Brucellosis Codes

Table of contents

## Data Processing

source('clean\_all\_data.R')

Loading required package: pacman

Warning: Expecting numeric in I18 / R18C9: got a date

Warning: Expecting numeric in L18 / R18C12: got a date

Warning: Expecting numeric in I19 / R19C9: got a date

Warning: Expecting numeric in L19 / R19C12: got a date

Warning: Expecting numeric in I85 / R85C9: got a date

New names:  
• `year` -> `year...10`  
• `year` -> `year...13`

Rows: 282 Columns: 3  
── Column specification ────────────────────────────────────────────────────────  
Delimiter: ","  
chr (2): County, species  
dbl (1): species\_num  
  
ℹ Use `spec()` to retrieve the full column specification for this data.  
ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.  
`summarise()` has grouped output by 'county', 'year', 'date', 'diseases', 'diagnosis'. You can override using the `.groups` argument.

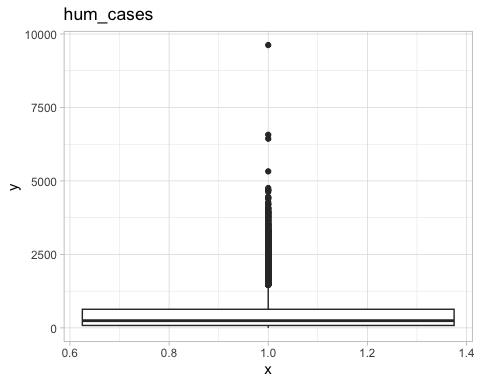
# Identifying the outliers in the number of cases  
numeric\_columns <- df\_incidence2.1 %>%  
 dplyr::select(contains('cases')) |>   
 keep(is.numeric)  
  
numeric\_columns %>%  
 map(~summary(.))

$hum\_cases  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 0.0 82.0 245.0 501.4 633.0 9623.0 26   
  
$catt\_cases  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 0.000 1.000 1.000 3.584 3.000 70.000 9240   
  
$cam\_cases  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 1.000 1.000 1.500 3.125 2.500 13.000 9369   
  
$goat\_cases  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 0.00 1.00 4.00 21.24 13.00 528.00 9306   
  
$shp\_cases  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 0.00 1.00 2.00 6.76 8.00 81.00 9352

numeric\_columns %>%  
 imap(  
 ~ ggplot(data = data.frame(y = .x)) +  
 geom\_boxplot(aes(x = 1, y = y)) +  
 labs(title = .y) +  
 theme\_light()  
 )

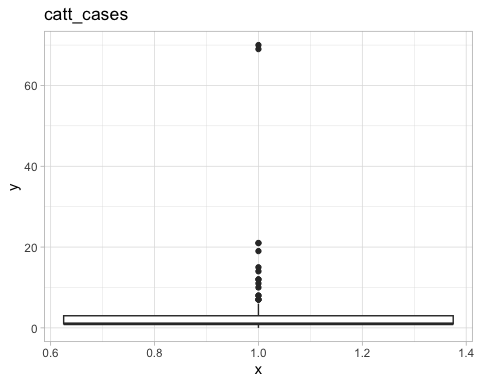
$hum\_cases

Warning: Removed 26 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



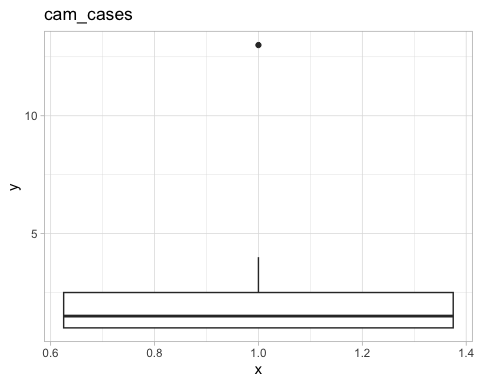
$catt\_cases

Warning: Removed 9240 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



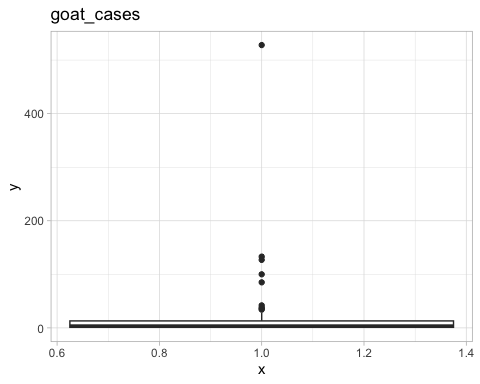
$cam\_cases

Warning: Removed 9369 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



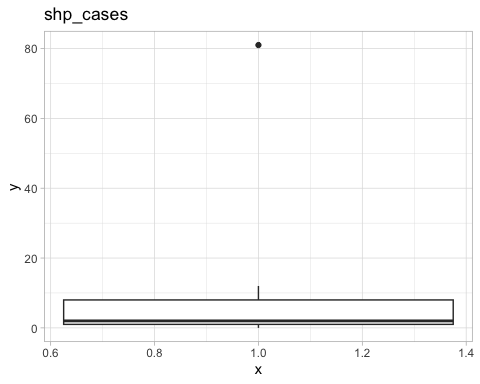
$goat\_cases

Warning: Removed 9306 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



$shp\_cases

Warning: Removed 9352 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



# 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)))

`summarise()` has grouped output by 'date'. You can override using the  
`.groups` argument.

# Population per year, per county  
df\_pop <- df\_incidence2 |>   
 dplyr::select(date, county, contains("pop")) %>%  
 distinct(.) |>  
 as\_tibble() |>  
 group\_by(date, county) %>%  
 summarise(across(where(is.numeric), ~unique(.)))

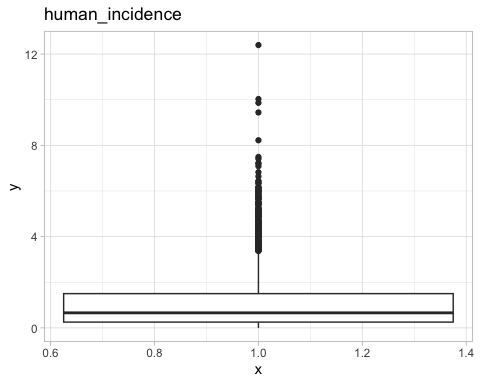
`summarise()` has grouped output by 'date'. You can override using the  
`.groups` argument.

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)  
 ) |>  
 dplyr::select(date, county, contains(c("incidence", "cases"))) |>  
 as\_tibble() |>   
 arrange(county)  
  
df\_1\_pop <- df\_tot\_cases |>  
 merge(df\_pop, by = c("date", "county")) |>   
 dplyr::select(date, county, pop)  
write.csv(df\_1\_pop, 'county\_humanpop.csv', row.names = F)  
  
df\_1\_complete <- df\_1 |>  
 dplyr::select(  
 date,  
 county,  
 human\_incidence,  
 catt\_incidence,  
 goat\_incidence,  
 hum\_cases,  
 catt\_cases,  
 goat\_cases  
 ) |>  
 filter(catt\_incidence > 0)  
write.csv(df\_1\_complete, "df\_1\_complete.csv", row.names = F)  
  
# Outliers  
numeric\_columns <- df\_1 %>%  
 dplyr::select(contains('incidence')) |>   
 keep(is.numeric)  
  
numeric\_columns %>%  
 map(~summary(.))

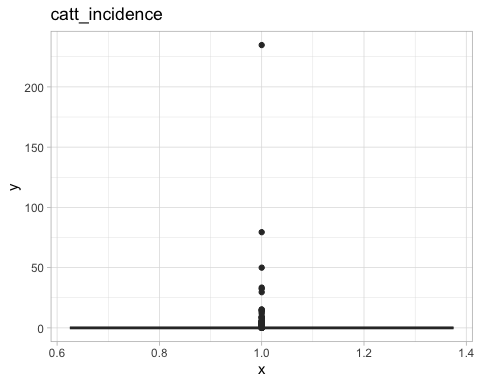
$human\_incidence  
 Min. 1st Qu. Median Mean 3rd Qu. Max.   
 0.0000 0.2528 0.6593 1.0400 1.5011 12.3950   
  
$catt\_incidence  
 Min. 1st Qu. Median Mean 3rd Qu. Max.   
 0.0000 0.0000 0.0000 0.1546 0.0000 234.7197   
  
$cam\_incidence  
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
 0.0000 0.0000 0.0000 0.0417 0.0000 121.9558 1911   
  
$goat\_incidence  
 Min. 1st Qu. Median Mean 3rd Qu. Max.   
 0.00000 0.00000 0.00000 0.03123 0.00000 27.45740   
  
$shp\_incidence  
 Min. 1st Qu. Median Mean 3rd Qu. Max.   
 0.00000 0.00000 0.00000 0.01431 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

