)), county) %>%
  summarise(across(where(is.numeric), ~unique(.))) |>
  mutate(across(contains('cases'), ~ifelse(. == 0, NA, .)))


# Individual Cases per year per county month
df_tot_cases_spatial_month <- df_incidence2 |>
  group_by(date, county) |>
  summarise(across(contains("cases"), ~ sum(., na.rm = T))) |>
  mutate(year = year(as.Date(date))) |>
  merge(df_pop_spatial, by = c("year", "county")) |>
  filter(!is.na(year)) |>
  mutate(
    human_incidence = round((hum_cases / pop) * 1000, 4),
    catt_incidence = round((catt_cases / catt_pop) * 1000000, 4),
    cam_incidence = round((cam_cases / cam_pop) * 1000000, 4),
    goat_incidence = round((goat_cases / goat_pop) * 1000000, 4),
    shp_incidence = round((shp_cases / sheep_pop) * 1000000, 4)
  ) |>
  select(year, date, county, contains(c("incidence", "cases"))) |>
  as_tibble()

write.csv(df_tot_cases_spatial_month, "df_tot_cases_spatial_month.csv")

df_tot_cases_spatial_month_complete <- df_tot_cases_spatial_month |>
  select(year, date, county, human_incidence, goat_incidence, catt_incidence,
         goat_cases, catt_cases, hum_cases )

write.csv(df_tot_cases_spatial_month_complete, "df_tot_cases_spatial_month_complete.csv")


# Combined cases
df_spatial_cum_month <- df_tot_cases_spatial_month |>
  merge(df_pop_spatial, by = c("year", "county")) |>
  rowwise() |>
  mutate(
    animal_cases = sum(catt_cases, goat_cases, shp_cases, cam_cases, na.rm = T),
    animal_pop = sum(catt_pop, goat_pop, sheep_pop, cam_pop, na.rm = T),
    animal_incidence = round((animal_cases / animal_pop) * 1000000, 4),
    human_incidence = round((hum_cases / pop) * 1000, 4)
  ) |>
  select(date, year, county, contains(c("incidence", "cases"))) |>
```

```r
  as_tibble()
write.csv(df_spatial_cum_month, "df_spatial_cum_month.csv")


# Population per year, per county
df_pop_spatial <- df_incidence2 |>
  select(date, county, contains("pop")) %>%
  distinct(.) |>
  as_tibble() |>
  group_by(year = year(date), county) %>%
  summarise(across(where(is.numeric), ~unique(.))) |>
  mutate(across(contains('cases'), ~ifelse(. == 0, NA, .)))

df_spatial <- df_tot_cases_spatial |>
  merge(df_pop_spatial, by = c("year", "county")) |>
  filter(!is.na(year)) |>
  mutate(
    human_incidence = round((hum_cases / pop) * 1000, 4),
    catt_incidence = round((catt_cases / catt_pop) * 1000000, 4),
    cam_incidence = round((cam_cases / cam_pop) * 1000000, 4),
    goat_incidence = round((goat_cases / goat_pop) * 1000000, 4),
    shp_incidence = round((shp_cases / sheep_pop) * 1000000, 4)
  ) |>
  select(year, county, contains("incidence")) |>
  #mutate(across(is.numeric, ~ifelse(is.na(.), 0, .))) |>
  as_tibble()

df_spatial_cum <- df_tot_cases_spatial |>
  merge(df_pop_spatial, by = c("year", "county")) |>
  rowwise() |>
  mutate(
    animal_cases = sum(catt_cases, goat_cases, shp_cases, cam_cases, na.rm = T),
    animal_pop = sum(catt_pop, goat_pop, sheep_pop, cam_pop, na.rm = T),
    animal_cases = ifelse(animal_cases == 0, NA, animal_cases),
    animal_incidence = round((animal_cases / animal_pop) * 1000000, 4),
    human_incidence = round((hum_cases / pop) * 1000, 4)
  ) |>
  select(year, county, contains("incidence")) |>
  as_tibble()

# Checking for mismatch of county names in the shapefiles and in our data
setdiff(shp$Name, df_spatial$county)
```

```
## [1] "Muranga"
```

```r
# Replacing Muranga to Murang'a
shp <- shp |>
  mutate(Name = ifelse(Name == "Muranga", "Murang'a", Name))
setdiff(shp$Name, df_spatial$county)
```

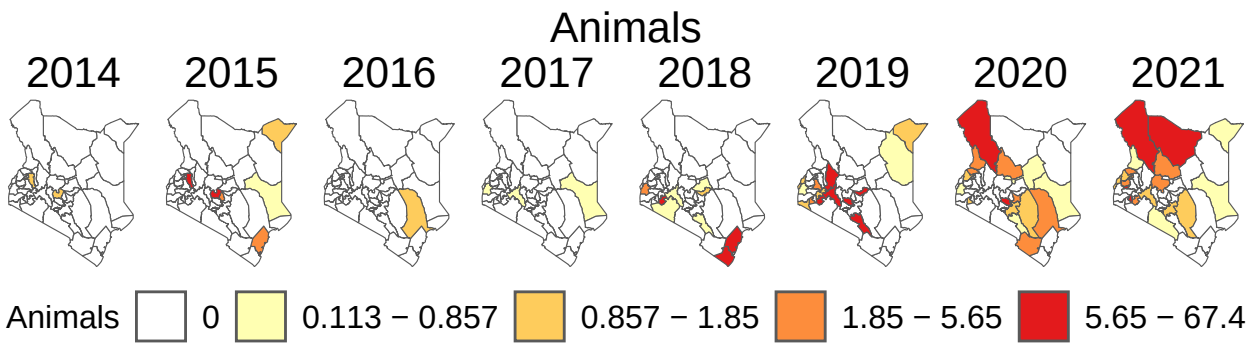
```
## character(0)
```

```r
length(unique(df_spatial$county))
```

```
## [1] 47
```

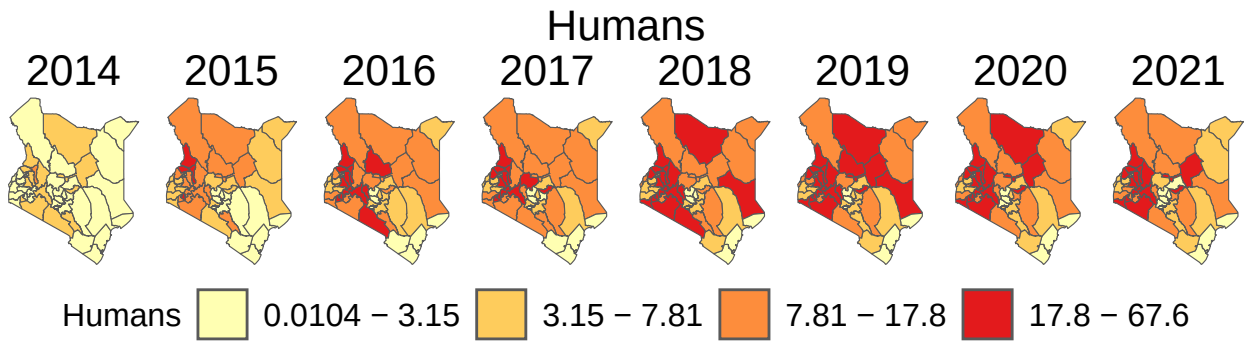
```r
# Merging
df_spatial_merged <- df_spatial |>
  merge(shp, by.x = "county", by.y = 'Name')
indi_incidence_columns <- grep("incidence", names(df_spatial_merged), value = TRUE)

df_spatial_merged <- df_spatial_merged |>
    mutate(across(all_of(indi_incidence_columns),
                  ~ cut(., breaks = quantile(., na.rm = TRUE), include.lowest = TRUE),
                  .names = "{col}_range")) |>
  st_as_sf()

df_spatial_merged_cum <- df_spatial_cum |>
  merge(shp, by.x = "county", by.y = 'Name')

all_incidence_columns <- grep("incidence", names(df_spatial_merged_cum), value = TRUE)
df_spatial_merged_cum <- df_spatial_merged_cum |>
      mutate(across(all_of(all_incidence_columns),
                  ~ cut(., breaks = quantile(., na.rm = TRUE), include.lowest = TRUE),
                  .names = "{col}_range")) |>
  st_as_sf()

# Plotting

# Convert year to factor for better plotting
df_spatial_merged$year <- as.factor(df_spatial_merged$year)
df_spatial_merged_cum$year <- as.factor(df_spatial_merged_cum$year)

# All animals incidence
cate_animal <- length(levels(df_spatial_merged_cum$animal_incidence_range))
animals <- df_spatial_merged_cum |>
  mutate(animal_incidence_range = ifelse(is.na(animal_incidence_range),
                                         "0",
                                         as.character(animal_incidence_range)) %>%
           factor(., levels = c(
             "0",
             "[0.113,0.857]",
             "(0.857,1.85]",
             "(1.85,5.65]",
             "(5.65,67.4]"
           ))) |>
  ggplot() +
  geom_sf(aes(fill = animal_incidence_range)) +
  scale_fill_manual(values = c("white", brewer.pal(cate_animal, "YlOrRd")),
                    labels = function(breaks) {
       str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      },

                    na.value = "white") +
  theme_void() +
```

```
  facet_wrap( ~ year, nrow = 1) +
  theme(
    plot.title = element_text(
      color = "black",
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Animals") +
  labs(fill = "Animals")
animals
```

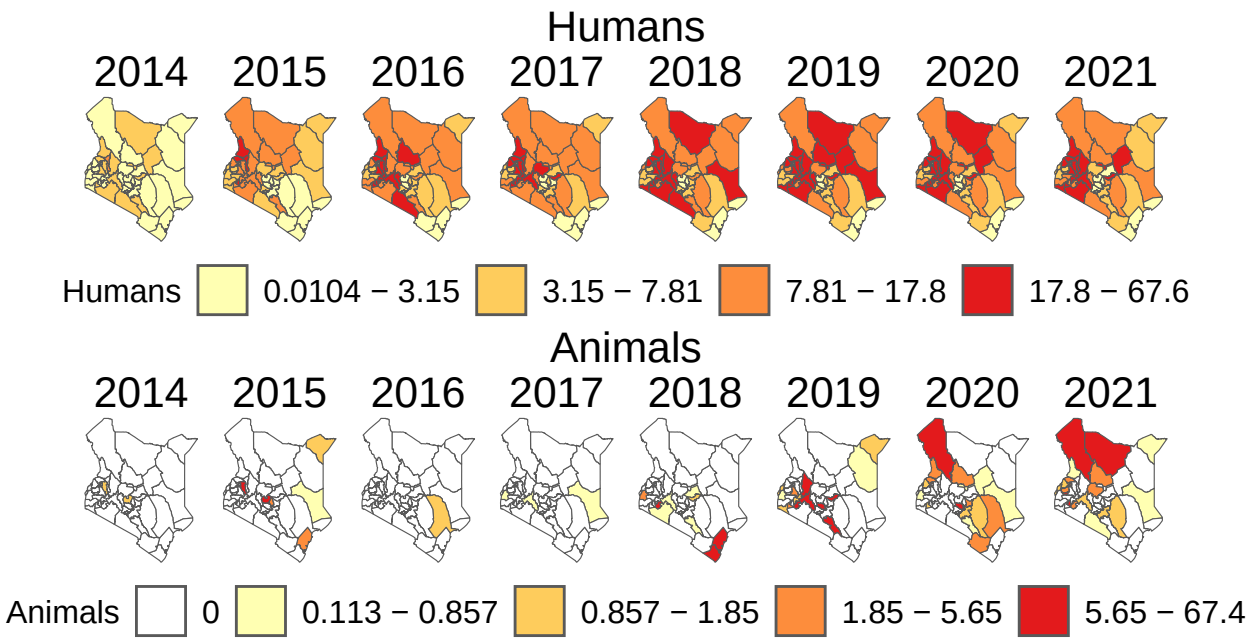



```
# Humans
cate_human <-
  length(levels(df_spatial_merged_cum$human_incidence_range))

humans <- df_spatial_merged_cum |>
  # mutate(human_incidence_range = ifelse(is.na(human_incidence_range),
  #                                       "0",
  #                                       as.character(human_incidence_range)) %>%
```

```r
  #           factor(., levels = c(
  #             "0",
  #             "[0.113,0.865]",
  #             "(0.865,1.85]",
  #             "(1.85,5.62]",
  #             "(5.62,67.4]"
  #           ))) |>
  ggplot() +
  geom_sf(aes(fill = human_incidence_range)) +
  scale_fill_manual(
    values = c(brewer.pal(cate_human, "YlOrRd")),
    labels =
      function(breaks) {
        str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      }

    ,

    na.value = "white"
  ) +
  theme_void() +
  facet_wrap(~ year, nrow = 1) +
  theme(
    plot.title = element_text(
      color = "black",
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Humans") +
  labs(fill = "Humans")
humans
```



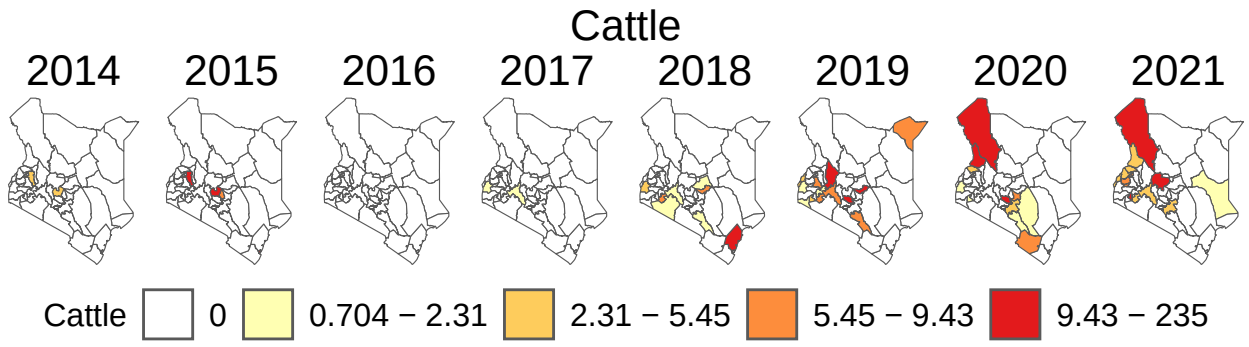

```
animals_humans <-
  wrap_plots(humans,
             animals,
             ncol = 1,
             guides = "keep") +
  plot_annotation(caption = "For humans, the incidence rate is per
                  1,000 population while for animal,
                  the incidence rate is per 1,000,000 population
                  ") &
  theme(plot.caption = element_text(size = 16, colour = "black"))

animals_humans
```



For humans, the incidence rate is per 1,000 population while for animal, the incidence rate is per 1,000,000 population

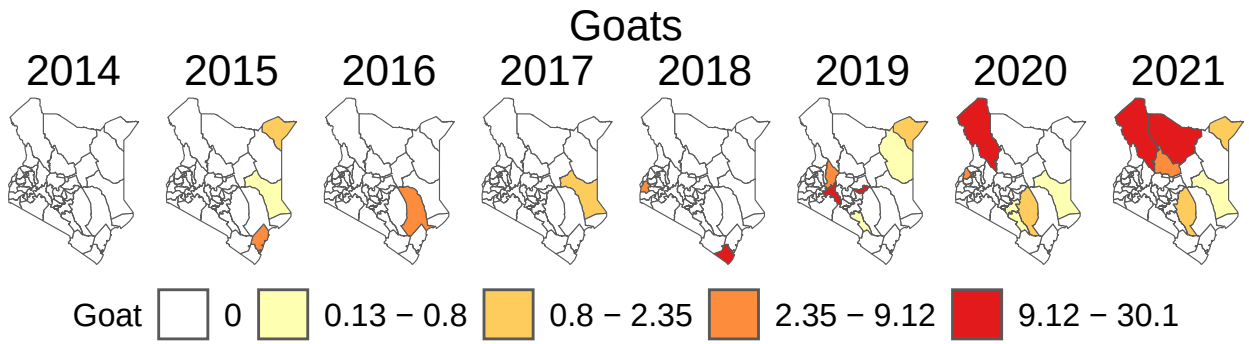
```
cate_catt <- length(levels(df_spatial_merged$catt_incidence_range))
cattle <- df_spatial_merged |>
  mutate(catt_incidence_range = ifelse(is.na(catt_incidence_range),
                                       "0",
                                       as.character(catt_incidence_range)) %>%
           factor(., levels = c(
             "0",
             "[0.704,2.31]",
             "(2.31,5.45]",
             "(5.45,9.43]",
             "(9.43,235]"
           ))) |>
  ggplot() +
  geom_sf(aes(fill = catt_incidence_range)) +
  scale_fill_manual(values = c("white", brewer.pal(cate_catt, "YlOrRd")),
                    labels =
      function(breaks) {
        str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      },

                    na.value = "white") +
  theme_void() +
  facet_wrap( ~ year, nrow = 1) +
  theme(
    plot.title = element_text(
      color = "black",
```

```
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Cattle") +
  labs(fill = "Cattle")
cattle
```



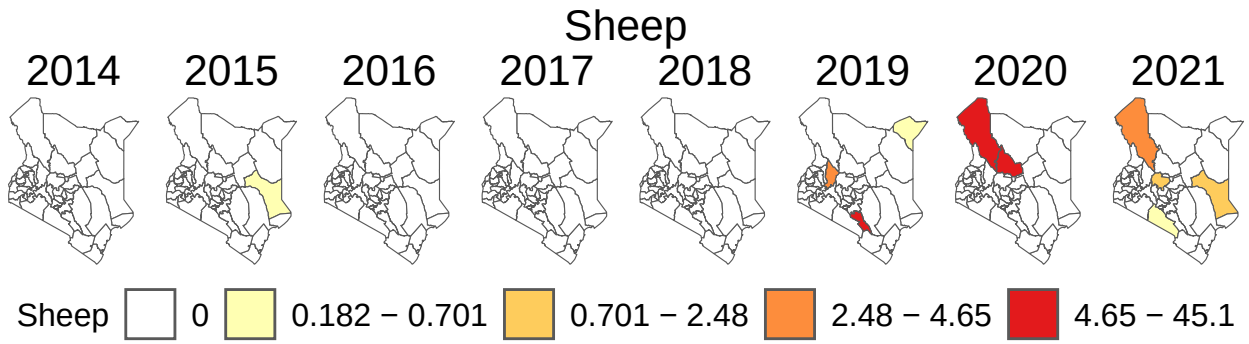