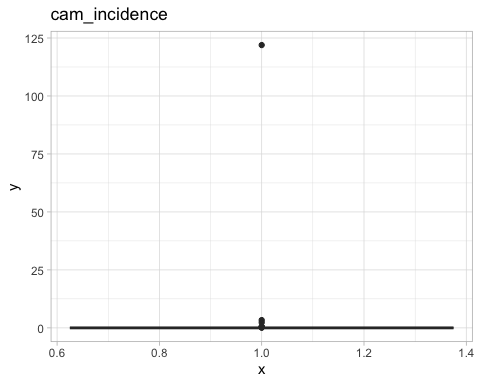


$catt\_incidence

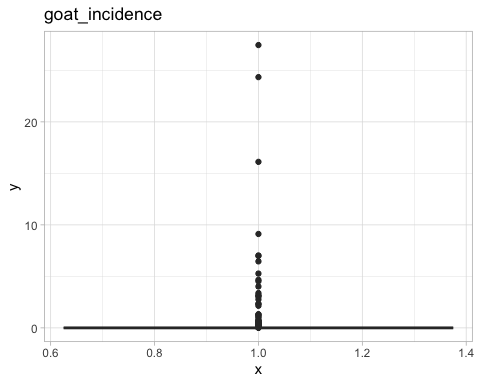


$cam\_incidence

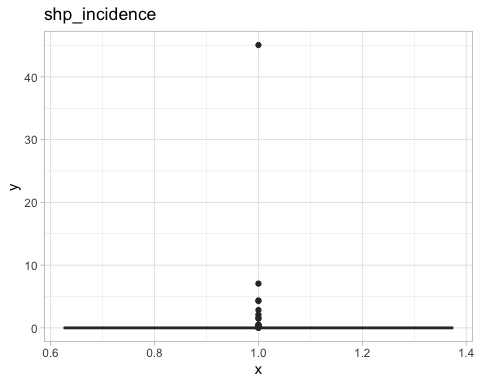
Warning: Removed 1911 rows containing non-finite outside the scale range  
(`stat\_boxplot()`).



$goat\_incidence

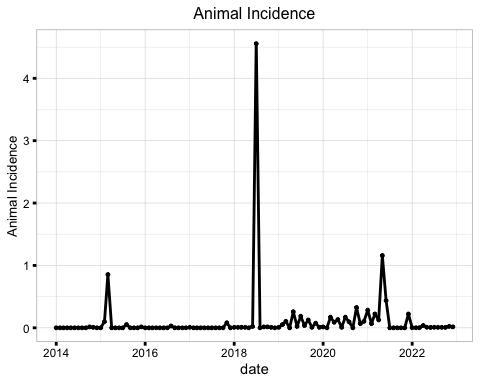


$shp\_incidence

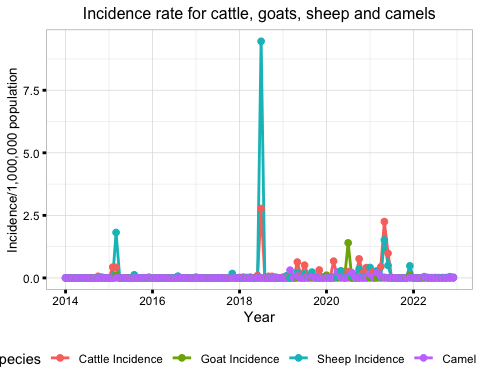


df\_cum <- df\_tot\_cases |>  
 merge(df\_pop, by = c("date", "county")) |>   
 filter(!is.na(date)) |>   
 rowwise() |>   
 mutate(  
 animal\_cases = sum(catt\_cases, goat\_cases, shp\_cases, cam\_cases, na.rm = T),  
 animal\_pop = sum(catt\_pop, goat\_pop, sheep\_pop, cam\_pop, na.rm = T),  
 animal\_cases = ifelse(animal\_cases == 0, NA, animal\_cases),  
 animal\_incidence = round((animal\_cases / animal\_pop) \* 1000000, 4),  
 human\_incidence = round((hum\_cases / pop) \* 1000, 4)  
 ) |>   
 dplyr::select(date, county, contains(c("incidence", 'cases'))) |>   
 as\_tibble() |>   
 mutate()  
  
df\_cum\_complete <- df\_cum |>   
 dplyr::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\_incidence2.1 |>   
 dplyr::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))))

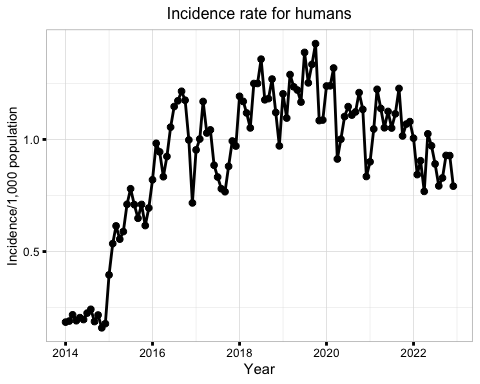
df\_incidence2\_trend <- df\_incidence2.1 |>   
 dplyr::select(date,   
 county,   
 diseases,   
 diagnosis,   
 contains("cases"),   
 catt\_pop,   
 goat\_pop,   
 sheep\_pop,   
 cam\_pop) |>  
 mutate(year = year(as.Date(date))) |>   
 merge(pop, by = c("county", "year"))   
  
df\_tot\_cases\_trend <- 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 |>  
 dplyr::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)  
 ) |>  
 dplyr::select(date, contains("incidence")) |>  
 as\_tibble() |>   
 mutate(  
   
 )  
  
write.csv(df\_1\_trend, "individual\_incidence.csv")  
  
df\_1\_trend\_complete <- df\_1\_trend |>  
 dplyr::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)  
 ) |>  
 dplyr::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)  
 ) |>   
 dplyr::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() |>   
 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)  
 ) |>   
 dplyr::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))

Warning in adf.test(na.omit(df\_1$human\_incidence)): p-value smaller than  
printed p-value

adf2 <- adf.test(na.omit(df\_1$catt\_incidence))

Warning in adf.test(na.omit(df\_1$catt\_incidence)): p-value smaller than printed  
p-value

adf3 <- adf.test(na.omit(df\_1$cam\_incidence))

Warning in adf.test(na.omit(df\_1$cam\_incidence)): p-value smaller than printed  
p-value

adf4 <- adf.test(na.omit(df\_1$goat\_incidence))

Warning in adf.test(na.omit(df\_1$goat\_incidence)): p-value smaller than printed  
p-value

adf5 <- adf.test(na.omit(df\_1$shp\_incidence))

Warning in adf.test(na.omit(df\_1$shp\_incidence)): p-value smaller than printed  
p-value

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))

Warning in adf.test(na.omit(df\_1\_trend$cam\_incidence)): p-value smaller than  
printed p-value

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))

Warning in adf.test(na.omit(df\_cum$animal\_incidence)): p-value smaller than  
printed p-value

adf\_combined\_trend <- adf.test(df\_cum\_trend$animal\_incidence)

Warning in adf.test(df\_cum\_trend$animal\_incidence): p-value smaller than  
printed p-value

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)  
)

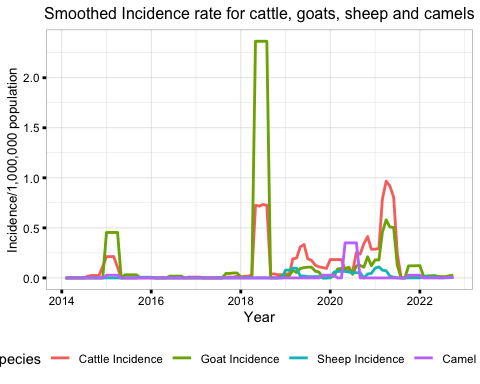
# dplyr::select columns containing "incidence"  
incidence\_cols <- grep("incidence", names(df\_1\_trend), value = TRUE)  
  
# Apply moving average smoothing to dplyr::selected columns  
smoothed\_df <- df\_1\_trend %>%  
 mutate(across(incidence\_cols, ~ifelse(is.na(.), 0, .))) |>   
 mutate(across(all\_of(incidence\_cols), ~zoo::rollmean(., k = 4, fill = NA), .names = "smoothed\_{.col}")) |>   
 na.omit()

Warning: There was 1 warning in `mutate()`.  
ℹ In argument: `across(incidence\_cols, ~ifelse(is.na(.), 0, .))`.  
Caused by warning:  
! Using an external vector in selections was deprecated in tidyselect 1.1.0.  
ℹ Please use `all\_of()` or `any\_of()` instead.  
 # Was:  
 data %>% select(incidence\_cols)  
  
 # Now:  
 data %>% select(all\_of(incidence\_cols))  
  
See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.

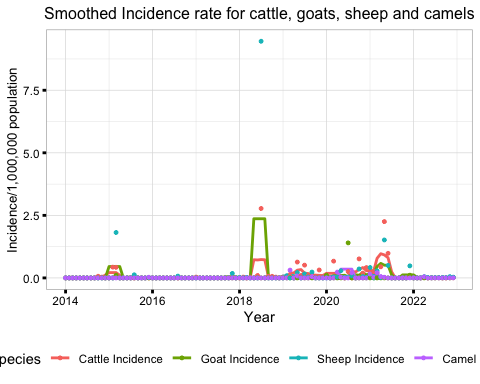
# Print the first few rows of the smoothed data  
head(smoothed\_df)

# A tibble: 6 × 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>

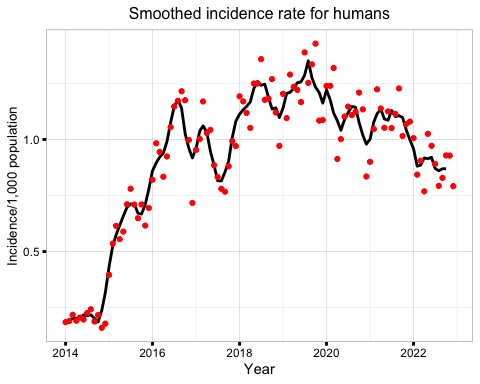
order <- c(  
 "smoothed\_human\_incidence",  
 "smoothed\_catt\_incidence",  
 "smoothed\_goat\_incidence",  
 "smoothed\_shp\_incidence",  
 "smoothed\_cam\_incidence"  
)  
  
  
order2 <- c("Human Incidence", "Cattle Incidence", "Camel Incidence", "Goat Incidence",   
"Sheep Incidence")  
  
trend\_data\_smoothed <- smoothed\_df %>%  
 dplyr::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, .))) |>   
 dplyr::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



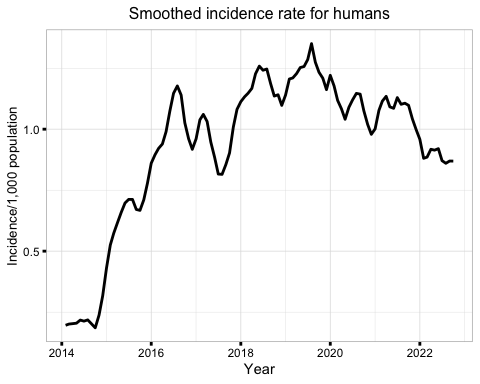
# With points  
species\_sm\_plt\_points <- trend\_data\_smoothed |>   
 filter(name != "Human Incidence") |>   
 ggplot(aes(x = date)) +  
 geom\_line(aes(y = value, col = name), linewidth = 1) +  
 geom\_point(data = trend\_data |>   
 mutate(value = ifelse(is.na(value), 0, value)) |>   
 filter(name != "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 dplyr::selected columns  
# dplyr::select columns containing "incidence"  
incidence\_cols <- grep("incidence", names(df\_cum\_trend), value = TRUE)  
  
# Apply moving average smoothing to dplyr::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

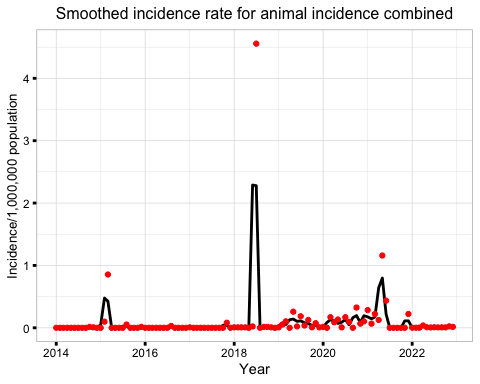


table1.1 <- df\_incidence2.1 |>  
 dplyr::select(county, diagnosis, contains("cases")) |>  
 mutate(  
 catt\_cases = ifelse(catt\_cases >= 69, round(mean(catt\_cases, na.rm = T)), catt\_cases),  
 goat\_cases = ifelse(goat\_cases > 28, round(mean(goat\_cases, na.rm = T)), goat\_cases)  
 ) |>  
 pivot\_longer(cols = -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() |>   
 dplyr::select(-name) |>   
 group\_by(Species, Diagnosis) |>   
 group\_by(Species) |>   
 mutate(`Percent(%)` = round((Cases/sum(Cases, na.rm = T)) \* 100, 2)) |>   
 mutate(Cases = format(round(Cases), big.mark = ',') |> str\_squish()) |>   
 dplyr::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")

`summarise()` has grouped output by 'name'. You can override using the  
`.groups` argument.

table1.1

Number of cases according to the type of Diagnosis

| Species | Diagnosis | Cases | Percent(%) |
| --- | --- | --- | --- |
| Camels | Clinically confirmed | 23 | 92.00 |
| Camels | Lab confirmed | 2 | 8.00 |
| Camels | Post Mortem | 0 | 0.00 |
| Cattle | Clinically confirmed | 225 | 62.50 |
| Cattle | Lab confirmed | 121 | 33.61 |
| Cattle | Post Mortem | 14 | 3.89 |
| Goats | Clinically confirmed | 451 | 78.57 |
| Goats | Lab confirmed | 123 | 21.43 |
| Goats | Post Mortem | 0 | 0.00 |
| Humans | Clinically confirmed | 1,058,294 | 22.57 |
| Humans | Lab confirmed | 3,630,512 | 77.43 |
| Humans | Post Mortem | 0 | 0.00 |
| Sheep | Clinically confirmed | 152 | 89.94 |
| Sheep | Lab confirmed | 17 | 10.06 |
| Sheep | Post Mortem | 0 | 0.00 |

write.csv( df\_incidence |>   
 dplyr::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() |>   
 dplyr::select(-name) |>   
 group\_by(Species, Diagnosis) |>   
 group\_by(Species) |>   
 mutate(`Percent(%)` = round((Cases/sum(Cases)) \* 100, 2)) |>   
 dplyr::select(3, 1,2,4),  
   
 "cases\_table.csv")

`summarise()` has grouped output by 'name'. You can override using the  
`.groups` argument.

# The descriptive statistics are for the Incidence Rate National Wide  
table2 <- df\_1 %>%  
 dplyr::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)  
 ) |>   
 dplyr::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

Descriptive Statistics for Incidence Rate

| Species | Mean Incidence Rate | Minimum | Median | Maximum | Standard Deviation |
| --- | --- | --- | --- | --- | --- |
| Human | 1.0400485 | 0 | 0.66 | 12 | 1.114 |
| Cattle | 0.1546213 | 0 | 0.00 | 235 | 3.710 |
| Camel | 0.0416705 | 0 | 0.00 | 122 | 2.191 |
| Goat | 0.0312345 | 0 | 0.00 | 27 | 0.630 |
| Sheep | 0.0143072 | 0 | 0.00 | 45 | 0.653 |

write.csv(df\_1 %>%  
 dplyr::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)  
 ) |>   
 dplyr::select(Species = name,  
 `Mean Incidence Rate`,  
 Minimum,  
 Median,  
 Maximum,  
 `Standard Deviation`),  
 "descriptive\_table.csv")

# Cases per year per county  
df\_tot\_cases\_spatial <- df\_incidence2.1 |>  
 group\_by(year = year(as.Date(date)), county) |>  
 summarise(across(contains("cases"), ~ sum(., na.rm = T))) |>  
 mutate(across(contains('cases'), ~ ifelse(. == 0, NA, .)))

`summarise()` has grouped output by 'year'. You can override using the  
`.groups` argument.

# Population per year, per county  
df\_pop\_spatial <- df\_incidence2.1 |>  
 dplyr::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, .)))

`summarise()` has grouped output by 'year'. You can override using the  
`.groups` argument.

# Individual Cases per year per county month  
df\_tot\_cases\_spatial\_month <- df\_incidence2.1 |>  
 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)  
 ) |>  
 dplyr::select(year, date, county, contains(c("incidence", "cases"))) |>  
 as\_tibble()

`summarise()` has grouped output by 'date'. You can override using the  
`.groups` argument.

write.csv(df\_tot\_cases\_spatial\_month,  
 "df\_tot\_cases\_spatial\_month.csv")  
  
df\_tot\_cases\_spatial\_month\_complete <- df\_tot\_cases\_spatial\_month |>  
 dplyr::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) %>%  
 ifelse(. == 0, NA, .),  
 animal\_incidence = round((animal\_cases / animal\_pop) \* 1000000, 4),  
 human\_incidence = round((hum\_cases / pop) \* 1000, 4)  
 ) |>  
 dplyr::select(date, year, county, contains(c("incidence", "cases"))) |>  
 as\_tibble()  
write.csv(df\_spatial\_cum\_month, "df\_spatial\_cum\_month.csv")  
  
# Population per year, per county  
df\_pop\_spatial <- df\_incidence2.1 |>  
 dplyr::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, .)))