```
# Goats
cate_goat <- length(levels(df_spatial_merged$goat_incidence_range))
goat <- df_spatial_merged |>
  mutate(goat_incidence_range = ifelse(is.na(goat_incidence_range),
                                       "0",
                                       as.character(goat_incidence_range)) %>%
           factor(., levels = c(
             "0",
             "[0.13,0.8]",
             "(0.8,2.35]",
             "(2.35,9.12]",
             "(9.12,30.1]"
```

```
          ))) |>
  ggplot() +
  geom_sf(aes(fill = goat_incidence_range)) +
  scale_fill_manual(values = c("white", brewer.pal(cate_goat, "YlOrRd")),
                        labels =
      function(breaks) {
        str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      },

                    na.value = "white") +
  theme_void() +
  facet_wrap( ~ year, nrow = 1) +
  theme(
    plot.title = element_text(
      color = "black",
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Goats") +
  labs(fill = "Goat")
goat
```



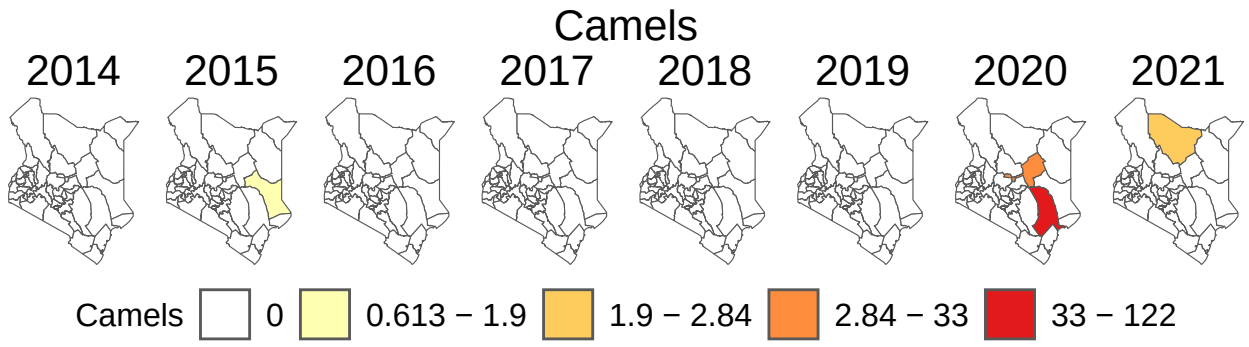

```
# sheep
cate_shp <- length(levels(df_spatial_merged$shp_incidence_range))
sheep <- df_spatial_merged |>
  mutate(shp_incidence_range = ifelse(is.na(shp_incidence_range),
                                      "0",
                                      as.character(shp_incidence_range)) %>%
           factor(., levels = c(
             "0",
             "[0.182,0.701]",
             "(0.701,2.48]",
             "(2.48,4.65]",
             "(4.65,45.1]"
           ))) |>
  ggplot() +
  geom_sf(aes(fill = shp_incidence_range)) +
  scale_fill_manual(values = c("white", brewer.pal(cate_shp, "YlOrRd")),
                    labels =
      function(breaks) {
        str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      },

                    na.value = "white") +
  theme_void() +
  facet_wrap( ~ year, nrow = 1) +
  theme(
    plot.title = element_text(
```

```
      color = "black",
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Sheep") +
  labs(fill = "Sheep")
sheep
```




```
# Camels
cate_cam <- length(levels(df_spatial_merged$cam_incidence_range))
camels <- df_spatial_merged |>
  mutate(cam_incidence_range = ifelse(is.na(cam_incidence_range),
                                      "0",
                                      as.character(cam_incidence_range)) %>%
           factor(., levels = c(
             "0",
             "[0.613,1.9]",
             "(1.9,2.84]",
             "(2.84,33]",
```

```
            "(33,122]"
          ))) |>
  ggplot() +
  geom_sf(aes(fill = cam_incidence_range)) +
  scale_fill_manual(values = c("white", brewer.pal(cate_cam, "YlOrRd")),
                        labels =
      function(breaks) {
        str_replace_all(breaks, "\\[|\\)|\\]|\\(", "") %>% str_replace_all(., ",", " - ")
      },

                    na.value = "white") +
  theme_void() +
  facet_wrap( ~ year, nrow = 1) +
  theme(
    plot.title = element_text(
      color = "black",
      hjust = .5,
      size = 16
    ),
    legend.position = "bottom",
    legend.text = element_text(size = 12),
    legend.title = element_text(size = 12, colour = "black"),
    legend.key.size = unit(0.7, "cm"),
    strip.text = element_text(colour = "black", size = 16)
  ) +
  ggtitle("Camels") +
  labs(fill = "Camels")
camels
```

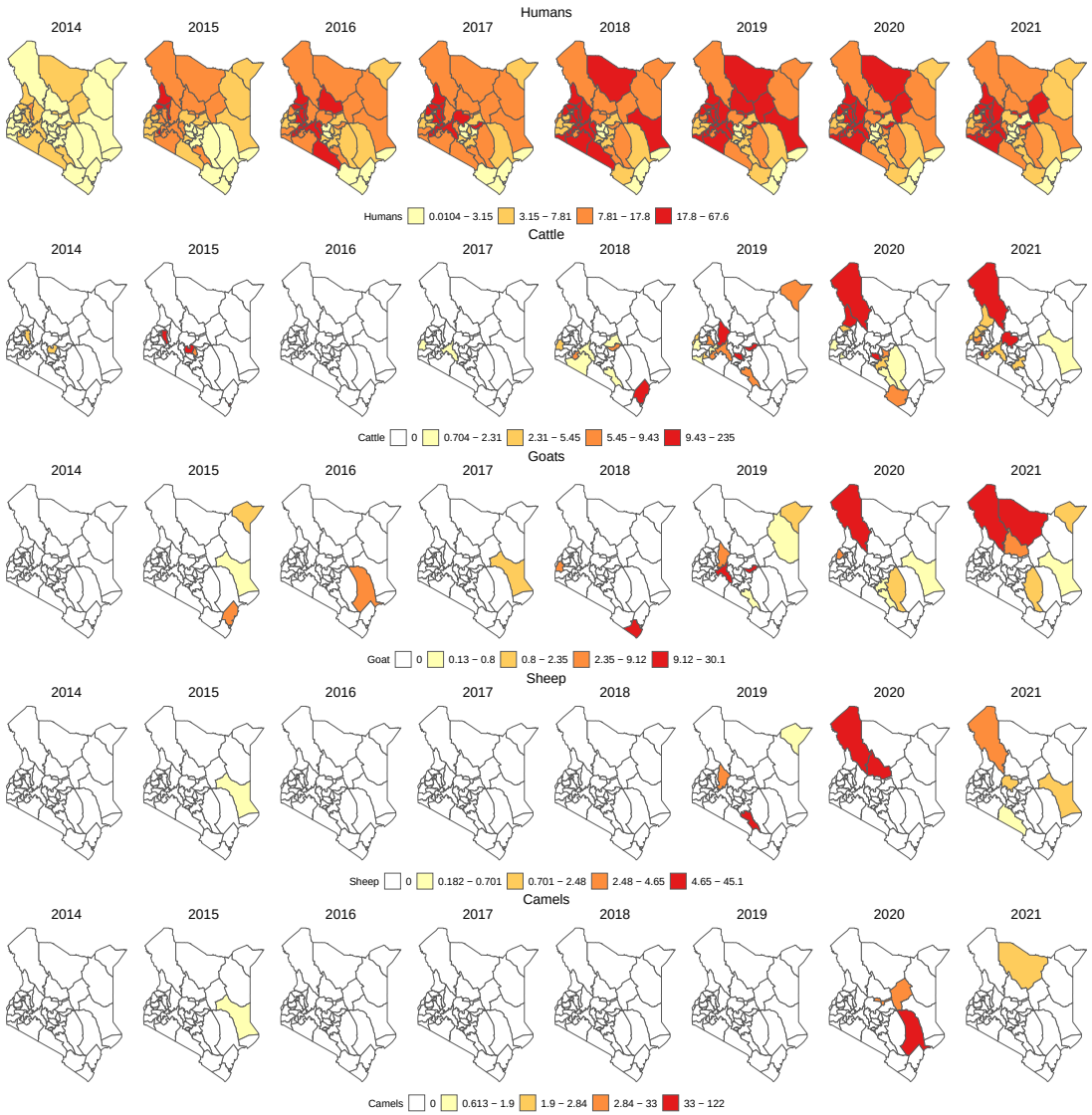



```
all_plots <-
  wrap_plots(humans,
             cattle,
             goat,
             sheep,
             camels,
             ncol = 1,
             guides = "keep") &
  theme(plot.caption = element_text(size = 16, colour = "black"))


dev.off()
```

```
## null device
##           1
```

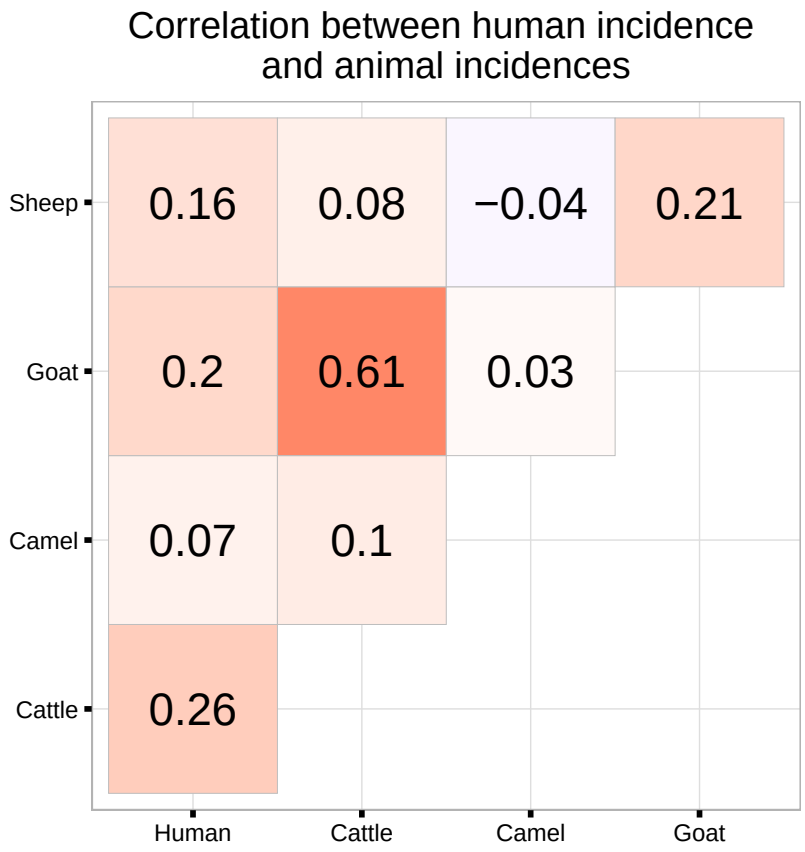
```
ggsave("all_plots.png", dpi = 1e3, height = 18, width = 20)
```

```
all_plots
```

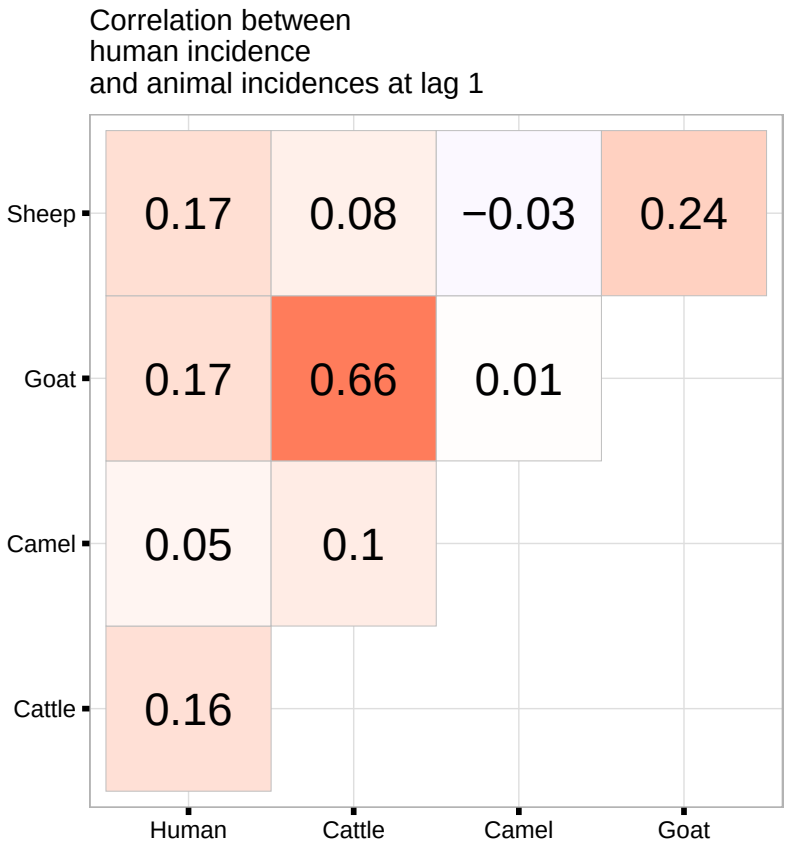



```
# Correlation Plot
df_1_trend <- df_1_trend |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))

cor_lag <- df_1_trend %>%
  as_tibble() %>%
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between human incidence \nand animal incidences",
```

```
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 35
    ),
    plot.subtitle = element_text(
      color = "black",
      hjust = 0.5,
      size = 14
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag
```



```r
# Correlation Plot at lag 1
cor_lag1 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 1))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 1",
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag1
```




```
# Correlation Plot at lag 2
cor_lag2 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 2))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 2",
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
```

```
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag2
```


```
# Correlation Plot at lag 3
cor_lag3 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 3))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
```

```
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 3",
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag3
```

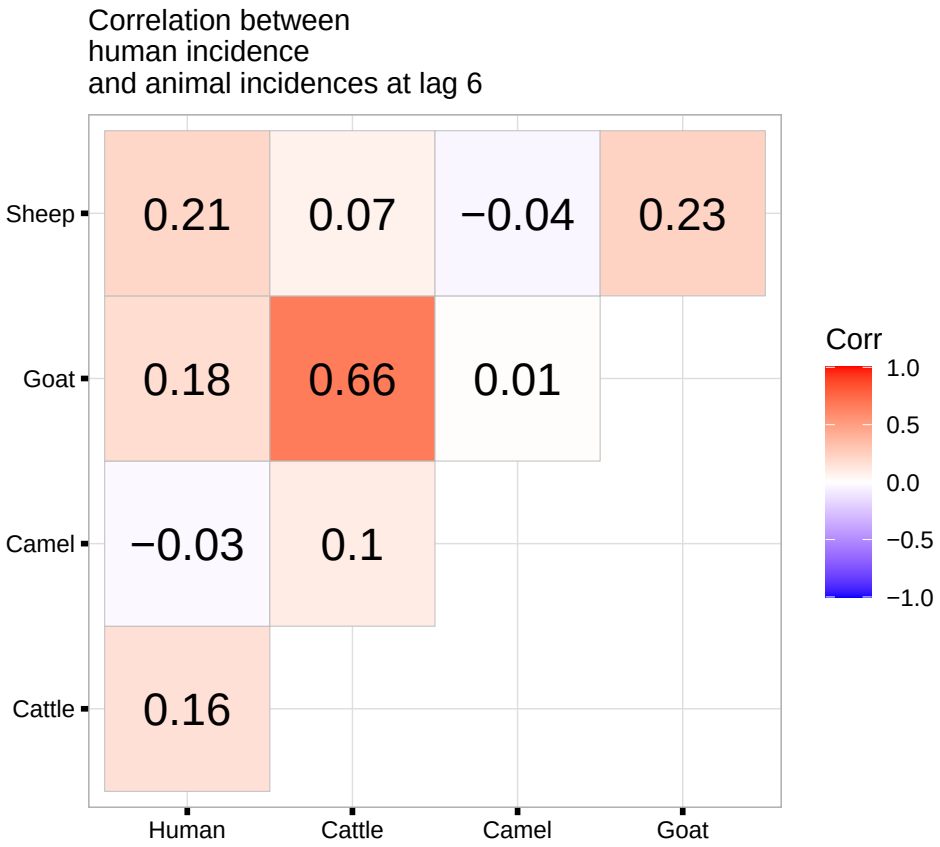
```r
# Correlation Plot at lag 4
cor_lag4 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 4))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 4",
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag4
```


```
# Correlation Plot at lag 5
cor_lag5 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 5))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 5",
       x = NULL,
       y = NULL) +
  guides(fill = "none") +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
```

```
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag5
```


```
# Correlation Plot at lag 6
cor_lag6 <- df_1_trend %>%
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ lag(., n = 6))) |>
  na.omit() |>
  select(-date) %>%
  setNames(c("Human",
             "Cattle",
             "Camel",
             "Goat",
             "Sheep")) |>
  cor() %>%
```

```
  ggcorrplot::ggcorrplot(type = "upper",
                         lab = TRUE,
                         lab_size = 6) +
  theme_light() +
  labs(subtitle = "Correlation between\nhuman incidence \nand animal incidences at lag 6",
       x = NULL,
       y = NULL) +
  theme(
    strip.background = element_rect(fill = "white", colour = "grey"),
    strip.text = element_text(color = "black", size = 12),
    axis.title = element_text(colour = "black"),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black", linewidth = 1),
    plot.title = element_text(
      color = "black",
      hjust = 0.5,
      size = 20
    ),
    axis.title.y = element_text(color = "black", size = 10),
    legend.position = "right",
    legend.text = element_text(color = "black")
  )
cor_lag6
```



```r
all_cols <- wrap_plots(
  cor_lag,
  cor_lag1,
  cor_lag2,
  cor_lag3,
  cor_lag4,
  cor_lag5,
  cor_lag6,
  ncol = 3,
  guides = "collect"
) |>
  plot_grid(
    rel_widths = c(7, 7,7)
  )
all_cols <- all_cols + theme(plot.title = element_text(size = 16),
                             axis.text.y = element_text(color = 'black', size = 13))
```

```r
lag_values <- 0:6  # Assuming you want lag values from 0 to 6
cor_dats <- list(

    cor_lag$data,
    cor_lag1$data,
    cor_lag2$data,
    cor_lag3$data,
    cor_lag4$data,
    cor_lag5$data,
    cor_lag6$data
  )
result_table <- tibble(
  Lag = lag_values,
  `Average Correlation` = cor_dats %>%
    map(~ filter(.x, Var1 == "Human")) %>%
    map_dbl(~mean(.$value))
) |>
  knitr::kable(align = "l",
               caption = "Average correlation between human incidence and other species incidence",
               format = "pipe",
               latex_options = "hold_position")

print(result_table)
```

```
##
##
## Table: Average correlation between human incidence and other species incidence
##
## |Lag |Average Correlation |
## |:---|:-------------------|
## |0   |0.1725              |
## |1   |0.1375              |
## |2   |0.1525              |
## |3   |0.1725              |
## |4   |0.1450              |
## |5   |0.1175              |
```

```
## |6   |0.1300                |
```

```
 # This helps us to choose the lag with the highest average correlation
```

```
## This model fits the data without differencing, at difference lags, (0-3) and for individual
## animal incidences.

df_1_trend <- df_1_trend |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))

# Models with NA
run_lag_models <- function(df, max_lag = 3, ...) {
  suppressMessages({
  result_df <- tibble()

  for (lag_value in 0:max_lag) {
    df_lagged <- df |>
      as_tibble() %>%
      mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
                list( ~ lag(., n = lag_value))) |>
      na.omit() |>
      mutate(date = as.Date(date))

    mod <- df_lagged |>
      as_tsibble() |>
      model(
        TSLM(
          (human_incidence) ~ cam_incidence + shp_incidence + catt_incidence + goat_incidence
        )
      ) |>
      report()

    mod_results <- tidy(mod) %>%
      select(-.model) %>%
      as_tibble() %>%
      mutate(term = case_when(
        term == "goat_incidence" ~ "Goat Incidence",
        term == "catt_incidence" ~ "Cattle incidence",
        term == "shp_incidence" ~ "Sheep incidence",
        term == "cam_incidence" ~ "Camel incidence",
        TRUE ~ as.character(term)
      ),
      variable = term
      ) |>  select(6, 2:5) |>
      group_by(variable) %>%
      mutate(
        conf_low = min(estimate - std.error * 1.645),
        conf_high = max(estimate + std.error * 1.645)
      ) %>%
      mutate(lag = lag_value)

    adj_r_squared <- glance(mod) %>%
      select(r_squared, AIC, adj_r_squared)
```

```
    mod_results <- bind_cols(mod_results, adj_r_squared) |>
      mutate(across(c(estimate, std.error, statistic, p.value, conf_low, conf_high, adj_r_squared), ~rou
      mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
      select(c(1:8, 12, 9:11))

    result_df <- bind_rows(result_df, mod_results)
  }
  })
  return(result_df)
}

# Models with complete cases

complete_run_lag_models <- function(df, max_lag = 3, ...) {
  suppressMessages({
  result_df <- tibble()

  for (lag_value in 0:max_lag) {
    df_lagged <- df |>
      as_tibble() %>%
      mutate_at(vars(catt_incidence, goat_incidence),
                list( ~ lag(., n = lag_value))) |>
      na.omit() |>
      mutate(date = as.Date(date))

    mod <- df_lagged |>
      as_tsibble() |>
      model(
        TSLM(
          human_incidence ~ catt_incidence + goat_incidence
        )
      ) |>
      report()

    mod_results <- tidy(mod) %>%
      select(-.model) %>%
      as_tibble() %>%
      mutate(term = case_when(
        term == "goat_incidence" ~ "Goat Incidence",
        term == "catt_incidence" ~ "Cattle incidence",
        TRUE ~ as.character(term)
      ),
      variable = term
      ) |>  select(6, 2:5) |>
      group_by(variable) %>%
      mutate(
        conf_low = min(estimate - std.error * 1.645),
        conf_high = max(estimate + std.error * 1.645)
      ) %>%
      mutate(lag = lag_value)

    adj_r_squared <- glance(mod) %>%
      select(r_squared, AIC, adj_r_squared)
```

```
    mod_results <- bind_cols(mod_results, adj_r_squared) |>
      mutate(across(c(estimate, std.error, statistic, p.value, conf_low, conf_high, adj_r_squared), ~ro
      mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
      select(c(1:8, 12, 9:11))

    result_df <- bind_rows(result_df, mod_results)
  }
  })
  return(result_df)
}


non_diff_indivi_with_NA <- run_lag_models(df_1_trend)
```

```
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.71818 -0.17309  0.08595  0.24882  0.54929
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.86593    0.03911  22.140   <2e-16 ***
## cam_incidence   0.12734    0.23991   0.531   0.5969
## shp_incidence   0.89421    0.67914   1.317   0.1913
## catt_incidence  0.38095    0.22634   1.683   0.0958 .
## goat_incidence  0.12248    0.36506   0.336   0.7380
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3344 on 91 degrees of freedom
## Multiple R-squared: 0.08894, Adjusted R-squared: 0.04889
## F-statistic: 2.221 on 4 and 91 DF, p-value: 0.072848
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74028 -0.12317  0.06415  0.25549  0.48306
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.88925    0.03912  22.734   <2e-16 ***
## cam_incidence   0.11185    0.24062   0.465    0.643
## shp_incidence   0.94577    0.68562   1.379    0.171
## catt_incidence  0.16715    0.24047   0.695    0.489
## goat_incidence  0.20366    0.41886   0.486    0.628
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.3341 on 90 degrees of freedom
## Multiple R-squared: 0.05402, Adjusted R-squared: 0.01198
## F-statistic: 1.285 on 4 and 90 DF, p-value: 0.28187
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73218 -0.14247  0.06283  0.23954  0.50872
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.89208    0.03822  23.344   <2e-16 ***
## cam_incidence   0.11518    0.23347   0.493   0.6230
## shp_incidence   1.14578    0.66541   1.722   0.0886 .
## catt_incidence  0.24761    0.23341   1.061   0.2916
## goat_incidence  0.08271    0.40642   0.204   0.8392
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3242 on 89 degrees of freedom
## Multiple R-squared: 0.07092, Adjusted R-squared: 0.02917
## F-statistic: 1.699 on 4 and 89 DF, p-value: 0.15739
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73377 -0.11367  0.05973  0.22663  0.49453
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.89367    0.03735  23.924   <2e-16 ***
## cam_incidence   0.14030    0.22663   0.619    0.537
## shp_incidence   0.94356    0.64606   1.460    0.148
## catt_incidence  0.34316    0.22665   1.514    0.134
## goat_incidence  0.06955    0.39451   0.176    0.860
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3147 on 88 degrees of freedom
## Multiple R-squared: 0.08695, Adjusted R-squared: 0.04545
## F-statistic: 2.095 on 4 and 88 DF, p-value: 0.088176
```

```
non_diff_indivi_without_NA <-
  complete_run_lag_models(df_1_trend_complete)
```

```
## Series: human_incidence
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.88676 -0.04431  0.06685  0.17300  0.38067
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.042492   0.063230  16.487   <2e-16 ***
## catt_incidence  0.130671   0.210765   0.620    0.539
## goat_incidence -0.003588   0.313046  -0.011    0.991
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2841 on 36 degrees of freedom
## Multiple R-squared: 0.01348, Adjusted R-squared: -0.04132
## F-statistic: 0.246 on 2 and 36 DF, p-value: 0.7832
## Series: human_incidence
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93069 -0.07315  0.03081  0.15032  0.30364
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.10386    0.05485  20.127   <2e-16 ***
## catt_incidence -0.20289    0.19345  -1.049    0.301
## goat_incidence  0.32852    0.30550   1.075    0.290
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2455 on 35 degrees of freedom
## Multiple R-squared: 0.03973, Adjusted R-squared: -0.01514
## F-statistic: 0.7241 on 2 and 35 DF, p-value: 0.49187
## Series: human_incidence
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.58337 -0.07528  0.02456  0.12638  0.30785
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.12055    0.04537  24.699   <2e-16 ***
## catt_incidence -0.02723    0.16506  -0.165    0.870
## goat_incidence  0.04170    0.24723   0.169    0.867
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1955 on 34 degrees of freedom
## Multiple R-squared: 0.001129,    Adjusted R-squared: -0.05763
## F-statistic: 0.01921 on 2 and 34 DF, p-value: 0.98098
## Series: human_incidence
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.52343 -0.08308  0.01215  0.10229  0.29819
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.13925    0.03942  28.900   <2e-16 ***
## catt_incidence -0.02180    0.14644  -0.149    0.883
## goat_incidence  0.00607    0.23470   0.026    0.980
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1696 on 33 degrees of freedom
## Multiple R-squared: 0.0007565,   Adjusted R-squared: -0.0598
## F-statistic: 0.01249 on 2 and 33 DF, p-value: 0.98759
```

```
diff_indivi <- run_lag_models(df_1_trend_diff |>
                              mutate(across(contains('incidence'), ~ifelse(is.na(.), 0, .))))
```

```
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.397324 -0.064865  0.008085  0.078366  0.226085
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.01041    0.01276   0.816   0.4164
## cam_incidence        NA         NA      NA       NA
## shp_incidence   0.18745    0.36379   0.515   0.6076
## catt_incidence  0.15320    0.07559   2.027   0.0456 *
## goat_incidence  0.10563    0.15055   0.702   0.4847
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1235 on 91 degrees of freedom
## Multiple R-squared: 0.05053, Adjusted R-squared: 0.01923
## F-statistic: 1.614 on 3 and 91 DF, p-value: 0.19151
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.411884 -0.060546  0.002184  0.080816  0.230416
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.006084   0.012653   0.481   0.6318
## cam_incidence         NA         NA      NA       NA
## shp_incidence  -0.381098   0.358955  -1.062   0.2912
## catt_incidence -0.133678   0.074580  -1.792   0.0764 .
## goat_incidence  0.311968   0.148539   2.100   0.0385 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1218 on 90 degrees of freedom
## Multiple R-squared: 0.08587, Adjusted R-squared: 0.0554
## F-statistic: 2.818 on 3 and 90 DF, p-value: 0.043503
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.415056 -0.065048  0.004744  0.081527  0.227244
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.009256   0.013317   0.695    0.489
## cam_incidence         NA         NA      NA       NA
## shp_incidence  -0.170616   0.375769  -0.454    0.651
## catt_incidence  0.043912   0.078070   0.562    0.575
## goat_incidence -0.089412   0.155490  -0.575    0.567
##
## Residual standard error: 0.1275 on 89 degrees of freedom
## Multiple R-squared: 0.009229,    Adjusted R-squared: -0.02417
## F-statistic: 0.2763 on 3 and 89 DF, p-value: 0.84233
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41730 -0.06594  0.00399  0.08013  0.22499
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.011510   0.013391   0.860    0.392
## cam_incidence         NA         NA      NA       NA
## shp_incidence   0.412086   0.375800   1.097    0.276
## catt_incidence  0.063272   0.078073   0.810    0.420
## goat_incidence -0.003601   0.155495  -0.023    0.982
##
## Residual standard error: 0.1275 on 88 degrees of freedom
## Multiple R-squared: 0.01945, Adjusted R-squared: -0.01398
## F-statistic: 0.5819 on 3 and 88 DF, p-value: 0.62845
```

```
write_csv(non_diff_indivi_with_NA |>
  select(-r_squared, -AIC, -adj_r_squared), "non_diff_individual_with_NA.csv")
write_csv(non_diff_indivi_without_NA |>
  select(-r_squared, -AIC, -adj_r_squared), "non_diff_individual_without_NA.csv")

write_csv(diff_indivi, "diff_individual.csv")


## The following function fits the model, without differencing, at difference lags, (0-3) and for
## animal incidences combined
```

```r
lag_models_full <- function(df, max_lag = 3, ...) {
  result_df <- tibble()

  for (lag_value in 0:max_lag) {
    df_lagged <- df |>
      as_tibble() %>%
      mutate_at(vars(animal_incidence),
                list(~ lag(., n = lag_value))) |>
      na.omit() |>
      mutate(date = as.Date(date))

    mod <- df_lagged |>
      as_tsibble() |>
      model(TSLM((human_incidence) ~ animal_incidence)) |>
      report()

    mod_results <- tidy(mod) %>%
      select(-.model) %>%
      as_tibble() %>%
      mutate(
        term = case_when(
          term == "animal_incidence" ~ "Animal Incidence",
          TRUE ~ as.character(term)
        ),
        variable = term
      ) |>  select(6, 2:5) |>
      group_by(variable) %>%
      mutate(
        conf_low = min(estimate - std.error * 1.645),
        conf_high = max(estimate + std.error * 1.645)
      ) %>%
      mutate(lag = lag_value)

    adj_r_squared <- glance(mod) |>
      select(r_squared, adj_r_squared, AIC) |>
      as.data.frame()


    mod_results <- bind_cols(mod_results, adj_r_squared) |>
      mutate(across(
        c(
          estimate,
          std.error,
          statistic,
          p.value,
          conf_low,
          conf_high,
          adj_r_squared
        ),
        ~ round(., 3)
      )) |>
      mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant"))
```

```
    result_df <- bind_rows(result_df, mod_results)

  }

  return(result_df)
}

non_diff_full_with_NA <- lag_models_full(df_cum_trend |>
                                mutate(across(contains('incidence'), ~ifelse(is.na(.), 0, .))))
```

```
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7196 -0.1612  0.1023  0.2554  0.5478
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.87176    0.03841  22.697  < 2e-16 ***
## animal_incidence  1.04921    0.37752   2.779  0.00658 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3314 on 94 degrees of freedom
## Multiple R-squared: 0.07593, Adjusted R-squared: 0.0661
## F-statistic: 7.724 on 1 and 94 DF, p-value: 0.0065808
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74616 -0.15249  0.07314  0.25649  0.47682
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.89434    0.03835  23.321   <2e-16 ***
## animal_incidence  0.77098    0.38459   2.005   0.0479 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3309 on 93 degrees of freedom
## Multiple R-squared: 0.04142, Adjusted R-squared: 0.03111
## F-statistic: 4.019 on 1 and 93 DF, p-value: 0.047907
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73943 -0.14117  0.08695  0.23543  0.49755
```

```
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.89933    0.03763  23.900   <2e-16 ***
## animal_incidence  0.82439    0.37537   2.196   0.0306 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3225 on 92 degrees of freedom
## Multiple R-squared: 0.04982, Adjusted R-squared: 0.03949
## F-statistic: 4.823 on 1 and 92 DF, p-value: 0.03059
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74032 -0.12022  0.08039  0.21808  0.48798
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.90022    0.03667  24.547  < 2e-16 ***
## animal_incidence  0.96019    0.36388   2.639  0.00979 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3121 on 91 degrees of freedom
## Multiple R-squared: 0.07108, Adjusted R-squared: 0.06087
## F-statistic: 6.963 on 1 and 91 DF, p-value: 0.0097893
```

```
non_diff_full_without_NA <- lag_models_full(df = df_cum_trend)
```

```
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7196 -0.1612  0.1023  0.2554  0.5478
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.87176    0.03841  22.697  < 2e-16 ***
## animal_incidence  1.04921    0.37752   2.779  0.00658 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3314 on 94 degrees of freedom
## Multiple R-squared: 0.07593, Adjusted R-squared: 0.0661
## F-statistic: 7.724 on 1 and 94 DF, p-value: 0.0065808
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74616 -0.15249  0.07314  0.25649  0.47682
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.89434    0.03835  23.321   <2e-16 ***
## animal_incidence  0.77098    0.38459   2.005   0.0479 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3309 on 93 degrees of freedom
## Multiple R-squared: 0.04142, Adjusted R-squared: 0.03111
## F-statistic: 4.019 on 1 and 93 DF, p-value: 0.047907
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73943 -0.14117  0.08695  0.23543  0.49755
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.89933    0.03763  23.900   <2e-16 ***
## animal_incidence  0.82439    0.37537   2.196   0.0306 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3225 on 92 degrees of freedom
## Multiple R-squared: 0.04982, Adjusted R-squared: 0.03949
## F-statistic: 4.823 on 1 and 92 DF, p-value: 0.03059
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74032 -0.12022  0.08039  0.21808  0.48798
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.90022    0.03667  24.547  < 2e-16 ***
## animal_incidence  0.96019    0.36388   2.639  0.00979 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3121 on 91 degrees of freedom
## Multiple R-squared: 0.07108, Adjusted R-squared: 0.06087
## F-statistic: 6.963 on 1 and 91 DF, p-value: 0.0097893
```