`summarise()` has grouped output by 'year'. You can override using the  
`.groups` argument.

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)  
 ) |>  
 dplyr::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) %>%  
 ifelse(. == 0, NA, .),  
 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)  
 ) |>  
 dplyr::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"

setdiff(df\_spatial$county, shp$Name)

[1] "Murang'a"

# Replacing Muranga to Murang'a  
shp <- shp |>  
 mutate(Name = ifelse(Name == "Muranga", "Murang'a", Name))  
setdiff(shp$Name, df\_spatial$county)

character(0)

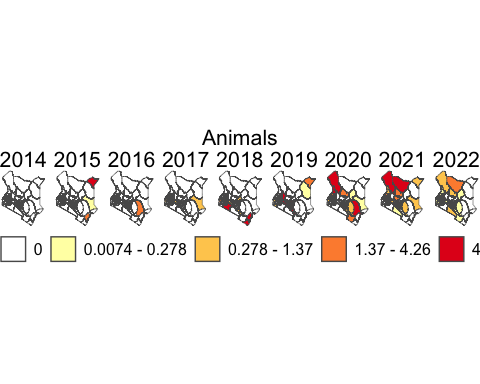
setdiff(df\_spatial$county, shp$Name)

character(0)

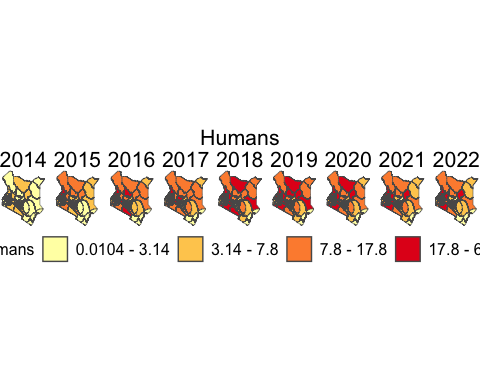
length(unique(df\_spatial$county))

[1] 47

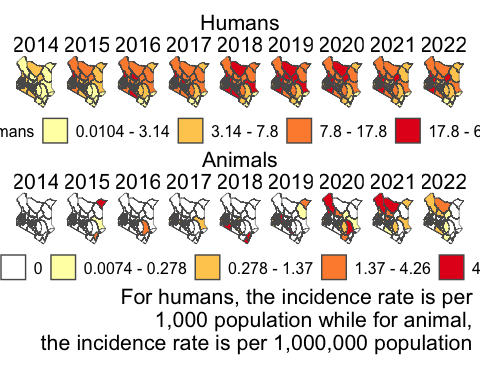
# Merging  
df\_spatial\_merged <- df\_spatial |>  
 merge(shp, by.x = "county", by.y = 'Name')  
indi\_incidence\_columns <- grep("incidence", names(df\_spatial\_merged), value = TRUE)  
  
df\_spatial\_merged <- df\_spatial\_merged |>  
 mutate(across(  
 all\_of(indi\_incidence\_columns),  
 ~ cut(  
 .,  
 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", # These are the quantiles  
 "[0.0074,0.278]",  
 "(0.278,1.37]" ,  
 "(1.37,4.26]" ,  
 "(4.26,336]")  
 )) |>  
 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



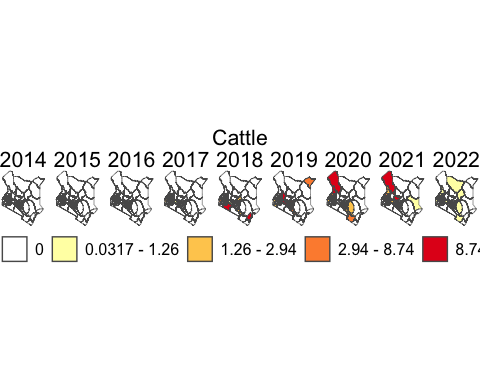
# Humans  
cate\_human <-  
 length(levels(df\_spatial\_merged\_cum$human\_incidence\_range))  
  
humans <-   
 # mutate(human\_incidence\_range = ifelse(is.na(human\_incidence\_range),  
 # "0",  
 # as.character(human\_incidence\_range)) %>%  
 # factor(., levels = c(  
 # "0",  
 # "[0.113,0.865]",  
 # "(0.865,1.85]",  
 # "(1.85,5.62]",  
 # "(5.62,67.4]"  
 # ))) |>  
 ggplot() +  
 #geom\_sf(data = shp, fill = 'white') +  
 geom\_sf(data = df\_spatial\_merged\_cum, 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



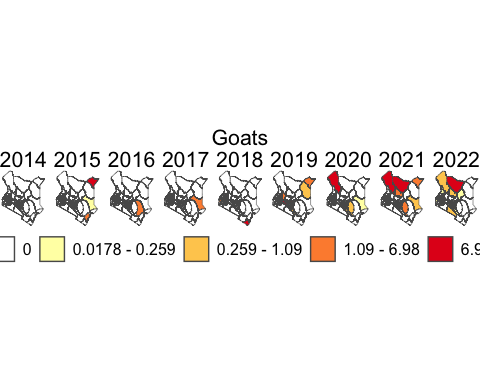
animals\_humans <-  
 wrap\_plots(humans,  
 animals,  
 ncol = 1,  
 guides = "keep") +  
 plot\_annotation(caption = "For humans, the incidence rate is per  
 1,000 population while for animal,  
 the incidence rate is per 1,000,000 population  
 ") &  
 theme(plot.caption = element\_text(size = 16, colour = "black"))  
  
animals\_humans



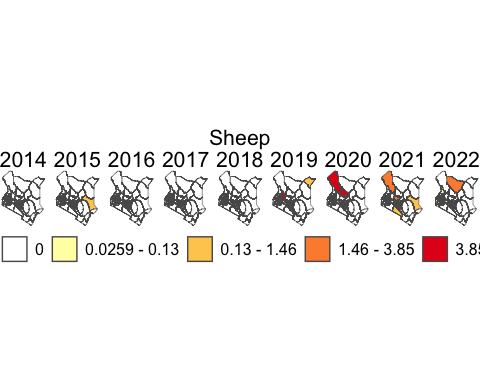
cate\_catt <- length(levels(df\_spatial\_merged$catt\_incidence\_range))  
cattle <- df\_spatial\_merged |>  
 mutate(catt\_incidence\_range = ifelse(is.na(catt\_incidence\_range),   
 "0",   
 as.character(catt\_incidence\_range)) %>%  
 factor(., levels = c(  
 "0",  
 "[0.0317,1.26]",  
 "(1.26,2.94]",  
 "(2.94,8.74]",  
 "(8.74,235]"   
 ))) |>   
 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



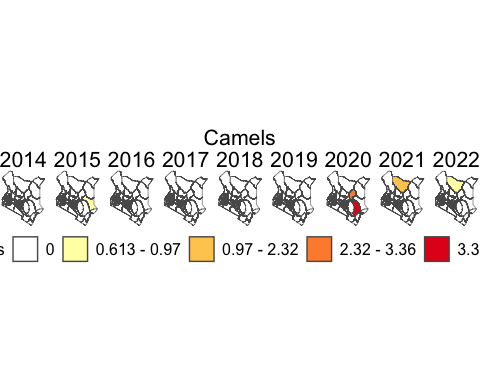
# Goats  
cate\_goat <- length(levels(df\_spatial\_merged$goat\_incidence\_range))  
goat <- df\_spatial\_merged |>  
 mutate(goat\_incidence\_range = ifelse(is.na(goat\_incidence\_range),   
 "0",   
 as.character(goat\_incidence\_range)) %>%  
 factor(., levels = c(  
 "0",  
"[0.0178,0.259]",  
"(0.259,1.09]",  
"(1.09,6.98]" ,  
"(6.98,690]"   
 ))) |>   
 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",  
"[0.0259,0.13]",  
"(0.13,1.46]",  
"(1.46,3.85]",  
"(3.85,45.1]"   
 ))) |>   
 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",  
 "[0.613,0.97]",  
 "(0.97,2.32]",  
 "(2.32,3.36]",  
 "(3.36,122]"   
 ))) |>   
 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

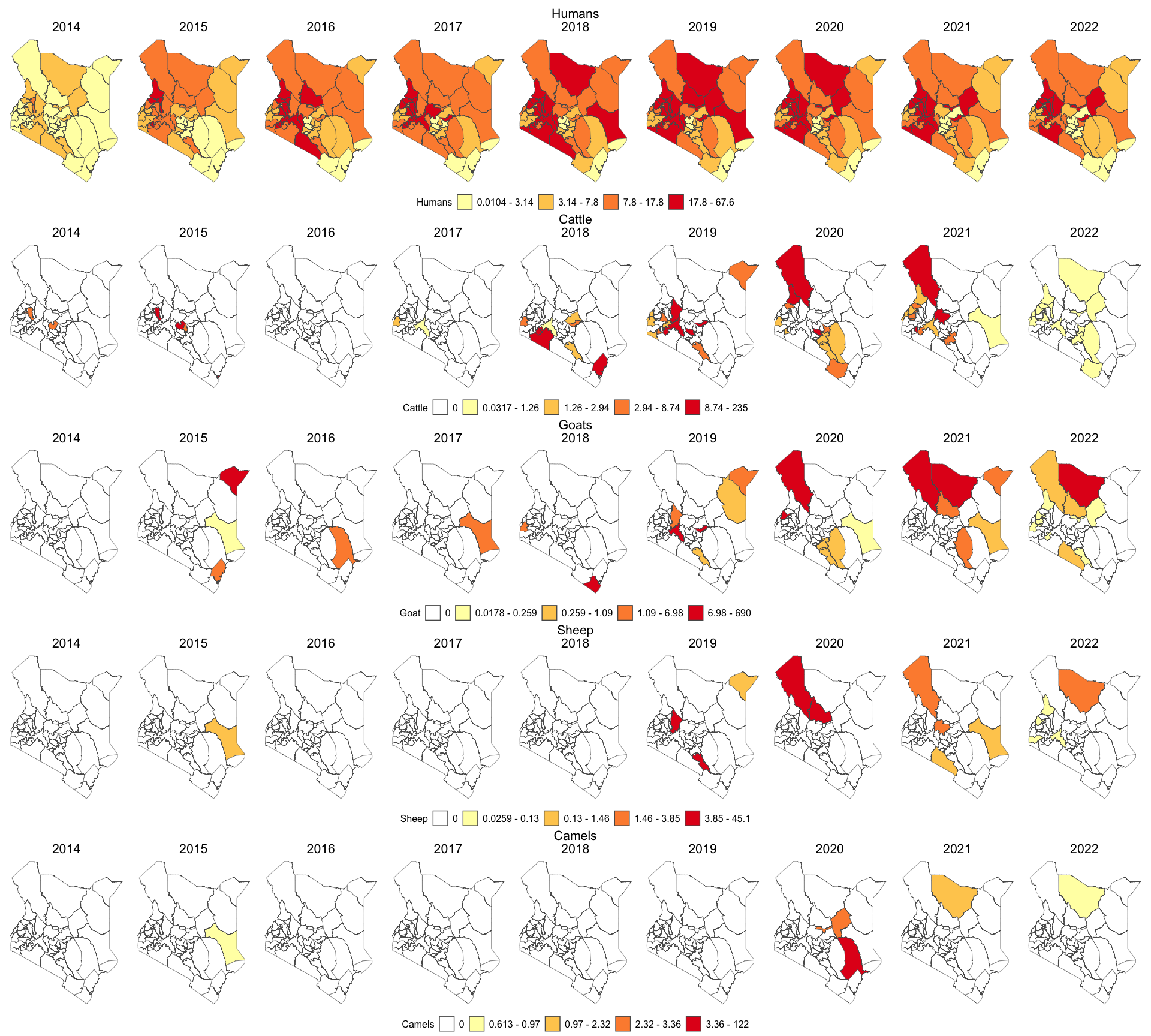


all\_plots <-  
 wrap\_plots(humans,  
 cattle,  
 goat,  
 sheep,  
 camels,  
 ncol = 1,  
 guides = "keep") &  
 theme(plot.caption = element\_text(size = 16, colour = "black"))  
dev.off()

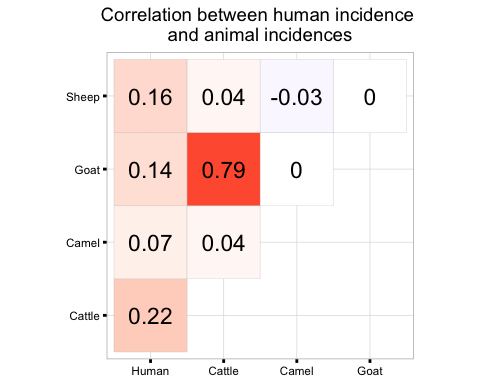
null device   
 1

ggsave("all\_plots.png", dpi = 1e3, height = 18, width = 20)

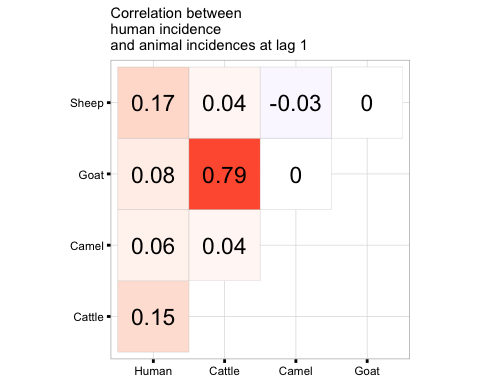
all\_plots



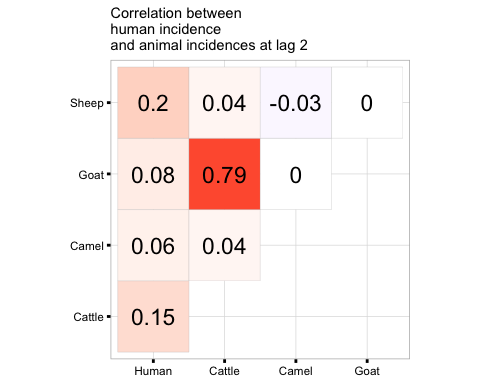
# Correlation Plot  
df\_1\_trend <- df\_1\_trend |>   
 mutate(across(where(is.numeric), ~ifelse(is.na(.), 0, .)))  
  
cor\_lag <- df\_1\_trend %>%  
 as\_tibble() %>%  
 dplyr::select(-date) %>%  
 setNames(c("Human",  
 "Cattle",  
 "Camel",  
 "Goat",  
 "Sheep")) |>  
 cor() %>%  
 ggcorrplot::ggcorrplot(type = "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



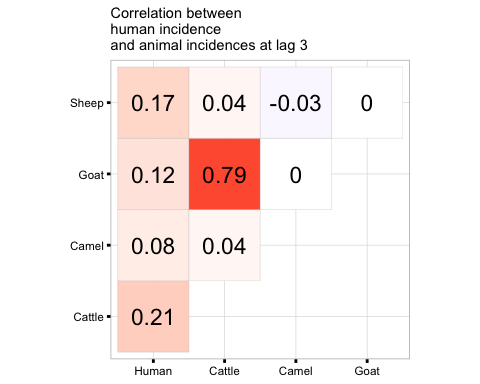
# Correlation Plot at lag 1  
cor\_lag1 <- df\_1\_trend %>%  
 as\_tibble() %>%  
 mutate\_at(vars(catt\_incidence, cam\_incidence, goat\_incidence, shp\_incidence),  
 list( ~ lag(., n = 1))) |>  
 na.omit() |>  
 dplyr::select(-date) %>%  
 setNames(c("Human",  
 "Cattle",  
 "Camel",  
 "Goat",  
 "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



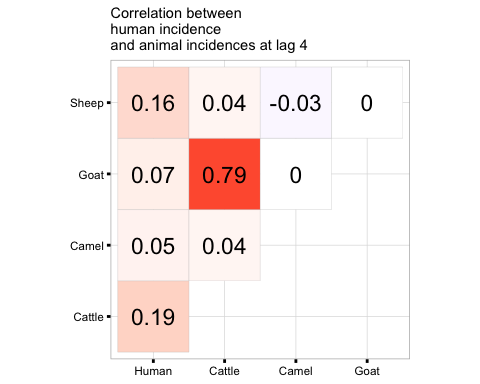
# Correlation Plot at lag 2  
cor\_lag2 <- df\_1\_trend %>%  
 as\_tibble() %>%  
 mutate\_at(vars(catt\_incidence, cam\_incidence, goat\_incidence, shp\_incidence),  
 list( ~ lag(., n = 2))) |>  
 na.omit() |>  
 dplyr::select(-date) %>%  
 setNames(c("Human",  
 "Cattle",  
 "Camel",  
 "Goat",  
 "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



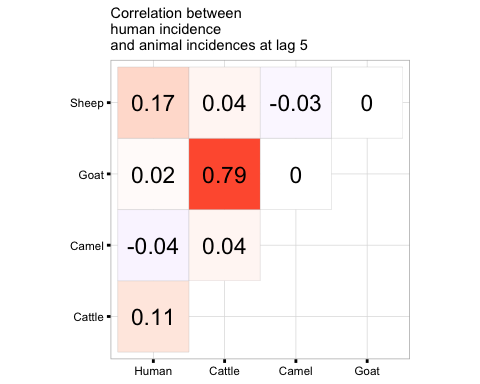
# Correlation Plot at lag 3  
cor\_lag3 <- df\_1\_trend %>%  
 as\_tibble() %>%  
 mutate\_at(vars(catt\_incidence, cam\_incidence, goat\_incidence, shp\_incidence),  
 list( ~ lag(., n = 3))) |>  
 na.omit() |>  
 dplyr::select(-date) %>%  
 setNames(c("Human",  
 "Cattle",  
 "Camel",  
 "Goat",  
 "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