```
diff_full <- lag_models_full(df_cum_trend_diff)
```

```
## Series: human_incidence
```

```
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39191 -0.06067  0.00674  0.07284  0.22568
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.008767   0.012442   0.705   0.4828
## animal_incidence 0.276970   0.109255   2.535   0.0129 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1212 on 93 degrees of freedom
## Multiple R-squared: 0.06464, Adjusted R-squared: 0.05458
## F-statistic: 6.427 on 1 and 93 DF, p-value: 0.012912
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.380512 -0.061612  0.004388  0.082896  0.227038
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.009462   0.012779    0.74   0.4609
## animal_incidence -0.203928   0.113919   -1.79   0.0767 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1239 on 92 degrees of freedom
## Multiple R-squared: 0.03366, Adjusted R-squared: 0.02316
## F-statistic: 3.205 on 1 and 92 DF, p-value: 0.076725
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41581 -0.06626  0.01000  0.08148  0.22724
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.009263   0.013126   0.706    0.482
## animal_incidence -0.050532   0.116387  -0.434    0.665
##
## Residual standard error: 0.1266 on 91 degrees of freedom
## Multiple R-squared: 0.002067,    Adjusted R-squared: -0.008899
## F-statistic: 0.1885 on 1 and 91 DF, p-value: 0.66519
## Series: human_incidence
## Model: TSLM
## Transformation: (human_incidence)
```

```
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.416871 -0.065766  0.005488  0.080884  0.231750
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.009651   0.013079   0.738   0.4625
## animal_incidence 0.191912   0.115341   1.664   0.0996 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1254 on 90 degrees of freedom
## Multiple R-squared: 0.02984, Adjusted R-squared: 0.01906
## F-statistic: 2.768 on 1 and 90 DF, p-value: 0.099615
```

```
write.csv(non_diff_full_with_NA |>
  select(-r_squared, -AIC, -adj_r_squared), "non_diff_full_with_NA.csv")

write.csv(non_diff_full_without_NA |>
  select(-r_squared, -AIC, -adj_r_squared), "non_diff_full_without_NA.csv")

# write_csv(non_diff_full, "non_diff_full.csv")
# write_csv(diff_full, "diff_full.csv")

## Getting the AIC, R-Squared and Adjusted R squared for each lag

Table_lag_indivi_withoutNA <- non_diff_indivi_without_NA |>
  select(Lag = lag, `R-Squared(%)` = r_squared, `Adjusted R-Squared(%)` = adj_r_squared, AIC) |>
  mutate(across(c(`R-Squared(%)` , `Adjusted R-Squared(%)`, ), ~ (. * 100))) |>
    mutate(across(where(is.numeric), ~round(., 2))) |>
  unique() |>
  arrange(desc(`Adjusted R-Squared(%)`)) |>
  knitr::kable(
    align = "l",
    caption = "The AIC, R-Squared and Adjusted R-Squared for each lag for individual species. The data a
    format = "pipe",
    latex_options = "hold_position"
  )
Table_lag_indivi_withoutNA
```

Table 3: The AIC, R-Squared and Adjusted R-Squared for each lag for individual species. The data as been arranged in decreasing order of Adjusted R-Squared

Lag	R-Squared(%)	Adjusted R-Squared(%)	AIC
1	3.97	-1.5	-101.87
0	1.35	-4.1	-93.28
2	0.11	-5.8	-115.92
3	0.08	-6.0	-122.86

```
Table_lag_full <- non_diff_full_without_NA |>
  select(Lag = lag, `R-Squared(%)` = r_squared, `Adjusted R-Squared(%)` = adj_r_squared, AIC) |>
  mutate(across(c(`R-Squared(%)` , `Adjusted R-Squared(%)`, ), ~ (. * 100))) |>
    mutate(across(where(is.numeric), ~round(., 2))) |>
  unique() |>
  arrange(desc(`Adjusted R-Squared(%)`)) |>
    knitr::kable(
    align = "l",
    caption = "The AIC, R-Squared and Adjusted R-Squared for each lag for combined species. The data as
    format = "pipe",
    latex_options = "hold_position"
  )
Table_lag_full
```

Table 4: The AIC, R-Squared and Adjusted R-Squared for each lag for combined species. The data as been arranged in decreasing order of Adjusted R-Squared

Lag	R-Squared(%)	Adjusted R-Squared(%)	AIC
0	7.59	6.6	-208.08
3	7.11	6.1	-212.59
2	4.98	3.9	-208.79
1	4.14	3.1	-206.16

```
# Lag 2
df_2_2 <- df_1 |>
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ dplyr::lag(., n = 2))) |>
  na.omit() |>
  mutate(date = as.Date(date))

# Lag 3
df_2_3 <- df_1 |>
  as_tibble() %>%
  mutate_at(vars(catt_incidence, cam_incidence, goat_incidence, shp_incidence),
            list( ~ dplyr::lag(., n = 3))) |>
  na.omit() |>
  mutate(date = as.Date(date))


fit_county_model <- function(county_name, data, type) {
  # Subset the data for the specific county
  county_data <- filter(data, county == county_name)

  if (type == "full")
  {
    # Check if all incidences are zero
    if (all(county_data$animal_incidence == 0)) {
      message(paste("Skipping model for", county_name, "as all incidences are zero."))
      return(NULL)
    }
```

```
  # Fit the model
  mod_county <- county_data |>
    as_tsibble() |>
    model(
      TSLM(
        human_incidence ~ animal_incidence
      )
    ) |>
    tidy() |>
    select(-.model) |>
    as_tibble() |>
    mutate(term = case_when(
      term == "animal_incidence" ~ "Animal Incidence",
      TRUE ~ as.character(term)
    ),
    variable = term
    ) |>
    select(6, 2:5) |>
    group_by(variable) %>%
    mutate(
      conf_low = min(estimate - std.error * 1.645),
      conf_high = max(estimate + std.error * 1.645)
    )
}
else if(type == "individual") {
# Check if all incidences are zero
if (all(county_data$catt_incidence == 0 &
        county_data$cam_incidence == 0 &
        county_data$goat_incidence == 0 &
        county_data$shp_incidence == 0)) {
  message(paste("Skipping model for", county_name, "as all incidences are zero."))
  return(NULL)
}

# Fit the model
mod_county <- county_data |>
  as_tsibble() |>
  model(
    TSLM(
      human_incidence ~ catt_incidence + goat_incidence + shp_incidence +cam_incidence
    )
  ) |>
  tidy() |>
  select(-.model) |>
  as_tibble() |>
  mutate(term = case_when(
    term == "goat_incidence" ~ "Goat Incidence",
    term == "catt_incidence" ~ "Cattle incidence",
    term == "shp_incidence" ~ "Sheep incidence",
    term == "cam_incidence" ~ "Camel incidence",
    TRUE ~ as.character(term)
  ),
  variable = term
```

```
    ) |>
    select(6, 2:5) |>
    group_by(variable) %>%
    mutate(
      conf_low = min(estimate - std.error * 1.645),
      conf_high = max(estimate + std.error * 1.645)
    )
  }
  return(mod_county)
}

county_names <- unique(df_2_2$county)

# Create a list to store the models for each county
models_list <- list()

# Initialize the data frame for each county
coefficients_df_lag2 <- data.frame(county = character(),
                                   variable = character(),
                                   estimate = numeric(),
                                   stringsAsFactors = FALSE)

# At lag 2
for (county_name in county_names) {
  message(paste("Fitting model for", county_name))



  # Fit the model
  mod_county <- fit_county_model(county_name, df_2_2, type = "individual")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df_lag2 <- bind_rows(coefficients_df_lag2,
                                      mod_county %>%
                                        mutate(county = county_name,
                                               estimate = round(estimate, 3)))
  }
}

coefficients_df_lag2 <- coefficients_df_lag2 |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
  mutate(across(c(3:8), ~round(., 3)))
write_csv(coefficients_df_lag2, "individual_animal_incidence_per_county_lag2.csv")

# At lag 3
coefficients_df_lag3 <- data.frame(county = character(),
                                   variable = character(),
                                   estimate = numeric(),
                                   stringsAsFactors = FALSE)

for (county_name in county_names) {
```

```
  message(paste("Fitting model for", county_name))



  # Fit the model
  mod_county <- fit_county_model(county_name, df_2_3, type = "individual")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df_lag3 <- bind_rows(coefficients_df_lag3,
                                      mod_county %>%
                                        mutate(county = county_name,
                                               estimate = round(estimate, 3)))
  }
}

coefficients_df_lag3 <- coefficients_df_lag3 |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
  mutate(across(c(3:8), ~round(., 3)))
write_csv(coefficients_df_lag3, "individual_animal_incidence_per_county_lag3.csv")
```

```
# Lag 2
df_cum_2_2 <- df_cum |>
  as_tibble() %>%
  mutate_at(vars(animal_incidence),
            list( ~ dplyr::lag(., n = 2))) |>
  na.omit() |>
  mutate(date = as.Date(date))

# Lag 3
df_cum_2_3 <- df_cum |>
  as_tibble() %>%
  mutate_at(vars(animal_incidence),
            list( ~ dplyr::lag(., n = 3))) |>
  na.omit() |>
  mutate(date = as.Date(date))

# At lag 2
county_names2_2 <- unique(df_cum_2_2$county)

# Create a list to store the models for each county
models_list <- list()

# Initialize the data frame for each county
coefficients_df2_lag2 <- data.frame(county = character(),
                                    variable = character(),
                                    estimate = numeric(),
                                    stringsAsFactors = FALSE)

for (county_name in county_names2_2) {
  message(paste("Fitting model for", county_name))
```

```
  # Fit the model
  mod_county <- fit_county_model(county_name, df_cum_2_2, type = "full")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df2_lag2 <- bind_rows(coefficients_df2_lag2,
                                mod_county %>%
                                  mutate(county = county_name,
                                         estimate = round(estimate, 3)))
  }
}

coefficients_df2_lag2 <- coefficients_df2_lag2 |>
  mutate(across(c(3:8), ~round(., 3))) |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
  as_tibble()
write_csv(coefficients_df2_lag2, "all_animal_incidence_per_county_lag2.csv")

# At lag 3
county_names2_3 <- unique(df_cum_2_3$county)

# Create a list to store the models for each county
models_list <- list()

# Initialize the data frame for each county
coefficients_df2_lag3 <- data.frame(county = character(),
                                    variable = character(),
                                    estimate = numeric(),
                                    stringsAsFactors = FALSE)

for (county_name in county_names2_3) {
  message(paste("Fitting model for", county_name))



  # Fit the model
  mod_county <- fit_county_model(county_name, df_cum_2_3, type = "full")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df2_lag3 <- bind_rows(coefficients_df2_lag3,
                                    mod_county %>%
                                      mutate(county = county_name,
                                             estimate = round(estimate, 3)))
  }
}

coefficients_df2_lag3 <- coefficients_df2_lag3 |>
  mutate(across(c(3:8), ~round(., 3))) |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
```

```
  as_tibble()
write_csv(coefficients_df2_lag3, "all_animal_incidence_per_county_lag3.csv")
```

```
# Lag 2
df_2_2_complete <- df_1_complete |>
  as_tibble() %>%
  mutate_at(
    vars(catt_incidence, goat_incidence),
    list(~ dplyr::lag(., n = 2))
  ) |>
  na.omit() |>
  mutate(date = as.Date(date))

# Lag 3
df_2_3_complete  <- df_1_complete  |>
  as_tibble() %>%
  mutate_at(
    vars(catt_incidence, goat_incidence),
    list(~ dplyr::lag(., n = 4))
  ) |>
  na.omit() |>
  mutate(date = as.Date(date))


fit_county_model_complete <- function(county_name, data, type) {
  # Subset the data for the specific county
  county_data <- filter(data, county == county_name)

  if (type == "full")
  {
    # Check if all incidences are zero
    if (all(county_data$animal_incidence == 0)) {
      message(paste("Skipping model for", county_name, "as all incidences are zero."))
      return(NULL)
    }

    # Fit the model
    mod_county <- county_data |>
      as_tsibble() |>
      model(
        TSLM(
          human_incidence ~ animal_incidence
        )
      ) |>
      tidy() |>
      select(-.model) |>
      as_tibble() |>
      mutate(term = case_when(
        term == "animal_incidence" ~ "Animal Incidence",
        TRUE ~ as.character(term)
      ),
      variable = term
      ) |>
```

```
      select(6, 2:5) |>
      group_by(variable) %>%
      mutate(
        conf_low = min(estimate - std.error * 1.645),
        conf_high = max(estimate + std.error * 1.645)
      )
  }
  else if (type == "individual") {
  # Check if all incidences are zero
  if (all(county_data$catt_incidence == 0 &
          county_data$goat_incidence == 0 )) {
    message(paste("Skipping model for", county_name, "as all incidences are zero."))
    return(NULL)
  }

  # Fit the model
  mod_county <- county_data |>
    as_tsibble() |>
    model(
      TSLM(
        human_incidence ~ catt_incidence + goat_incidence
      )
    ) |>
    tidy() |>
    select(-.model) |>
    as_tibble() |>
    mutate(term = case_when(
      term == "goat_incidence" ~ "Goat Incidence",
      term == "catt_incidence" ~ "Cattle incidence",
      TRUE ~ as.character(term)
    ),
    variable = term
    ) |>
    select(6, 2:5) |>
    group_by(variable) %>%
    mutate(
      conf_low = min(estimate - std.error * 1.645),
      conf_high = max(estimate + std.error * 1.645)
    )
  }
  return(mod_county)
}

county_names <- unique(df_2_2_complete$county)

# Create a list to store the models for each county
models_list_withouNA <- list()

# Initialize the data frame for each county
coefficients_df_lag2_withoutNA <- data.frame(county = character(),
                                variable = character(),
                                estimate = numeric(),
                                stringsAsFactors = FALSE)
```

```r
# At lag 2
for (county_name in county_names) {
  message(paste("Fitting model for", county_name))



  # Fit the model
  mod_county <- fit_county_model_complete(county_name, df_2_2_complete, type = "individual")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df_lag2_withoutNA <- bind_rows(coefficients_df_lag2_withoutNA,
                                mod_county %>%
                                  mutate(county = county_name,
                                         estimate = round(estimate, 3)))
  }
}

coefficients_df_lag2_withoutNA <- coefficients_df_lag2_withoutNA |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
  mutate(across(c(3:8), ~round(., 3)))
write_csv(coefficients_df_lag2_withoutNA, "individual_animal_incidence_per_county_lag2_withoutNA.csv")

# At lag 3
coefficients_df_lag3_withoutNA <- data.frame(county = character(),
                                  variable = character(),
                                  estimate = numeric(),
                                  stringsAsFactors = FALSE)

for (county_name in county_names) {
  message(paste("Fitting model for", county_name))



  # Fit the model
  mod_county <- fit_county_model_complete(county_name, df_2_3_complete, type = "individual")

  # Check if model fitting was successful
  if (!is.null(mod_county)) {
    # Extract coefficients, round off, and add to the data frame
    coefficients_df_lag3_withoutNA <- bind_rows(coefficients_df_lag3_withoutNA,
                                mod_county %>%
                                  mutate(county = county_name,
                                         estimate = round(estimate, 3)))
  }
}

coefficients_df_lag3_withoutNA <- coefficients_df_lag3_withoutNA |>
  mutate(significant = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant")) |>
  mutate(across(c(3:8), ~round(., 3)))
write_csv(coefficients_df_lag3_withoutNA, "individual_animal_incidence_per_county_lag4_withoutNA.csv")
```

Note: All counties model (combined without NA-complete case) at lag 2 and 3 was omitted as it had no difference to combined with NA

Mixed Effect Models

The section includes the mixed effect modelling. The mixed effect was done for both the incidence (as integers), cases and the proportion, and at lags 0 to 3.

```
# Importing packages
if (require(pacman))
{
  p_load(
    tidyverse,
    tseries,
    data.table,
    scales,
    zoo,
    forecast,
    sf,
    patchwork,
    grid,
    fable,
    patchwork,
    xts,
    feasts,
    cowplot,
    broom,
    kableExtra,
    readxl,
    stringi,
    stringr,
    rKenyaCensus,
    knitr,
    purrr,
    RColorBrewer,
    tscount,
    lme4
  )
}

# Importing data
individual <- fread("individual_incidence_cases.csv")
combined <- fread("combined_incidence_cases.csv")
individual_county <- fread("df_tot_cases_spatial_month.csv")
combined_county <- fread("df_spatial_cum_month.csv")


# Complete
county_indivi_complete <- fread("df_1_complete.csv") |>
  mutate(hum_cases = round(hum_cases))

county_comb_complete <- fread("df_cum_complete.csv") |>
```

```
  mutate(hum_cases = round(hum_cases))

county_humanpop <- fread('county_humanpop.csv')
```

```
# Individual: Incidence
# The cattle and goat incidence were calculated 10M per population
df_indivi_inci <- county_indivi_complete |>
  mutate(across(c(catt_incidence, goat_incidence), ~ round(.*10))) |>
  mutate(human_incidence = round(human_incidence * 10)) |>
  select(date, county, contains("incidence")) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))

# Individual: Proportion
df_indivi_inci_prop <- county_indivi_complete |>
  mutate(across(c(catt_incidence, goat_incidence), ~ .*10)) |>
  mutate(human_incidence = human_incidence/1e3) |>
  select(date, county, contains("incidence")) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .))) |>
  merge(county_humanpop, by = c("county", "date"))

# Individual: Cases
df_indivi_cases <- county_indivi_complete |>
  select(date, county, contains("cases")) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))


# Combined : Incidence
df_comb_inci <- county_comb_complete |>
  mutate(human_incidence = round(human_incidence*10),
         animal_incidence = round(animal_incidence*10)
         ) |>
  select(date, county, animal_incidence, human_incidence) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))

# combined: Proportion
df_comb_inci_prop <- county_comb_complete |>
  mutate(human_incidence = human_incidence/1e3,
         animal_incidence = animal_incidence*10
         ) |>
  select(date, county, animal_incidence, human_incidence) |>
  mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))|>
  merge(county_humanpop, by = c("county", "date"))

# Combined: Cases
df_comb_cases <- county_comb_complete |>
  select(date, county, animal_cases, hum_cases)
```

```
# Individual Incidence
indivi_models <- function(df, type, max_lag, ...) {
  # Type is either prop or incidence
  result_df <- tibble()
  if (type == "prop") {
    for (lag_value in 0:max_lag) {
```

```r
df_lagged <- df |>
  as_tibble() %>%
  mutate_at(
    vars(
      catt_incidence,
      goat_incidence
    ),
    list(~ lag(., n = lag_value))
  ) |>
  na.omit() |>
  mutate(date = as.Date(date))

mod <-
  glmer(
    human_incidence ~ catt_incidence + goat_incidence +
      (1 |
         county),
    data = df_lagged,
    family = binomial(link = "logit"),
    nAGQ = 0,
    weights = pop
  )

mod_results <- broom.mixed::tidy(mod) %>%
  as_tibble() %>%
  mutate(
    term = case_when(
      term == "goat_incidence" ~ "Goat Incidence",
      term == "catt_incidence" ~ "Cattle incidence",
      term == "sd__(Intercept)" ~ "Sd - Random Intercept",
      TRUE ~ as.character(term)
    ),
    effect = case_when(
      effect == "ran_pars" ~ "Random",
      TRUE ~ effect
    ),
    variable = term
  ) |>
  rename(`log odds` = estimate) |>
  mutate(odds = exp(`log odds`),
         odds = ifelse(effect == "Random", NA, odds)
         ) |>
  select(1, 8,4, 9, 5:8) |>
  group_by(variable) %>%
  mutate(
    conf_low = min(`log odds` - std.error * 1.645),
    conf_high = max(`log odds` + std.error * 1.645)
  ) %>%
  mutate(lag = lag_value)

metrics <- broom.mixed::glance(mod) %>%
  select(nobs, AIC, BIC)
```

```
    mod_results <- bind_cols(mod_results, metrics) |>
      mutate(across(
        c(
          `log odds`,
          odds,
          std.error,
          statistic,
          p.value,
          conf_low,
          conf_high,
          AIC,
          BIC
        ),
        ~ round(., 3)
      )) |>
      mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant"))
    cat(paste("Runnning model for lag", lag_value), "\n")
    result_df <- bind_rows(result_df, mod_results)
  }

} else if (type == "inci") {
  for (lag_value in 0:max_lag) {
    df_lagged <- df |>
      as_tibble() %>%
      mutate_at(
        vars(
          catt_incidence,
          goat_incidence
        ),
        list(~ lag(., n = lag_value))
      ) |>
      na.omit() |>
      mutate(date = as.Date(date))

    mod <-
       glmer.nb(
        human_incidence ~ catt_incidence + goat_incidence +
          (1 |
             county),
        data = df_lagged,
        nAGQ = 0
      )

    mod_results <- broom.mixed::tidy(mod) %>%
      as_tibble() %>%
      mutate(
        term = case_when(
          term == "goat_incidence" ~ "Goat Incidence",
          term == "catt_incidence" ~ "Cattle incidence",
          term == "sd__(Intercept)" ~ "Sd - Random Intercept",
          TRUE ~ as.character(term)
        ),
        effect = case_when(
```

```
            effect == "ran_pars" ~ "Random",
            TRUE ~ effect
          ),
          variable = term
        ) |>
        rename(`log IRR` = estimate) |>
        mutate(IRR = exp(`log IRR`),
               IRR = ifelse(effect == "Random", NA, IRR)
               ) |>
        select(1, 8,4, 9, 5:8) |>
        group_by(variable) %>%
        mutate(
          conf_low = min(`log IRR` - std.error * 1.645),
          conf_high = max(`log IRR` + std.error * 1.645)
        ) %>%
        mutate(lag = lag_value)

      metrics <- broom.mixed::glance(mod) %>%
        select(nobs, AIC, BIC)

      mod_results <- bind_cols(mod_results, metrics) |>
        mutate(across(
          c(
            `log IRR`,
            IRR,
            std.error,
            statistic,
            p.value,
            conf_low,
            conf_high,
            AIC,
            BIC
          ),
          ~ round(., 4)
        )) |>
        mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant"))
      cat(paste("Runnning model for lag", lag_value), "\n")
      result_df <- bind_rows(result_df, mod_results)
    }

  }

  return(result_df)

}

df_indivi_model_inci = indivi_models(df = df_indivi_inci, type = "inci", max_lag = 3)
```

```
## Runnning model for lag 0
## Runnning model for lag 1
## Runnning model for lag 2
## Runnning model for lag 3
```

```
df_indivi_model_prop = indivi_models(df = df_indivi_inci_prop, type = "prop", max_lag = 3)
```

```
## Runnning model for lag 0
## Runnning model for lag 1
## Runnning model for lag 2
## Runnning model for lag 3
```

```
write.csv(df_indivi_model_inci, "df_indivi_model_inci.csv")

# Combined Cases
comb_models <- function(df, type, max_lag, ...) {
  # Type is either prop or incidence
  result_df <- tibble()
  if (type == "prop") {
    for (lag_value in 0:max_lag) {
      df_lagged <- df |>
        as_tibble() %>%
        mutate_at(
          vars(
            animal_incidence,
          ),
          list(~ lag(., n = lag_value))
        ) |>
        na.omit() |>
        mutate(date = as.Date(date))

      mod <-
        glmer(
          human_incidence ~ animal_incidence +
            (1 |
               county),
          data = df_lagged,
          family = binomial(link = "logit"),
          nAGQ = 0,
          weights = pop
        )

      mod_results <- broom.mixed::tidy(mod) %>%
        as_tibble() %>%
        mutate(
          term = case_when(
            term == "animal_incidence" ~ "Animal incidence",
            term == "sd__(Intercept)" ~ "Sd - Random Intercept",
            TRUE ~ as.character(term)
          ),
          effect = case_when(
            effect == "ran_pars" ~ "Random",
            TRUE ~ effect
          ),
          variable = term
        ) |>
        rename(`log odds` = estimate) |>
        mutate(odds = exp(`log odds`),
```

```
              odds = ifelse(effect == "Random", NA, odds)
              ) |>
      select(1, 8,4, 9, 5:8) |>
      group_by(variable) %>%
      mutate(
        conf_low = min(`log odds` - std.error * 1.645),
        conf_high = max(`log odds` + std.error * 1.645)
      ) %>%
      mutate(lag = lag_value)

    metrics <- broom.mixed::glance(mod) %>%
      select(nobs, AIC, BIC)

    mod_results <- bind_cols(mod_results, metrics) |>
      mutate(across(
        c(
          `log odds`,
          odds,
          std.error,
          statistic,
          p.value,
          conf_low,
          conf_high,
          AIC,
          BIC
        ),
        ~ round(., 3)
      )) |>
      mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant"))
    cat(paste("Runnning model for lag", lag_value), "\n")
    result_df <- bind_rows(result_df, mod_results)
  }

} else if (type == "inci") {
  for (lag_value in 0:max_lag) {
    df_lagged <- df |>
      as_tibble() %>%
      mutate_at(
        vars(
       animal_incidence
        ),
        list(~ lag(., n = lag_value))
      ) |>
      na.omit() |>
      mutate(date = as.Date(date))

    mod <-
       glmer.nb(
        human_incidence ~ animal_incidence +
          (1 |
             county),
        data = df_lagged,
        nAGQ = 0
```

```
    )

  mod_results <- broom.mixed::tidy(mod) %>%
    as_tibble() %>%
    mutate(
      term = case_when(
        term == "animal_incidence" ~ "Animal incidence",
        term == "sd__(Intercept)" ~ "Sd - Random Intercept",
        TRUE ~ as.character(term)
      ),
      effect = case_when(
        effect == "ran_pars" ~ "Random",
        TRUE ~ effect
      ),
      variable = term
    ) |>
    rename(`log IRR` = estimate) |>
    mutate(IRR = exp(`log IRR`),
           IRR = ifelse(effect == "Random", NA, IRR)
           ) |>
    select(1, 8,4, 9, 5:8) |>
    group_by(variable) %>%
    mutate(
      conf_low = min(`log IRR` - std.error * 1.645),
      conf_high = max(`log IRR` + std.error * 1.645)
    ) %>%
    mutate(lag = lag_value)

  metrics <- broom.mixed::glance(mod) %>%
    select(nobs, AIC, BIC)

  mod_results <- bind_cols(mod_results, metrics) |>
    mutate(across(
      c(
        `log IRR`,
        IRR,
        std.error,
        statistic,
        p.value,
        conf_low,
        conf_high,
        AIC,
        BIC
      ),
      ~ round(., 4)
    )) |>
    mutate(significance = ifelse(conf_low * conf_high > 0, "Significant", "Not Significant"))
  cat(paste("Runnning model for lag", lag_value), "\n")
  result_df <- bind_rows(result_df, mod_results)
 }

}
```

```
  return(result_df)

}

df_combined_model_inci = comb_models(df = df_comb_inci, type = "inci", max_lag = 3)
```

```
## Runnning model for lag 0
## Runnning model for lag 1
## Runnning model for lag 2
## Runnning model for lag 3
```

```
df_combined_model_prop = comb_models(df = df_comb_inci_prop, type = "prop", max_lag = 3)
```

```
## Runnning model for lag 0
## Runnning model for lag 1
## Runnning model for lag 2
## Runnning model for lag 3
```

Time Series Analysis

The following section contains the codes for time series analysis. Data was imported as follows;

```
# Loading packages ------------------------------------------------------------------------

if (require(pacman)) {
  p_load(
    data.table,
    forecast,
    tseries,
    xts,
    tidyverse,
    ggalt,
    fable,
    tsibble,
    readxl,
    stringi,
    stringr
    )
}

# Importing data ------------------------------------------------------------------------

df <- fread("combined_incidence.csv")
df2022_animal <- read_excel("animal_brucellosis_2022.xlsx")
df2023_animal <- read_excel("animal_brucellosis_2023.xlsx")
human_22_23 <- fread('brucella_2022_2023.csv')


human_pop <- fread("kenya_pop_2020_2025.csv") |>
  filter(Year == "2022") |>
  select(pop)
```

```r
animal_pop <- read_csv('animal_pop_2019.csv') %>%
  pivot_wider(
    names_from = "species",
    values_from = "species_num"
  ) %>%
  setNames(c("county", paste0(colnames(.[,2:7]), "_pop") %>% str_to_lower())) |>
  summarise(across(where(is.numeric), ~sum(.))) |>
  select(-pigs_pop, -donkeys_pop)
```

The data was the cleaned as follows;

```r
# Cleaning 2022 and 2023 human brucella data ------------------------------------------------------------

## Human
df22_human <- human_22_23 |>
  select(periodname, dataname, Kenya) |>
  pivot_wider(id_cols = periodname, names_from = dataname, values_from = Kenya) |>
  mutate(date = dmy(paste("01", periodname, sep = "-")) |> ymd()) |>
  rowwise() |>
  mutate(hum_cases = sum(Brucellosis, `MOH 706_Brucella`)) |>
  select(date, hum_cases) |>
  cbind(human_pop) |>
  mutate(human_incidence = hum_cases/pop * 1e3) |>
  select(-hum_cases, -pop)

# Animals
df22 <- df2022_animal |>
  select(month, year...13, `Nature of Diagnosis`, `Number Sick`, `Species Affected`) |>
  mutate(month =
           ifelse(
             str_detect(month, "[:digit:]"),
             month.name[as.numeric(month[str_detect(month, "[:digit:]")])],
             month
           )) |>
  filter(!is.na(month)) |>
  mutate(date = ymd(paste(year...13, month, "01", sep = "-"))) |>
  group_by(date, `Species Affected`) |>
  summarise(count = sum(`Number Sick`, na.rm = T)) |>
  mutate(`Species Affected` = ifelse(
    str_detect(`Species Affected`, "caprine|Caprine|CAPRINE"),
    "Goats",
    `Species Affected`
  ),
  count = as.numeric(count)

  ) |>
  filter(`Species Affected` %in% c("Cattle", "Goats", "Camel", "Sheep")) |>
  ungroup() |>
  distinct() |>
  pivot_wider(
    id_cols = "date",
```

```
    values_from = count,
    names_from = `Species Affected`,
    values_fn = sum
  ) |>
  rowwise() |>
  mutate(animal_cases = sum(Cattle, Goats, Sheep, Camel, na.rm = T)) |>
  cbind(animal_pop) |>
  mutate(
    animal_pop = sum(cattle_pop, goats_pop, sheep_pop, camels_pop, na.rm = T),
    animal_incidence = animal_cases/animal_pop*1e6,
    date = ymd((date))
    )


# Cleaning df2023 data ---------------------------------------------------------

# Animals
df23 <- df2023_animal |>
  select(month = Month, year...13 = Year, `Nature of Diagnosis`, `Number Sick`, `Species Affected`) |>
  mutate(month =
           ifelse(
             str_detect(month, "[:digit:]"),
             month.name[as.numeric(month[str_detect(month, "[:digit:]")])],
             month
           )) |>
  filter(!is.na(month)) |>
  mutate(date = ymd(paste(year...13, month, "01", sep = "-"))) |>
  group_by(date, `Species Affected`) |>
  summarise(count = sum(`Number Sick`, na.rm = T)) |>
  mutate(`Species Affected` = ifelse(
    str_detect(`Species Affected`, "caprine|Caprine|CAPRINE"),
    "Goats",
    `Species Affected`
  ),
  count = as.numeric(count)

  ) |>
  filter(`Species Affected` %in% c("Cattle", "Goats", "Camel", "Sheep")) |>
  ungroup() |>
  distinct() |>
  pivot_wider(
    id_cols = "date",
    values_from = count,
    names_from = `Species Affected`,
    values_fn = sum
  ) |>
  rowwise() |>
  mutate(animal_cases = sum(Cattle, Goats, Sheep, Camel, na.rm = T)) |>
  cbind(animal_pop) |>
  mutate(
    animal_pop = sum(cattle_pop, goats_pop, sheep_pop, camels_pop, na.rm = T),
    animal_incidence = animal_cases/animal_pop*1e6,
    date = ymd((date))
```

```
  )

all_y <- df22 |>
  rbind(df23) |>
  merge(df22_human, by = "date") |>
  select(date, human_incidence, animal_incidence)



df_complete1 <- df |>
  select(-V1) |>
  mutate(date = ymd(date)) |>
  rbind(all_y) |>
  arrange(date) |>
  mutate(human_incidence = ifelse(human_incidence > 2, median(human_incidence), human_incidence))
```

Then we test for stationarity as follows;

```
# Testing for stationary ------------------------------------------------------------------------
# Test for human incidence
adf.test(df_complete1$human_incidence) # Not stationary
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  df_complete1$human_incidence
## Dickey-Fuller = -1.8568, Lag order = 4, p-value = 0.6361
## alternative hypothesis: stationary
```

```
# Test for animal incidence
adf.test(df_complete1$animal_incidence)
```

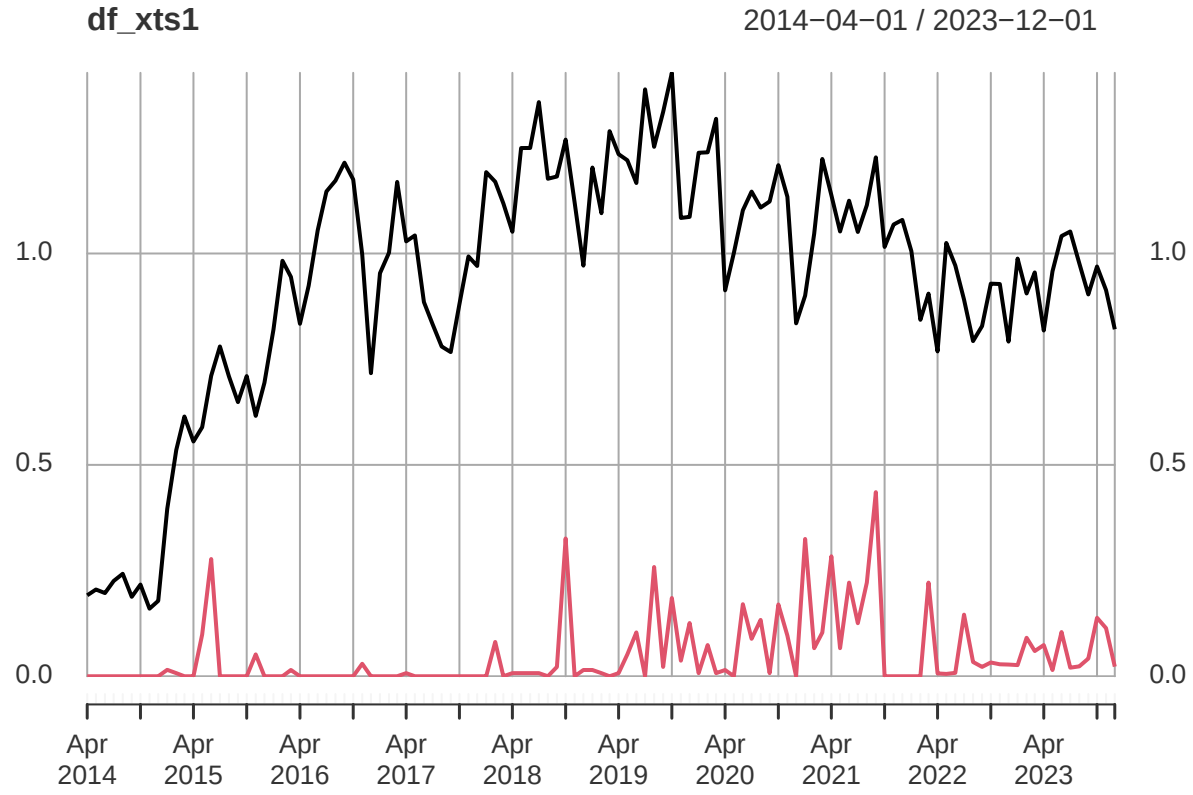
```
##
##  Augmented Dickey-Fuller Test
##
## data:  df_complete1$animal_incidence
## Dickey-Fuller = -3.6005, Lag order = 4, p-value = 0.03621
## alternative hypothesis: stationary
```

Lag 3 had the highest correlation so we lagged the time series at lag 3.