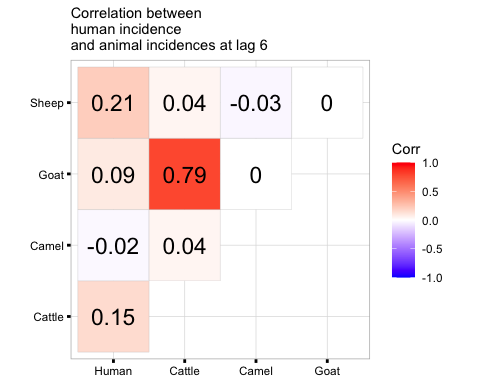
# Correlation Plot at lag 4  
cor\_lag4 <- df\_1\_trend %>%  
 as\_tibble() %>%  
 mutate\_at(vars(catt\_incidence, cam\_incidence, goat\_incidence, shp\_incidence),  
 list( ~ lag(., n = 4))) |>  
 na.omit() |>  
 dplyr::select(-date) %>%  
 setNames(c("Human",  
 "Cattle",  
 "Camel",  
 "Goat",  
 "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() |>  
 dplyr::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() |>  
 dplyr::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



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))

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))  
) |>  
 arrange(desc(`Average Correlation`)) |>   
 knitr::kable(align = "l",   
 caption = "Average correlation between human incidence and other species incidence, ordered from the largest to the smallest",   
 format = "pipe",  
 latex\_options = "hold\_position")  
  
print(result\_table)

Table: Average correlation between human incidence and other species incidence, ordered from the largest to the smallest  
  
|Lag |Average Correlation |  
|:---|:-------------------|  
|0 |0.1475 |  
|3 |0.1450 |  
|2 |0.1225 |  
|4 |0.1175 |  
|1 |0.1150 |  
|6 |0.1075 |  
|5 |0.0650 |

# 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) %>%  
 dplyr::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  
 ) |> dplyr::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) %>%  
 dplyr::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), ~round(., 3))) |>  
 mutate(significance = ifelse(conf\_low \* conf\_high > 0, "Significant", "Not Significant")) |>   
 dplyr::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) %>%  
 dplyr::select(-.model) %>%  
 as\_tibble() %>%  
 mutate(term = case\_when(  
 term == "goat\_incidence" ~ "Goat Incidence",  
 term == "catt\_incidence" ~ "Cattle incidence",  
 TRUE ~ as.character(term)   
 ),  
 variable = term  
 ) |> dplyr::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) %>%  
 dplyr::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), ~round(., 3))) |>  
 mutate(significance = ifelse(conf\_low \* conf\_high > 0, "Significant", "Not Significant")) |>   
 dplyr::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   
-0.7266 -0.1578 0.0911 0.2261 0.5408   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.87918 0.03355 26.209 <2e-16 \*\*\*  
cam\_incidence 0.15911 0.22709 0.701 0.4851   
shp\_incidence 0.97554 0.62776 1.554 0.1233   
catt\_incidence 0.23064 0.13455 1.714 0.0895 .   
goat\_incidence -0.02506 0.05387 -0.465 0.6428   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3179 on 103 degrees of freedom  
Multiple R-squared: 0.0758, Adjusted R-squared: 0.0399  
F-statistic: 2.112 on 4 and 103 DF, p-value: 0.084635  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.74358 -0.11750 0.05283 0.22167 0.47940   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.89177 0.03339 26.707 <2e-16 \*\*\*  
cam\_incidence 0.12606 0.22486 0.561 0.5763   
shp\_incidence 1.07181 0.62166 1.724 0.0877 .   
catt\_incidence 0.17900 0.13329 1.343 0.1823   
goat\_incidence -0.02870 0.05335 -0.538 0.5917   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3148 on 102 degrees of freedom  
Multiple R-squared: 0.05722, Adjusted R-squared: 0.02024  
F-statistic: 1.548 on 4 and 102 DF, p-value: 0.19413  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.73671 -0.12186 0.05724 0.21880 0.51482   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.89661 0.03267 27.445 <2e-16 \*\*\*  
cam\_incidence 0.12965 0.21882 0.593 0.5548   
shp\_incidence 1.22441 0.60500 2.024 0.0456 \*   
catt\_incidence 0.17812 0.12977 1.373 0.1729   
goat\_incidence -0.03005 0.05192 -0.579 0.5641   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3063 on 101 degrees of freedom  
Multiple R-squared: 0.06858, Adjusted R-squared: 0.03169  
F-statistic: 1.859 on 4 and 101 DF, p-value: 0.12348  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.74004 -0.11984 0.05346 0.18815 0.48826   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.89994 0.03194 28.175 <2e-16 \*\*\*  
cam\_incidence 0.16581 0.21277 0.779 0.4376   
shp\_incidence 1.02029 0.58835 1.734 0.0860 .   
catt\_incidence 0.22751 0.12624 1.802 0.0745 .   
goat\_incidence -0.03432 0.05049 -0.680 0.4982   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.2978 on 100 degrees of freedom  
Multiple R-squared: 0.08157, Adjusted R-squared: 0.04483  
F-statistic: 2.22 on 4 and 100 DF, p-value: 0.072126

non\_diff\_indivi\_without\_NA <-  
 complete\_run\_lag\_models(df\_1\_trend\_complete)

Series: human\_incidence   
Model: TSLM   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.84860 -0.11488 0.04294 0.16120 0.41882   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 1.004848 0.043141 23.292 <2e-16 \*\*\*  
catt\_incidence 0.114544 0.118264 0.969 0.338   
goat\_incidence -0.005748 0.045086 -0.127 0.899   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.2622 on 46 degrees of freedom  
Multiple R-squared: 0.04234, Adjusted R-squared: 0.0007068  
F-statistic: 1.017 on 2 and 46 DF, p-value: 0.36967  
Series: human\_incidence   
Model: TSLM   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.88147 -0.12067 0.00746 0.16625 0.38477   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 1.039113 0.040050 25.945 <2e-16 \*\*\*  
catt\_incidence 0.034544 0.108609 0.318 0.752   
goat\_incidence 0.005222 0.041319 0.126 0.900   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.2401 on 45 degrees of freedom  
Multiple R-squared: 0.01066, Adjusted R-squared: -0.03331  
F-statistic: 0.2425 on 2 and 45 DF, p-value: 0.78571  
Series: human\_incidence   
Model: TSLM   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.52295 -0.12363 0.02954 0.14731 0.36752   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 1.055838 0.033949 31.100 <2e-16 \*\*\*  
catt\_incidence 0.038460 0.091059 0.422 0.675   
goat\_incidence 0.007845 0.034565 0.227 0.822   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.2007 on 44 degrees of freedom  
Multiple R-squared: 0.02277, Adjusted R-squared: -0.02165  
F-statistic: 0.5125 on 2 and 44 DF, p-value: 0.60252  
Series: human\_incidence   
Model: TSLM   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.466496 -0.123334 0.008342 0.136314 0.349946   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 1.081553 0.032282 33.503 <2e-16 \*\*\*  
catt\_incidence -0.010042 0.085607 -0.117 0.907   
goat\_incidence 0.005739 0.032421 0.177 0.860   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.188 on 43 degrees of freedom  
Multiple R-squared: 0.0007612, Adjusted R-squared: -0.04572  
F-statistic: 0.01638 on 2 and 43 DF, p-value: 0.98376

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   
-0.386594 -0.066171 0.007478 0.082536 0.249146   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.006222 0.012200 0.510 0.611  
cam\_incidence -18.527691 17.256709 -1.074 0.286  
shp\_incidence 0.116650 0.368165 0.317 0.752  
catt\_incidence 0.039857 0.040692 0.979 0.330  
goat\_incidence -0.057560 0.075005 -0.767 0.445  
  
Residual standard error: 0.1253 on 102 degrees of freedom  
Multiple R-squared: 0.02158, Adjusted R-squared: -0.01679  
F-statistic: 0.5625 on 4 and 102 DF, p-value: 0.69041  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.413173 -0.063748 0.006477 0.076952 0.239656   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.007373 0.011809 0.624 0.53382   
cam\_incidence 65.712533 52.578727 1.250 0.21426   
shp\_incidence -0.375130 0.353663 -1.061 0.29136   
catt\_incidence -0.094868 0.039090 -2.427 0.01700 \*   
goat\_incidence 0.208432 0.072060 2.892 0.00468 \*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.1203 on 101 degrees of freedom  
Multiple R-squared: 0.1062, Adjusted R-squared: 0.07077  
F-statistic: 2.999 on 4 and 101 DF, p-value: 0.021975  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.410457 -0.063757 0.006243 0.083743 0.252240   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.004657 0.012417 0.375 0.708  
cam\_incidence NA NA NA NA  
shp\_incidence -0.211110 0.371877 -0.568 0.572  
catt\_incidence 0.006058 0.041103 0.147 0.883  
goat\_incidence -0.065731 0.075771 -0.867 0.388  
  