```
# Lagging ---------------------------------------------------------------

df2.1 <- df_complete1 |>
  mutate(date = as.Date(date),
         animal_incidence = lag(animal_incidence, 3)
         ) |>
  na.omit()
adf.test(df2.1$human_incidence)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  df2.1$human_incidence
## Dickey-Fuller = -2.1678, Lag order = 4, p-value = 0.5069
## alternative hypothesis: stationary
```

```
df_xts1 <- as.xts(x = cbind(df2.1$human_incidence, df2.1$animal_incidence), order.by = df2.1$date) |>
  setNames(c("human_incidence", "animal_incidence"))

plot(df_xts1)
```

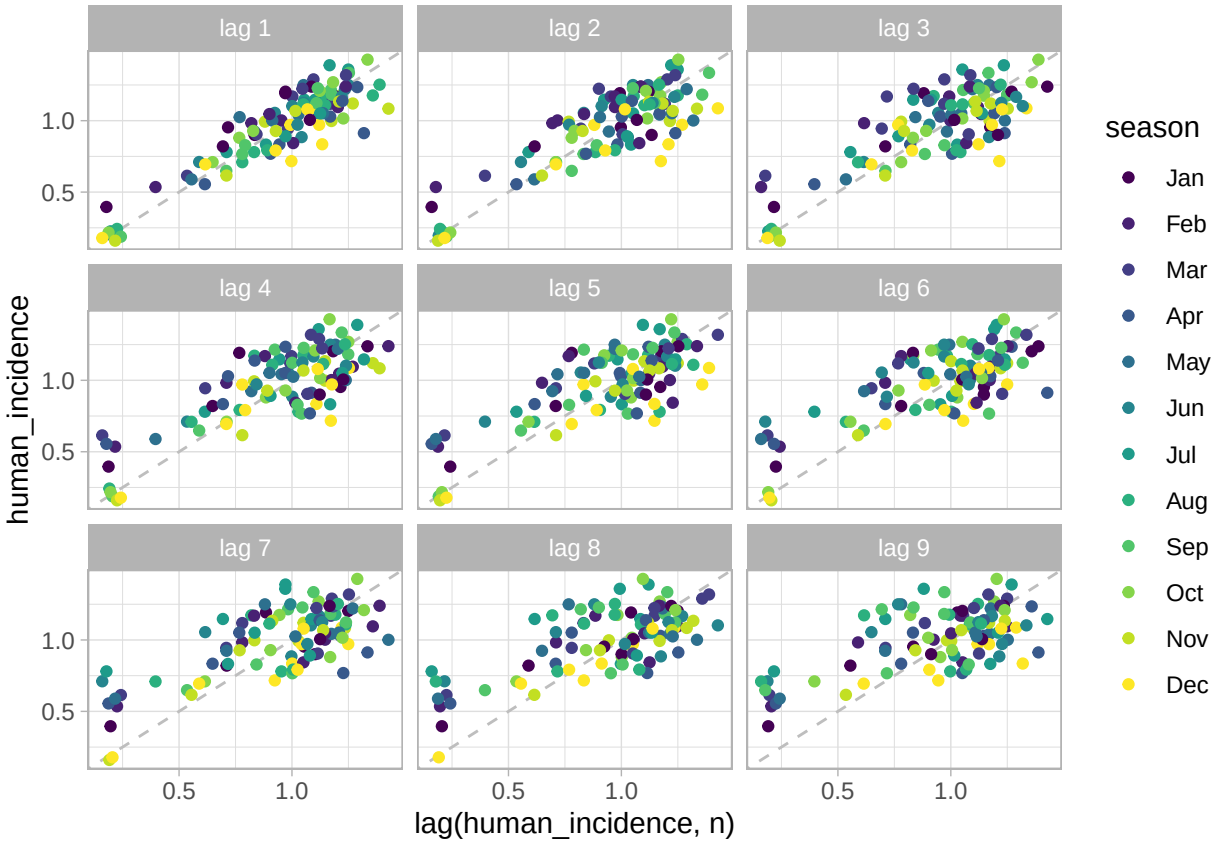



```
# Data for train and testing ----------------------------------------------------------------

df_xts <- window(df_xts1, start ='2014-04-01', end = '2022-12-01')
forecast.df <- window(df_xts1, start ='2023-01-01', end = '2023-12-01')

df2 <- df2.1 |>
  filter(date < as.Date('2023-01-01'))
# Our training and testing data reaches 2022-12-01
```

Testing for several aspects of the time series

1. Correlation of human incidence with it's past values

```
# Correlation between human incidence and the previous lags
dd = df_xts |>
  as.data.frame() %>%
  mutate(date = row.names(.) |>
           zoo::as.yearmon() |>
           yearmonth()) |>
  as_tsibble()

dd   |>
  gg_lag(human_incidence, geom = "point") +
  theme_light()
```



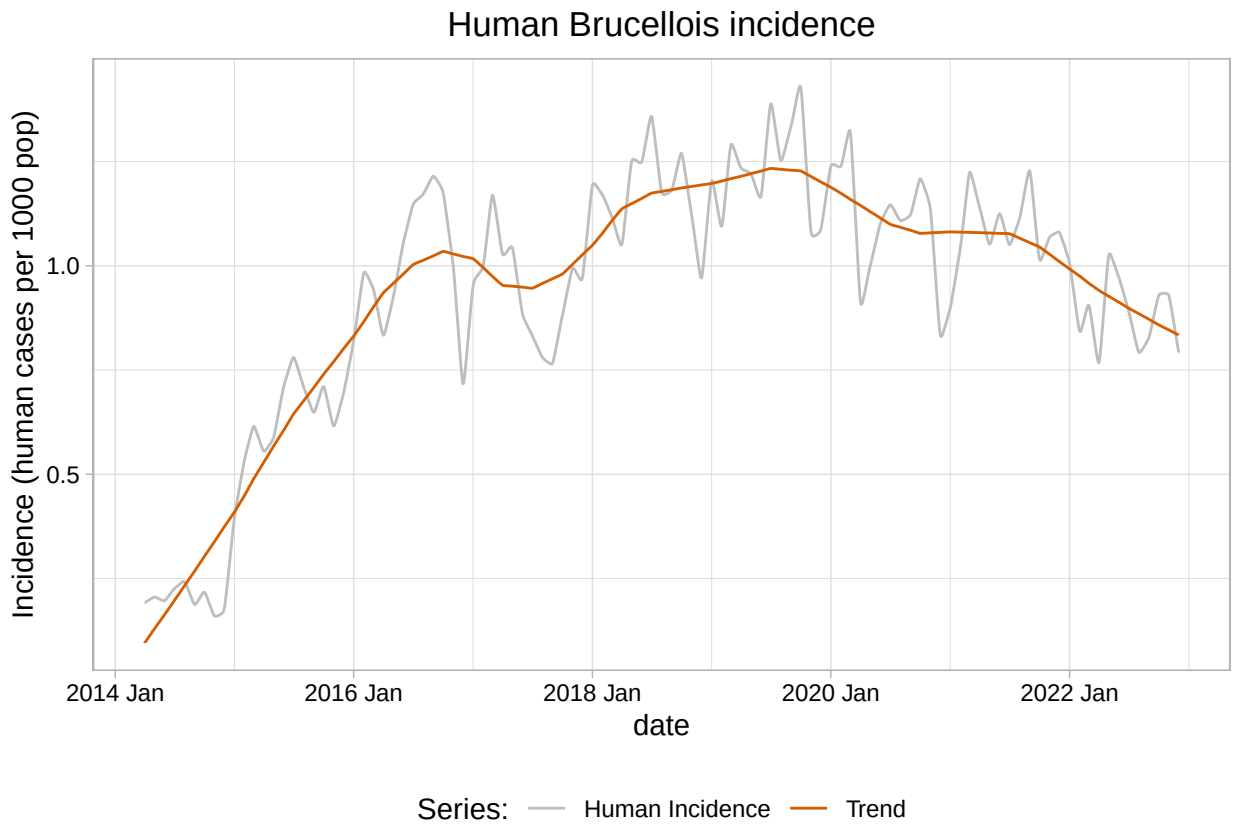

```
#> Lag 1 seems to have a strong correlation with the current value of human incidence
```

3. Decomposing the time series

```
dcmp <- dd |>
  model(stl = STL(human_incidence))
```

Below is the trend and the time series

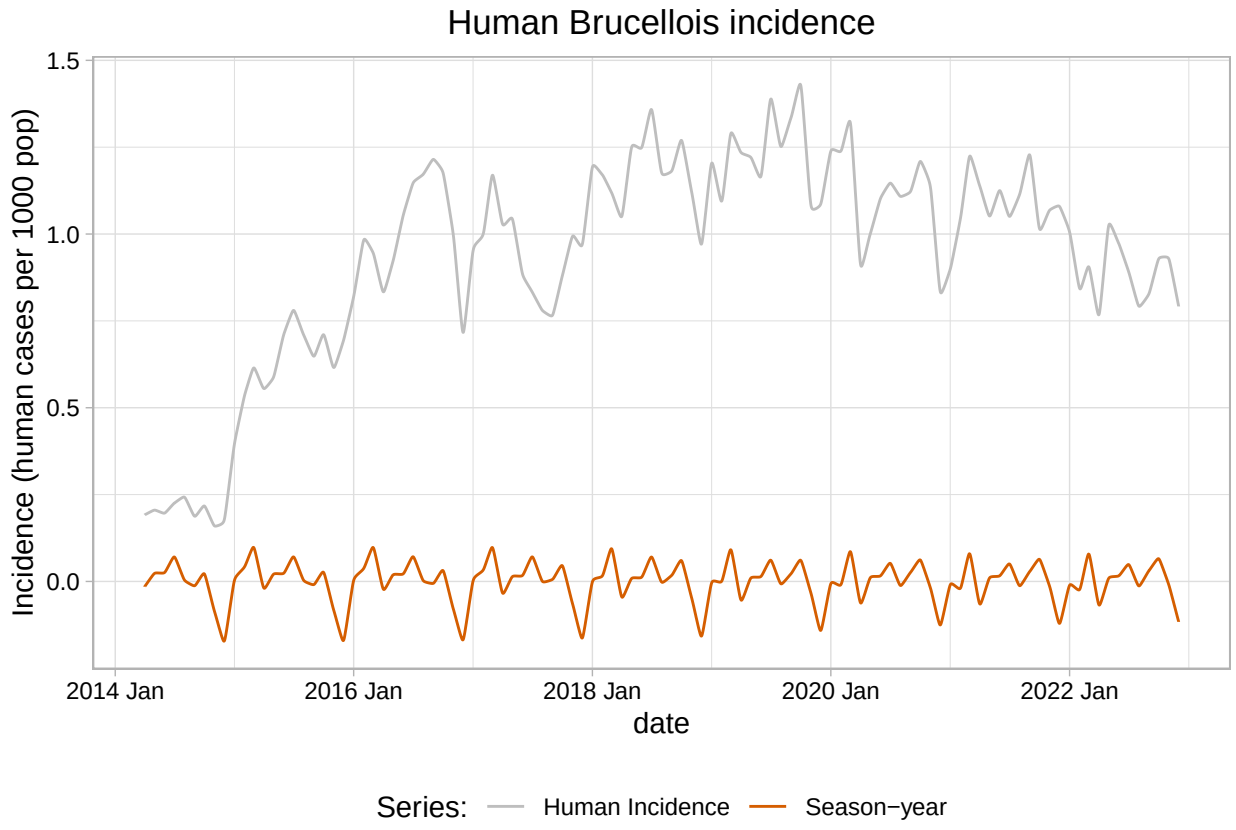
```
# The trend
components(dcmp) |>
  as_tsibble() |>
  ggplot(aes(x = date)) +
  geom_xspline(aes(y = human_incidence, colour = "Human Incidence")) +
  geom_xspline(aes(y = trend, color = 'Trend')) +
  labs(y = "Incidence (human cases per 1000 pop)",
       title = "Human Brucellois incidence") +
  theme_light() +
  theme(
    axis.title = element_text(color = 'black'),
    axis.text = element_text(color = 'black'),
    plot.title = element_text(color = 'black', hjust = .5),
    legend.position = 'bottom'
  ) +
  scale_color_manual(values = c("Human Incidence" = "gray", "Trend" = "#D55E00")) +
  labs(color = 'Series:')
```



The time series seasonality

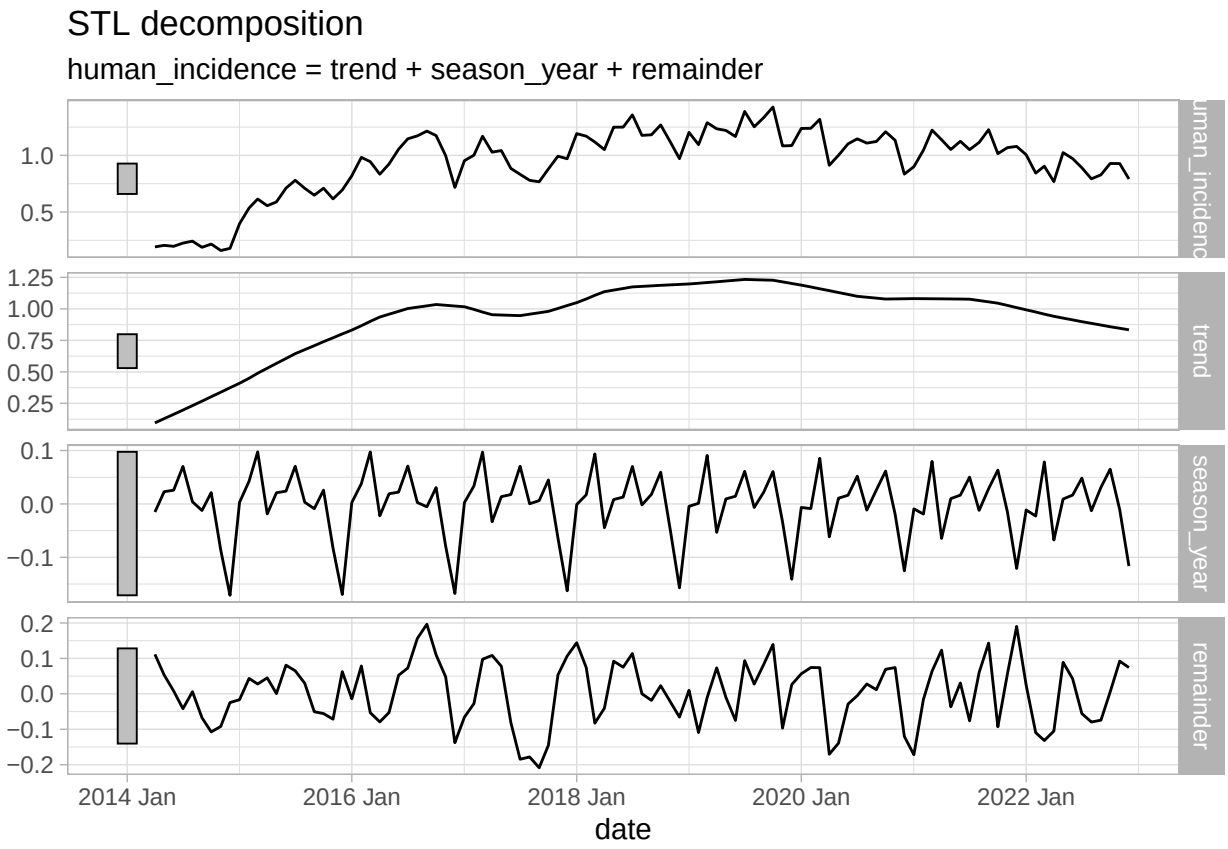
```
# Seasonality
components(dcmp) |>
  as_tsibble() |>
  ggplot(aes(x = date)) +
```

```
  geom_xspline(aes(y = human_incidence, colour = "Human Incidence")) +
  geom_xspline(aes(y = season_year, color = 'Season-year')) +
  labs(y = "Incidence (human cases per 1000 pop)",
       title = "Human Brucellois incidence") +
  theme_light() +
  theme(
    axis.title = element_text(color = 'black'),
    axis.text = element_text(color = 'black'),
    plot.title = element_text(color = 'black', hjust = .5),
    legend.position = 'bottom'
  ) +
  scale_color_manual(values = c("Human Incidence" = "gray", "Season-year" = "#D55E00")) +
  labs(color = 'Series:')
```



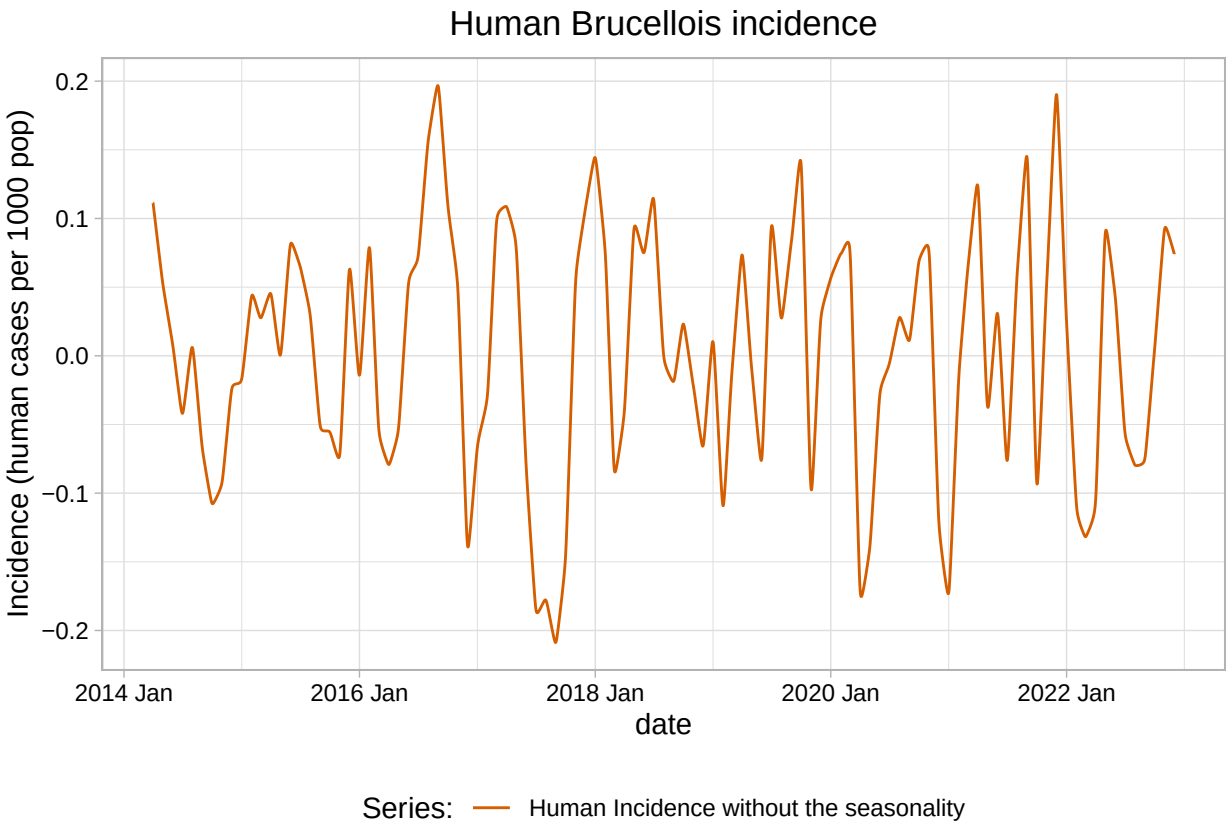
All of them together

```
# All of them together
components(dcmp) |>
  autoplot() +
  theme_light()
```



Plot without seasonality and trend

```
# The plot without seasonality and trend
components(dcmp) |>
  as_tsibble() |>
  ggplot(aes(x = date)) +
  #geom_xspline(aes(y = human_incidence, colour = "Human Incidence")) +
  geom_xspline(aes(y = season_adjust - trend, color = 'Human Incidence without the seasonality')) +
  labs(y = "Incidence (human cases per 1000 pop)",
       title = "Human Brucellois incidence") +
  theme_light() +
  theme(
    axis.title = element_text(color = 'black'),
    axis.text = element_text(color = 'black'),
    plot.title = element_text(color = 'black', hjust = .5),
    legend.position = 'bottom'
  ) +
  scale_color_manual(values = c("Human Incidence" = "gray", "Human Incidence without the seasonality" =
  labs(color = 'Series:')
```



Train and testing data

We split our data into training and testing (80%, 20%) respectively as follows:

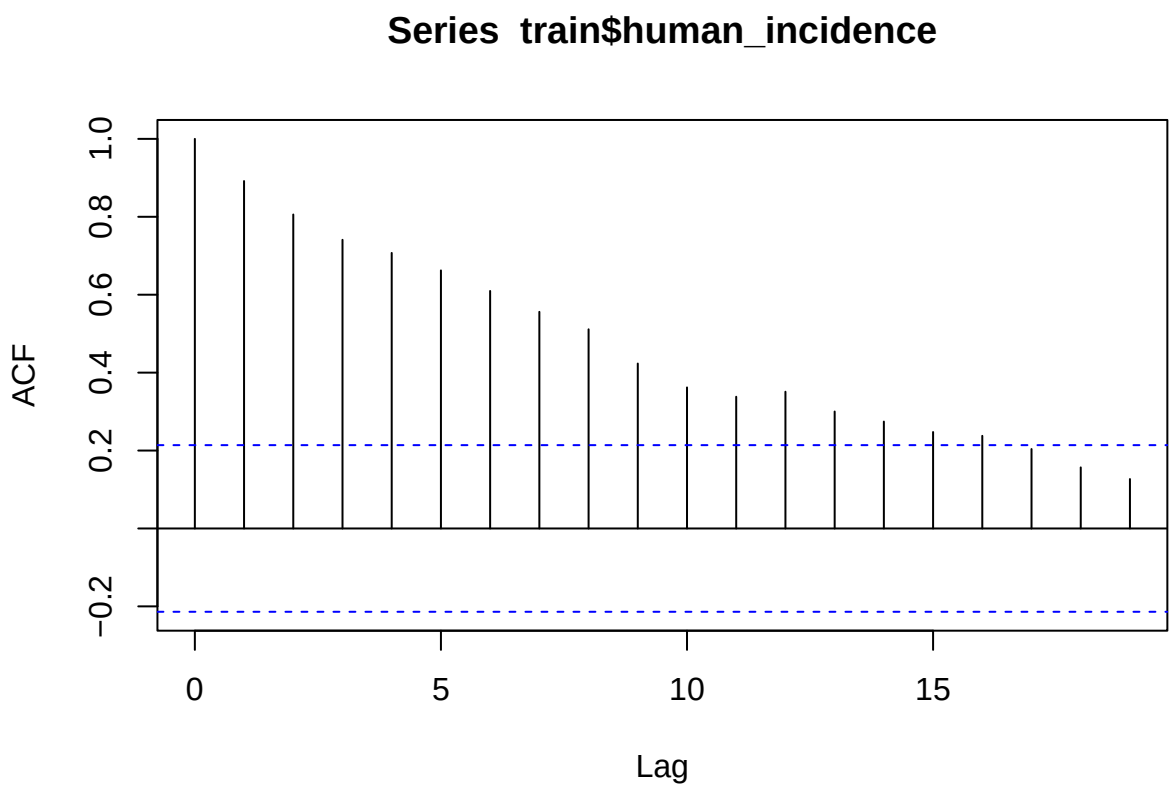
```
# Splitting ------------------------------------------------------------------------------

# train <- window(ts_diff, start = as.Date("2014-04-01"), end = "2021-12-01")
# test <- window(df_xts, start = as.Date("2022-01-01"), end = as.Date("2022-12-01"))

# We will take 70% for training, 15% for validation anf 15% for testing.
nrow(df_xts) # our data has 120 rows, thus 70% of this is 84 rows for training
```

```
## [1] 105
```

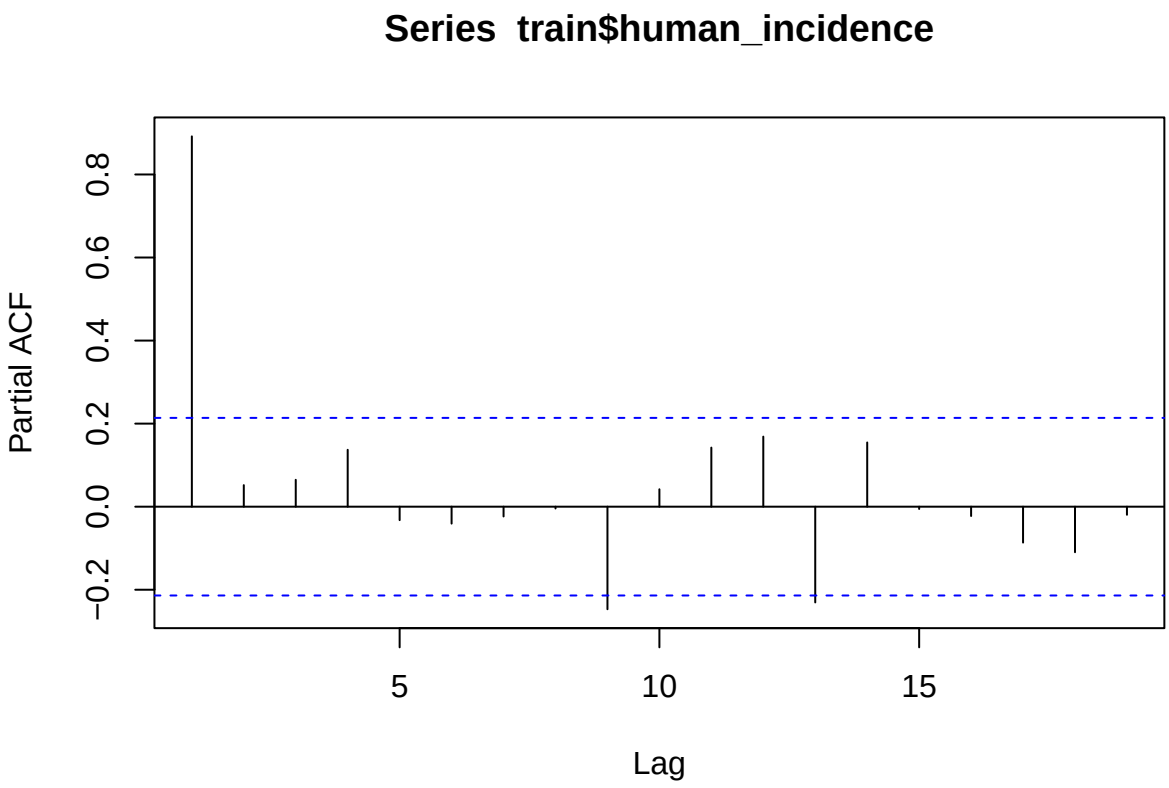
```
nrow_80 <- floor(.8 * nrow(df_xts))
train <- df_xts[1:nrow_80,]
test <- df_xts[(nrow_80 + 1):nrow(df_xts),]
```

The acf and pacf plot were as follows;

```r
# ACF and PACF plots ----------------------------------------------------------

acf(train$human_incidence)
```




```r
#> shows both trend and seasonality.
#> The slow decrease in the ACF as the lags increase is due to the trend, while the "scalloped" shape
#> is due to the seasonality. Time series that show no autocorrelation are called white noise.

pacf(train$human_incidence)
```



Training the Models

1. Model without the covariate

The model without covariate was trained as follows;

```
set.seed(123)
model_sarima <- train |>
  as.data.frame() %>%
  mutate(date = row.names(.) |>
           zoo::as.yearmon() |>
           yearmonth()) |>
  as_tsibble() |>
  model(
    ARIMA(human_incidence,
          ic = "aic",
          stepwise = T
          )
  ) |>
  report()
```

```
## Series: human_incidence
```

```
## Model: ARIMA(0,1,0)(2,0,0)[12]
##
## Coefficients:
##         sar1    sar2
##       0.2471  0.3418
## s.e.  0.1041  0.1158
##
## sigma^2 estimated as 0.01293:  log likelihood=61.28
## AIC=-116.56   AICc=-116.25   BIC=-109.3
```

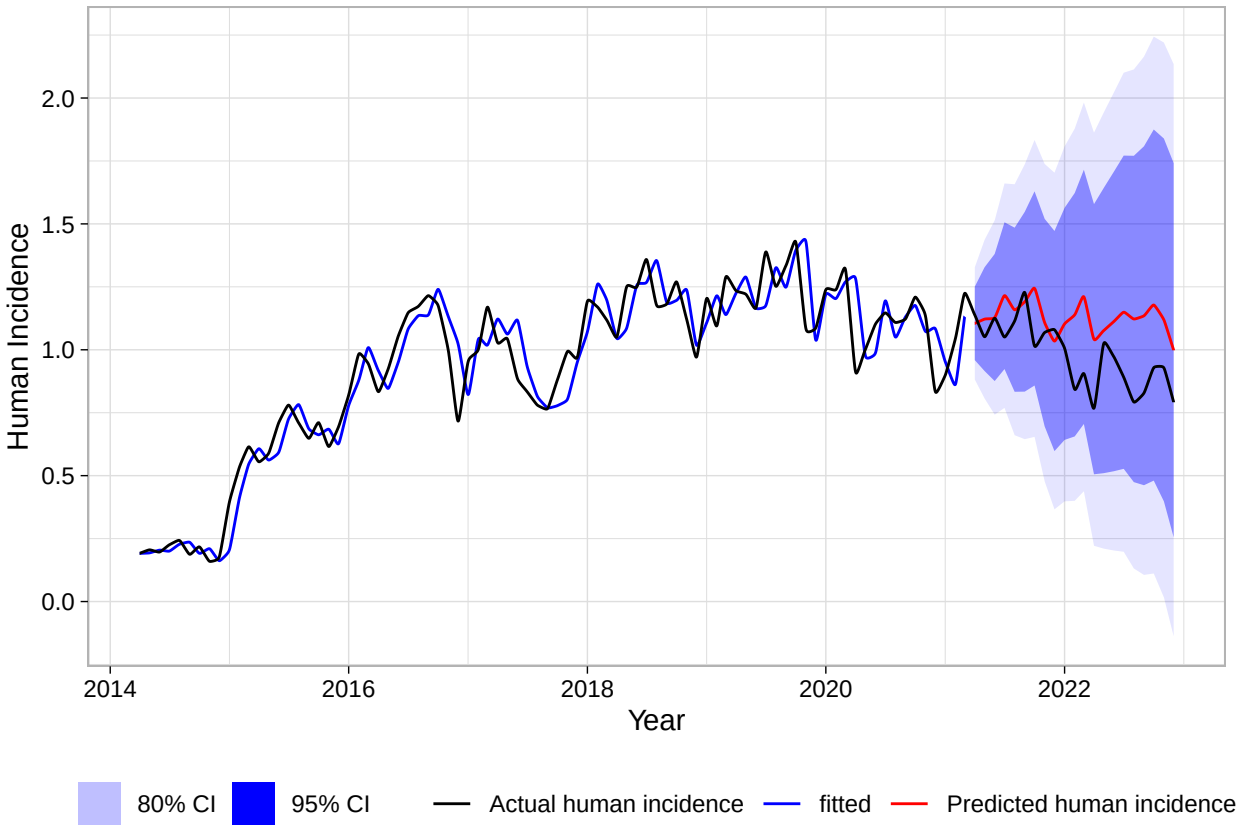
We then forecasted and extracted values for the forecasted as follows;

```
original_data <- df_xts |>
  as.data.frame() %>%
  mutate(date = row.names(.) |> ymd())

fitted <- augment(model_sarima) |>
  as.data.frame() %>%
  select(date, fitted =.fitted, residuals = .resid) |>
  mutate(date = ym(date)
         )

CI <-  hilo(forecast::forecast(model_sarima, h = nrow(test))) |>
  as.data.frame()

CI80 <- CI$`80%` |>
  lapply(FUN = first) |> unlist() |>
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo80", "Hi80"))


CI95 <- CI$`95%` |>
  lapply(FUN = first) |> unlist() |>
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo95", "Hi95"))

forecast_data_human_sarima <-
  data.frame(
    date = row.names(as.data.frame(test)) |>
      ymd(),
    PointForecast = CI$.mean,
    CI80,
    CI95
  )
```

The above data gave the following plot

```
forecast_plot_human_sarima <- ggplot(original_data) +
  geom_ribbon(data = forecast_data_human_sarima, aes(date, ymin = Lo95, ymax = Hi95, fill = "95% CI"), a
```

```
  geom_xspline(data = fitted, aes(date, y = fitted, color = "fitted")) +
  geom_ribbon(data = forecast_data_human_sarima, aes(date, ymin = Lo80, ymax = Hi80, fill = "80% CI"), a
  geom_xspline(data = forecast_data_human_sarima, aes(date, PointForecast, colour = "Predicted human inc
  geom_xspline(aes(date, human_incidence, color = "Actual human incidence")) +
  theme_light() +
  theme(axis.text = element_text(color = "black"),
        axis.title = element_text(color = "black"),
        axis.ticks = element_line(color = "black"),
        legend.position = "bottom",
        plot.title = element_text(color = "black", hjust = .5)
        ) +
  xlab("Year") +
  ylab("Human Incidence") +
  scale_color_manual(
    values = c(
      "Predicted human incidence" = "red",
      "Actual human incidence" = "black",
      'fitted' = 'blue'
    )
  ) +
  scale_fill_manual(
    values = c(
      "95% CI" = "blue",
      "80% CI" = "blue"
    ),
        guide = guide_legend(
      override.aes = list(alpha = c(0.25, 1))
    )
  ) +
  labs(col = NULL, fill = NULL)
#plotly::ggplotly( forecast_plot_human_sarima)
forecast_plot_human_sarima
```



The model metrics were calculated as follows:

```
# Accuracy
test_acc <- test |>
  as.data.frame() %>%
  mutate(date = rownames(.) |> ymd() |> yearmonth()) |>
  as_tsibble() |>
  select(-animal_incidence)



model_sarima_accuracy <- accuracy(forecast::forecast(model_sarima, h = nrow(test)), test_acc,
  measures = list(
    point_accuracy_measures,
    interval_accuracy_measures,
    distribution_accuracy_measures
  )
)

model_sarima_accuracy
```

```
## # A tibble: 1 x 13
##   .model          .type     ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1 winkler
##   <chr>           <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 "ARIMA(human_i~ Test  -0.150 0.194 0.160 -17.0  18.0   NaN   NaN 0.445    1.47
```

```
## # i 2 more variables: percentile <dbl>, CRPS <dbl>
```

2. Model with the covariate

The model without covariate was trained as follows;