Residual standard error: 0.1265 on 101 degrees of freedom  
Multiple R-squared: 0.01143, Adjusted R-squared: -0.01793  
F-statistic: 0.3893 on 3 and 101 DF, p-value: 0.76096  
Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.412447 -0.066177 0.005648 0.082223 0.250031   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.006652 0.012521 0.531 0.596  
cam\_incidence NA NA NA NA  
shp\_incidence 0.389456 0.373215 1.044 0.299  
catt\_incidence 0.025046 0.041243 0.607 0.545  
goat\_incidence -0.018898 0.076028 -0.249 0.804  
  
Residual standard error: 0.1269 on 100 degrees of freedom  
Multiple R-squared: 0.01396, Adjusted R-squared: -0.01563  
F-statistic: 0.4718 on 3 and 100 DF, p-value: 0.70262

## individual animal spp national level ------------------  
write\_csv(non\_diff\_indivi\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared), "non\_diff\_individual\_with\_NA.csv")   
  
  
write\_csv(non\_diff\_indivi\_without\_NA |>  
 dplyr::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  
  
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) %>%  
 dplyr::select(-.model) %>%  
 as\_tibble() %>%  
 mutate(  
 term = case\_when(  
 term == "animal\_incidence" ~ "Animal Incidence",  
 TRUE ~ as.character(term)  
 ),  
 variable = term  
 ) |> dplyr::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) |>  
 dplyr::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, .))))

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.74840 -0.12736 0.07724 0.22423 0.51900   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.90744 0.03163 28.689 <2e-16 \*\*\*  
animal\_incidence 0.11629 0.06767 1.718 0.0886 .   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3215 on 106 degrees of freedom  
Multiple R-squared: 0.0271, Adjusted R-squared: 0.01793  
F-statistic: 2.953 on 1 and 106 DF, p-value: 0.088637

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.75967 -0.13346 0.07628 0.22781 0.49967   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.91847 0.03141 29.243 <2e-16 \*\*\*  
animal\_incidence 0.07279 0.06689 1.088 0.279   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3177 on 105 degrees of freedom  
Multiple R-squared: 0.01115, Adjusted R-squared: 0.001734  
F-statistic: 1.184 on 1 and 105 DF, p-value: 0.27901

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.76573 -0.13419 0.06972 0.21866 0.49909   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.92563 0.03090 29.959 <2e-16 \*\*\*  
animal\_incidence 0.07000 0.06549 1.069 0.288   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3111 on 104 degrees of freedom  
Multiple R-squared: 0.01087, Adjusted R-squared: 0.001356  
F-statistic: 1.143 on 1 and 104 DF, p-value: 0.28759

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.76995 -0.13787 0.07195 0.21685 0.47979   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.92985 0.03024 30.750 <2e-16 \*\*\*  
animal\_incidence 0.09533 0.06380 1.494 0.138   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.303 on 103 degrees of freedom  
Multiple R-squared: 0.02122, Adjusted R-squared: 0.01171  
F-statistic: 2.233 on 1 and 103 DF, p-value: 0.13817

non\_diff\_full\_without\_NA <- lag\_models\_full(df = df\_cum\_trend)

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.74840 -0.12736 0.07724 0.22423 0.51900   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.90744 0.03163 28.689 <2e-16 \*\*\*  
animal\_incidence 0.11629 0.06767 1.718 0.0886 .   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3215 on 106 degrees of freedom  
Multiple R-squared: 0.0271, Adjusted R-squared: 0.01793  
F-statistic: 2.953 on 1 and 106 DF, p-value: 0.088637

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.75967 -0.13346 0.07628 0.22781 0.49967   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.91847 0.03141 29.243 <2e-16 \*\*\*  
animal\_incidence 0.07279 0.06689 1.088 0.279   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3177 on 105 degrees of freedom  
Multiple R-squared: 0.01115, Adjusted R-squared: 0.001734  
F-statistic: 1.184 on 1 and 105 DF, p-value: 0.27901

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.76573 -0.13419 0.06972 0.21866 0.49909   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.92563 0.03090 29.959 <2e-16 \*\*\*  
animal\_incidence 0.07000 0.06549 1.069 0.288   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.3111 on 104 degrees of freedom  
Multiple R-squared: 0.01087, Adjusted R-squared: 0.001356  
F-statistic: 1.143 on 1 and 104 DF, p-value: 0.28759

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.76995 -0.13787 0.07195 0.21685 0.47979   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.92985 0.03024 30.750 <2e-16 \*\*\*  
animal\_incidence 0.09533 0.06380 1.494 0.138   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.303 on 103 degrees of freedom  
Multiple R-squared: 0.02122, Adjusted R-squared: 0.01171  
F-statistic: 2.233 on 1 and 103 DF, p-value: 0.13817

diff\_full <- lag\_models\_full(df\_cum\_trend\_diff)

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.4087348 -0.0635705 -0.0008561 0.0813921 0.2519912   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 0.005663 0.011879 0.477 0.6346   
animal\_incidence 0.033347 0.018293 1.823 0.0712 .  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 0.1229 on 105 degrees of freedom  
Multiple R-squared: 0.03068, Adjusted R-squared: 0.02145  
F-statistic: 3.323 on 1 and 105 DF, p-value: 0.071158

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.408009 -0.073618 0.007568 0.083920 0.251737   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.00568 0.01211 0.469 0.640  
animal\_incidence -0.02037 0.01856 -1.097 0.275  
  
Residual standard error: 0.1247 on 104 degrees of freedom  
Multiple R-squared: 0.01144, Adjusted R-squared: 0.001939  
F-statistic: 1.204 on 1 and 104 DF, p-value: 0.27505

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.411473 -0.066265 0.008535 0.084835 0.251244   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.005465 0.012262 0.446 0.657  
animal\_incidence -0.014059 0.018706 -0.752 0.454  
  
Residual standard error: 0.1257 on 103 degrees of freedom  
Multiple R-squared: 0.005454, Adjusted R-squared: -0.004202  
F-statistic: 0.5648 on 1 and 103 DF, p-value: 0.45403

Using `date` as index variable.

Series: human\_incidence   
Model: TSLM   
Transformation: (human\_incidence)   
  
Residuals:  
 Min 1Q Median 3Q Max   
-0.411774 -0.068145 0.006532 0.083605 0.250940   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)  
(Intercept) 0.005768 0.012279 0.47 0.640  
animal\_incidence 0.027778 0.018641 1.49 0.139  
  
Residual standard error: 0.1252 on 102 degrees of freedom  
Multiple R-squared: 0.02131, Adjusted R-squared: 0.01171  
F-statistic: 2.221 on 1 and 102 DF, p-value: 0.13927

## individual animal spp national level ---------  
write.csv(non\_diff\_full\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared), "non\_diff\_full\_with\_NA.csv")  
  
write.csv(non\_diff\_full\_without\_NA |>  
 dplyr::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 |>   
 dplyr::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 as been arranged in decreasing order of Adjusted R-Squared",  
 format = "pipe",  
 latex\_options = "hold\_position"  
 )  
Table\_lag\_indivi\_withoutNA

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 |
| --- | --- | --- | --- |
| 0 | 4.23 | 0.1 | -126.28 |
| 2 | 2.28 | -2.2 | -146.08 |
| 1 | 1.07 | -3.3 | -132.06 |
| 3 | 0.08 | -4.6 | -148.84 |

Table\_lag\_full <- non\_diff\_full\_without\_NA |>   
 dplyr::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 been arranged in decreasing order of Adjusted R-Squared",  
 format = "pipe",  
 latex\_options = "hold\_position"  
 )  
Table\_lag\_full

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 | 2.71 | 1.8 | -241.11 |
| 3 | 2.12 | 1.2 | -246.80 |
| 1 | 1.12 | 0.2 | -241.37 |
| 2 | 1.09 | 0.1 | -243.58 |

# At lag 0  
lag0\_forest\_df\_lag0 <- non\_diff\_indivi\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared) %>%   
 filter(lag == 0 & variable != '(Intercept)')  
  
lag0\_forest\_plot\_lag0 <- lag0\_forest\_df\_lag0 %>%   
 ggplot(aes(x = variable, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2, color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 0",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 )+  
 ggtitle('Individual animal incidence at Lag 0')  
  