```
set.seed(123)
model_sarima_covariates <- train |>
  as.data.frame() %>%
  mutate(date = row.names(.) |>
           zoo::as.yearmon() |>
           yearmonth()) |>
  as_tsibble() |>
  model(ARIMA(
    human_incidence ~ animal_incidence,
    ic = "aic",
    stepwise = T
  )) |>
  report()
```

```
## Series: human_incidence
## Model: LM w/ ARIMA(0,1,0)(2,0,0)[12] errors
##
## Coefficients:
##         sar1    sar2  animal_incidence
##       0.2479  0.3386            0.0299
## s.e.  0.1043  0.1170            0.1272
##
## sigma^2 estimated as 0.0131:  log likelihood=61.31
## AIC=-114.61   AICc=-114.1   BIC=-104.94
```

We then forecasted and extracted values for the forecasted as follows;

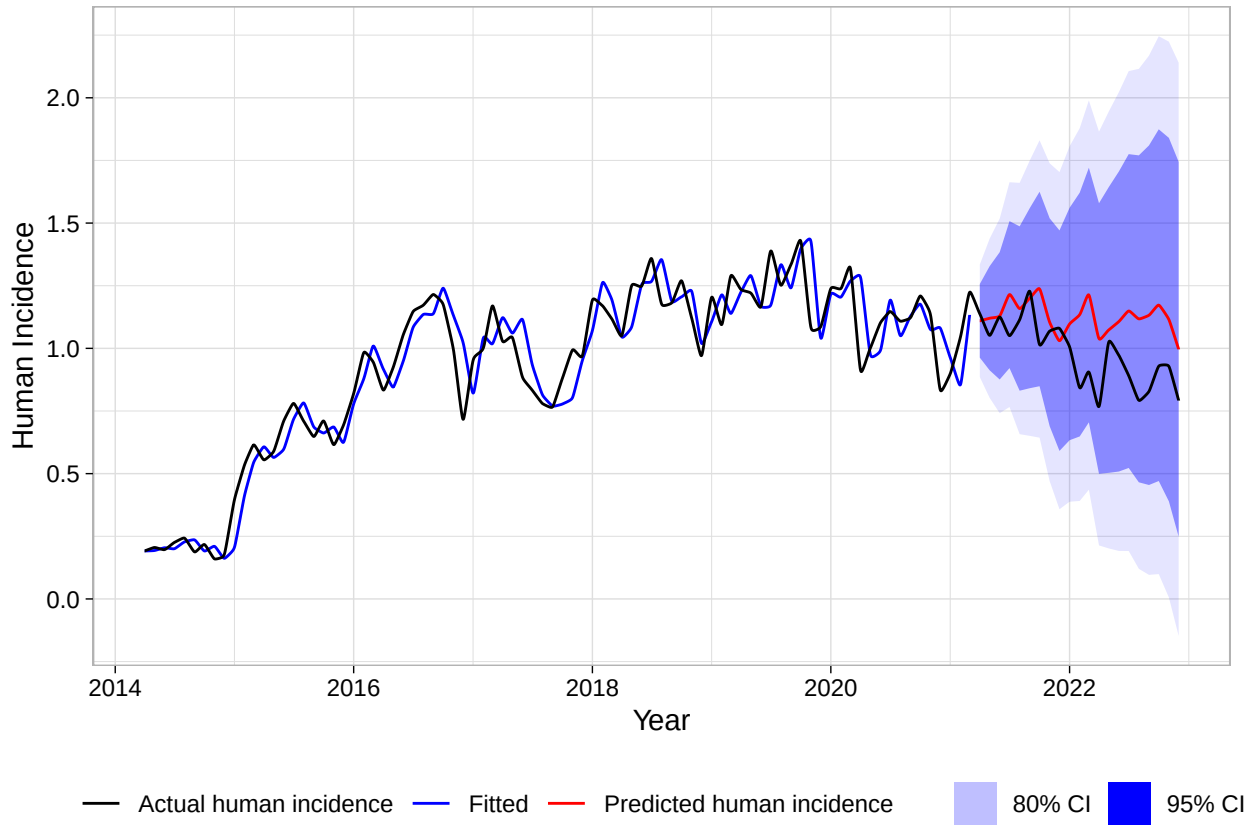
```
fitted_covariate <- augment(model_sarima_covariates) |>
  as.data.frame() %>%
  select(date, fitted =.fitted, residuals = .resid) |>
  mutate(date = ym(date)
  )


test_tsibble <- test |>
  as.data.frame() %>%
  mutate(date = rownames(.) |> ymd() |> yearmonth()) |>
  as_tsibble() |>
  select(-human_incidence)


CI_covariate <- model_sarima_covariates |>
  forecast(new_data = test_tsibble) |>
  hilo()
```

```
CI_covariate80 <- CI_covariate$`80%` |>
  lapply(FUN = first) |>
  unlist() |>
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo80", "Hi80"))


CI_covariate95 <- CI_covariate$`95%` |>
  lapply(FUN = first) |> unlist() |>
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo95", "Hi95"))

forecast_data_covariate_sarima <-
  data.frame(
    date = row.names(as.data.frame(test)) |>
      ymd(),
    PointForecast = CI_covariate$.mean,
    CI_covariate80,
    CI_covariate95
  )
```

The above data gave the following plot

```
forecast_plot_covariate_sarima <- ggplot(original_data) +
  geom_ribbon(data = forecast_data_covariate_sarima, aes(date, ymin = Lo95, ymax = Hi95, fill = "95% CI
  geom_ribbon(data = forecast_data_covariate_sarima, aes(date, ymin = Lo80, ymax = Hi80, fill = "80% CI
  geom_xspline(data = fitted_covariate, aes(date, y = fitted, color = "Fitted")) +
  geom_xspline(data = forecast_data_covariate_sarima, aes(date, PointForecast, colour = "Predicted huma
  geom_xspline(aes(date, human_incidence, color = "Actual human incidence")) +
  theme_light() +
  theme(axis.text = element_text(color = "black"),
        axis.title = element_text(color = "black"),
        axis.ticks = element_line(color = "black"),
        legend.position = "bottom",
                plot.title = element_text(color = "black", hjust = .5)
        ) +
  xlab("Year") +
  ylab("Human Incidence") +
  scale_color_manual(
    values = c(
      "Predicted human incidence" = "red",
      "Actual human incidence" = "black",
      'Fitted' = 'blue'
    )
  ) +
  scale_fill_manual(
    values = c(
      "95% CI" = "blue",
      "80% CI" = "blue"
```

```
    ),
    guide = guide_legend(
      override.aes = list(alpha = c(0.25, 1))
    )
  ) +
  labs(col = NULL, fill = NULL)
forecast_plot_covariate_sarima
```



The model metrics were calculated as follows:

```
model_sarima_covariates_coefficients <-
  tidy(model_sarima_covariates) |>
  select(-.model) %>%
  as_tibble() %>%
  group_by(term)


# Accuracy
test_acc2 <- test |>
  as.data.frame() %>%
  mutate(date = rownames(.) |> ymd() |> yearmonth()) |>
  as_tsibble() |>
  select(-animal_incidence)

model_sarima_covariate_accuracy <-
  accuracy(
```

```
    model_sarima_covariates |>
      forecast(new_data = test_tsibble) ,
    test_acc,
    measures = list(
      point_accuracy_measures,
      interval_accuracy_measures,
      distribution_accuracy_measures
    )
  )

model_sarima_covariate_accuracy
```

```
## # A tibble: 1 x 13
##   .model          .type     ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1 winkler
##   <chr>           <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 "ARIMA(human_i~ Test  -0.148 0.192 0.158 -16.8  17.7   NaN   NaN 0.454    1.48
## # i 2 more variables: percentile <dbl>, CRPS <dbl>
```

3. Comparison of the Model with and Without a covariate

```
model_sarima_accuracy2 <- model_sarima_accuracy |>
    mutate(model = "Model without Covariate") |>
  select(-.model, -.type) |>
  select(model, RMSE, MAE, MAPE)

model_sarima_covariate_accuracy2 <- model_sarima_covariate_accuracy |>
  mutate(model = "Model with Covariate") |>
  select(-.model, -.type) |>
  select(model, RMSE, MAE, MAPE)

metrics_df <- rbind(model_sarima_accuracy2, model_sarima_covariate_accuracy2)
write.csv(metrics_df, "metrics_df.csv", row.names = F)

kableExtra::kable(metrics_df)
```

model	RMSE	MAE	MAPE
Model without Covariate	0.1939374	0.1604385	17.97294
Model with Covariate	0.1915698	0.1576527	17.68426

The model with the covariate was better, based on the RMSE, MAE and MAPE. Thus, we used this to forecast the values for the year 2023,

Full model for forecasting 2023 data

we fit a time series data with animal incidence as the covariate with data from 2014 to 2022 and then use it to forecast 2023 human brucellosis incidence. The model was fitted as follows;

```r
# Time series so that we can be able to forecast 2023.
set.seed(123)
full.model <- df_xts |>
  as.data.frame() %>%
  mutate(date = row.names(.) |>
           zoo::as.yearmon() |>
           yearmonth()) |>
  as_tsibble() |>
  model(ARIMA(
    human_incidence ~ animal_incidence,
    ic = "aic",
    stepwise = T
  )) |>
  report()
```

```
## Series: human_incidence
## Model: LM w/ ARIMA(2,1,2)(1,0,1)[12] errors
##
## Coefficients:
##          ar1      ar2      ma1     ma2    sar1     sma1  animal_incidence
##       1.3320  -0.8608  -1.4960  0.9014  0.8822  -0.6681            0.2404
## s.e.  0.0775   0.0719   0.0728  0.0759  0.1273   0.2097            0.0950
##
## sigma^2 estimated as 0.01161:  log likelihood=85.34
## AIC=-154.67   AICc=-153.16   BIC=-133.52
```

Then, the forecasted data is curated as follows;

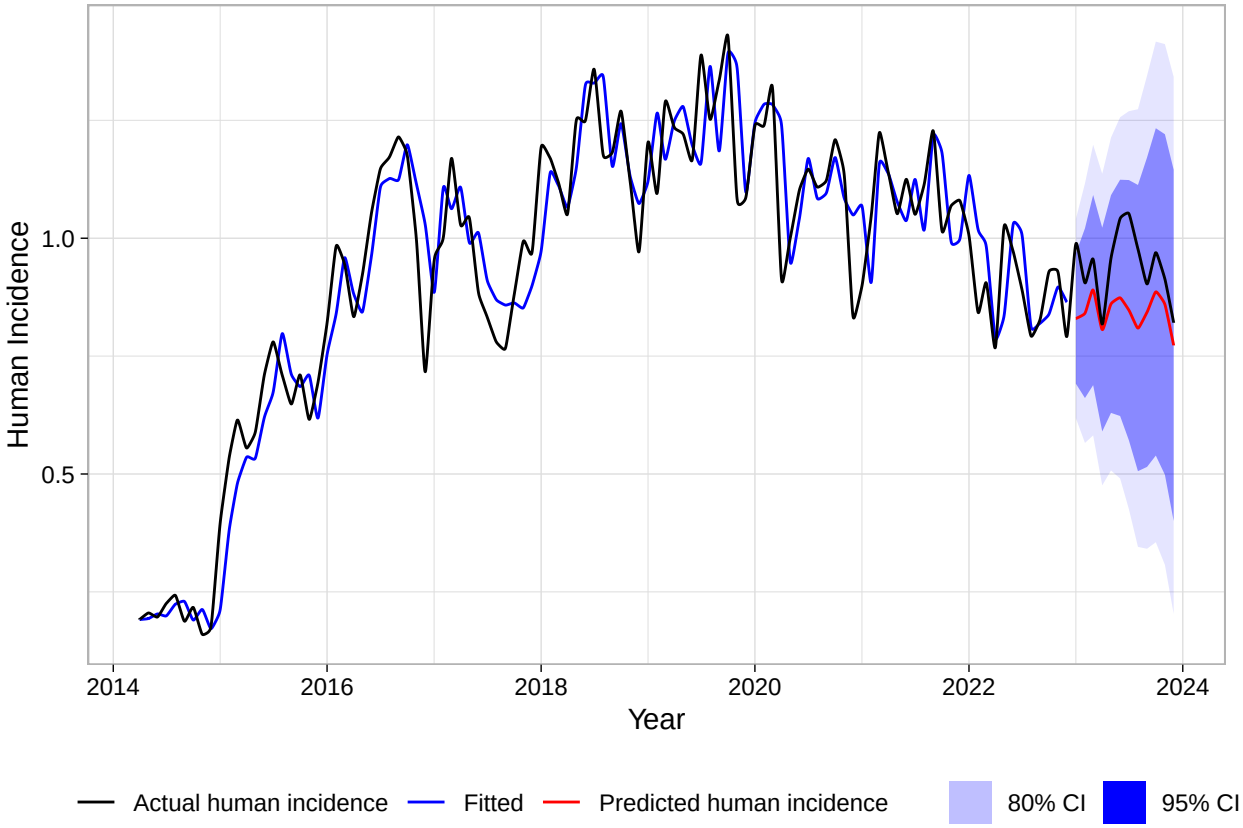
```r
fitted_full <- augment(full.model) |>
  as.data.frame() %>%
  select(date, fitted =.fitted, residuals = .resid) |>
  mutate(date = ym(date)
  )

original_data2 <-   df_xts |>
  rbind(forecast.df) |>
  as.data.frame() %>%
  mutate(date = row.names(.) |> ymd())

test_tsibble <- forecast.df |>
  as.data.frame() %>%
  mutate(date = rownames(.) |> ymd() |> yearmonth()) |>
  as_tsibble() |>
  select(-human_incidence)


CI_full <- full.model |>
  forecast(new_data = test_tsibble) |>
  hilo()

CI_full80 <- CI_full$`80%` |>
  lapply(FUN = first) |>
  unlist() |>
```

```
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo80", "Hi80"))


CI_full95 <- CI_full$`95%` |>
  lapply(FUN = first) |> unlist() |>
  matrix(ncol = 3, byrow = T) |>
  as.data.frame() |>
  select(-V3) |>
  setNames(c("Lo95", "Hi95"))

forecast_data_full_sarima <-
  data.frame(
    date = row.names(as.data.frame(forecast.df)) |>
      ymd(),
    PointForecast = CI_full$.mean,
    CI_full80,
    CI_full95
  )
```

The above gave the following results

```
forecast_plot_full_sarima <- ggplot(original_data2) +
  geom_ribbon(data = forecast_data_full_sarima, aes(date, ymin = Lo95, ymax = Hi95, fill = "95% CI"), al
  geom_ribbon(data = forecast_data_full_sarima, aes(date, ymin = Lo80, ymax = Hi80, fill = "80% CI"), al
  geom_xspline(data = fitted_full, aes(date, y = fitted, color = "Fitted")) +
  geom_xspline(data = forecast_data_full_sarima, aes(date, PointForecast, colour = "Predicted human inci
  geom_xspline(aes(date, human_incidence, color = "Actual human incidence")) +
  theme_light() +
  theme(axis.text = element_text(color = "black"),
        axis.title = element_text(color = "black"),
        axis.ticks = element_line(color = "black"),
        legend.position = "bottom",
        plot.title = element_text(color = "black", hjust = .5)
  ) +
  xlab("Year") +
  ylab("Human Incidence") +
  scale_color_manual(
    values = c(
      "Predicted human incidence" = "red",
      "Actual human incidence" = "black",
      'Fitted' = 'blue'
    )
  ) +
  scale_fill_manual(
    values = c(
      "95% CI" = "blue",
      "80% CI" = "blue"
    ),
    guide = guide_legend(
```

```
      override.aes = list(alpha = c(0.25, 1))
    )
  ) +
  labs(col = NULL, fill = NULL)
forecast_plot_full_sarima
```



The model coefficients are;

```
full.model_coefficients <-
  tidy(full.model) |>
  select(-.model) %>%
  as_tibble() %>%
  group_by(term)
knitr::kable(full.model_coefficients)
```

term	estimate	std.error	statistic	p.value
ar1	1.3320208	0.0775118	17.184739	0.0000000
ar2	-0.8607882	0.0719157	-11.969409	0.0000000
ma1	-1.4959604	0.0728340	-20.539327	0.0000000
ma2	0.9014467	0.0759216	11.873395	0.0000000
sar1	0.8821661	0.1273002	6.929811	0.0000000
sma1	-0.6681158	0.2097406	-3.185439	0.0019081
animal_incidence	0.2404006	0.0949881	2.530849	0.0128762

The results of the forecasted values versus the actual, together with the upper 95% CI were as follows;

```r
# Forecasted and Actual data
fore.actual <- forecast_data_full_sarima |>
  mutate(Actual = as.vector(forecast.df$human_incidence),
         Date = as.Date(date) |>
           zoo::as.yearmon() |>
           yearmonth()) |>
  select(Date = Date,
         Forecasted = PointForecast,
         Actual,
         `Lower 95% CI` = Lo95,
         `Upper 95% CI` = Hi95,
         ) |>
  mutate(across(where(is.numeric), ~as.numeric(round(., 3))))
knitr::kable(fore.actual)
```

Date	Forecasted	Actual	Lower 95% CI	Upper 95% CI
2023 Jan	0.830	0.988	0.618	1.041
2023 Feb	0.841	0.905	0.566	1.116
2023 Mar	0.890	0.955	0.582	1.198
2023 Apr	0.806	0.818	0.476	1.137
2023 May	0.860	0.958	0.507	1.213
2023 Jun	0.874	1.041	0.490	1.257
2023 Jul	0.847	1.052	0.425	1.269
2023 Aug	0.809	0.977	0.345	1.274
2023 Sep	0.843	0.903	0.341	1.345
2023 Oct	0.886	0.969	0.355	1.417
2023 Nov	0.859	0.913	0.307	1.412
2023 Dec	0.773	0.821	0.203	1.342