  
  
  
# At lag 3  
lag3\_forest\_df\_lag3 <- non\_diff\_indivi\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared) %>%   
 filter(lag == 3 & variable != '(Intercept)')  
  
lag3\_forest\_plot\_lag3 <- lag3\_forest\_df\_lag3 %>%   
 ggplot(aes(x = variable, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2, color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 3",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 ) +  
 ggtitle('Individual animal incidence at Lag 3')  
  
  
all\_forest\_individual <- lag0\_forest\_plot\_lag0 +  
lag3\_forest\_plot\_lag3  
  
dev.off()

null device   
 1

ggsave(  
 "images/all\_forest\_individual.png",  
 width = 13,  
 height = 6,  
 dpi = 1e3  
)

# At lag 0  
lag0\_forest\_df1\_lag0 <- non\_diff\_full\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared) %>%   
 filter(lag == 0 & variable != '(Intercept)')  
  
lag0\_forest\_plot1\_lag0 <- lag0\_forest\_df1\_lag0 %>%   
 ggplot(aes(x = variable, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2, color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 0",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 )+  
 ggtitle('Combined animal incidence at Lag 0')  
  
  
  
  
# At lag 3  
lag3\_forest\_df1\_lag3 <- non\_diff\_full\_with\_NA |>  
 dplyr::select(-r\_squared, -AIC, -adj\_r\_squared) %>%   
 filter(lag == 3 & variable != '(Intercept)')  
  
lag3\_forest\_plot1\_lag3 <- lag3\_forest\_df1\_lag3 %>%   
 ggplot(aes(x = variable, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2, color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 3",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 ) +  
 ggtitle('Combined animal incidence at Lag 3')  
  
  
all\_forest\_comb <- lag0\_forest\_plot1\_lag0 +  
lag3\_forest\_plot1\_lag3  
  
dev.off()

null device   
 1

ggsave(  
 "images/all\_forest\_full.png",  
 width = 13,  
 height = 6,  
 dpi = 1e3  
)  
  
  
# One plot  
all\_forests <- all\_forest\_individual/all\_forest\_comb  
  
dev.off()

null device   
 1

ggsave(  
 "images/all\_forest\_full.png",  
 width = 18,  
 height = 10,  
 dpi = 1e3  
)

# Lag 0  
df\_2\_0 <- df\_1 |>   
 as\_tibble() %>%  
 mutate\_at(vars(catt\_incidence, cam\_incidence, goat\_incidence, shp\_incidence),  
 list( ~ dplyr::lag(., n = 0))) |>  
 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() |>  
 dplyr::select(-.model) |>  
 as\_tibble() |>  
 mutate(term = case\_when(  
 term == "animal\_incidence" ~ "Animal Incidence",  
 TRUE ~ as.character(term)   
 ),  
 variable = term  
 ) |>   
 dplyr::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() |>  
 dplyr::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  
 ) |>   
 dplyr::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\_0$county)  
  
# Create a list to store the models for each county  
models\_list <- list()  
  
# Initialize the data frame for each county  
coefficients\_df\_lag0 <- data.frame(county = character(),   
 variable = character(),  
 estimate = numeric(),  
 stringsAsFactors = FALSE)  
  
# At lag 0  
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\_0, 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\_lag0 <- bind\_rows(coefficients\_df\_lag0,   
 mod\_county %>%   
 mutate(county = county\_name,  
 estimate = round(estimate, 3)))  
 }  
}

Fitting model for Baringo

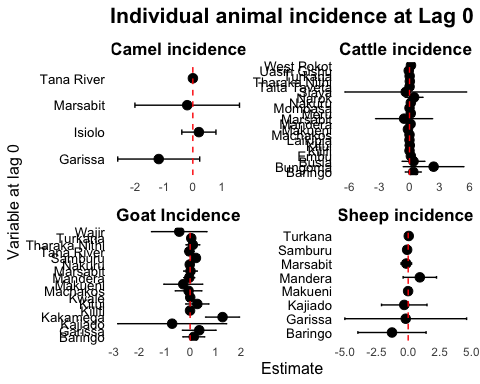
Using `date` as index variable.  
Fitting model for Bomet  
  
Using `date` as index variable.  
Fitting model for Bungoma  
  
Using `date` as index variable.  
Fitting model for Busia  
  
Using `date` as index variable.  
Fitting model for Elgeyo Marakwet  
  
Using `date` as index variable.  
Fitting model for Embu  
  
Using `date` as index variable.  
Fitting model for Garissa  
  
Using `date` as index variable.  
Fitting model for Homa Bay  
  
Using `date` as index variable.  
Fitting model for Isiolo  
  
Using `date` as index variable.  
Fitting model for Kajiado  
  
Using `date` as index variable.  
Fitting model for Kakamega  
  
Using `date` as index variable.  
Fitting model for Kericho  
  
Skipping model for Kericho as all incidences are zero.  
  
Fitting model for Kiambu  
  
Skipping model for Kiambu as all incidences are zero.  
  
Fitting model for Kilifi  
  
Using `date` as index variable.  
Fitting model for Kirinyaga  
  
Skipping model for Kirinyaga as all incidences are zero.  
  
Fitting model for Kisii  
  
Skipping model for Kisii as all incidences are zero.  
  
Fitting model for Kisumu  
  
Skipping model for Kisumu as all incidences are zero.  
  
Fitting model for Kitui  
  
Using `date` as index variable.  
Fitting model for Kwale  
  
Using `date` as index variable.  
Fitting model for Laikipia  
  
Using `date` as index variable.  
Fitting model for Lamu  
  
Skipping model for Lamu as all incidences are zero.  
  
Fitting model for Machakos  
  
Using `date` as index variable.  
Fitting model for Makueni  
  
Using `date` as index variable.  
Fitting model for Mandera  
  
Using `date` as index variable.  
Fitting model for Marsabit  
  
Using `date` as index variable.  
Fitting model for Meru  
  
Using `date` as index variable.  
Fitting model for Migori  
  
Skipping model for Migori as all incidences are zero.  
  
Fitting model for Mombasa  
  
Using `date` as index variable.  
Fitting model for Murang'a  
  
Skipping model for Murang'a as all incidences are zero.  
  
Fitting model for Nairobi  
  
Skipping model for Nairobi as all incidences are zero.  
  
Fitting model for Nakuru  
  
Using `date` as index variable.  
Fitting model for Nandi  
  
Using `date` as index variable.  
Fitting model for Narok  
  
Using `date` as index variable.  
Fitting model for Nyamira  
  
Skipping model for Nyamira as all incidences are zero.  
  
Fitting model for Nyandarua  
  
Skipping model for Nyandarua as all incidences are zero.  
  
Fitting model for Nyeri  
  
Skipping model for Nyeri as all incidences are zero.  
  
Fitting model for Samburu  
  
Using `date` as index variable.  
Fitting model for Siaya  
  
Using `date` as index variable.  
Fitting model for Taita Taveta  
  
Using `date` as index variable.  
Fitting model for Tana River  
  
Using `date` as index variable.  
Fitting model for Tharaka Nithi  
  
Using `date` as index variable.  
Fitting model for Trans Nzoia  
  
Using `date` as index variable.  
Fitting model for Turkana  
  
Using `date` as index variable.  
Fitting model for Uasin Gishu  
  
Using `date` as index variable.  
Fitting model for Vihiga  
  
Using `date` as index variable.  
Fitting model for Wajir  
  
Using `date` as index variable.  
Fitting model for West Pokot  
  
Using `date` as index variable.

coefficients\_df\_lag0 <- coefficients\_df\_lag0 |>  
 mutate(significant = ifelse(conf\_low \* conf\_high > 0, "Significant", "Not Significant")) |>   
 mutate(across(c(3:8), ~round(., 3)))  
  
  
lago\_county\_significant\_individual <- coefficients\_df\_lag0 %>%   
 filter(significant == 'Significant' & variable != '(Intercept)')   
  
write\_csv(lago\_county\_significant\_individual, "lago\_county\_significant\_individual.csv")  
  
write\_csv(coefficients\_df\_lag0, "individual\_animal\_incidence\_per\_county\_lag0.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)))  
 }  
}

Fitting model for Baringo  
Using `date` as index variable.Fitting model for Bomet  
Using `date` as index variable.Fitting model for Bungoma  
Using `date` as index variable.Fitting model for Busia  
Using `date` as index variable.Fitting model for Elgeyo Marakwet  
Using `date` as index variable.Fitting model for Embu  
Using `date` as index variable.Fitting model for Garissa  
Using `date` as index variable.Fitting model for Homa Bay  
Skipping model for Homa Bay as all incidences are zero.  
Fitting model for Isiolo  
Using `date` as index variable.Fitting model for Kajiado  
Using `date` as index variable.Fitting model for Kakamega  
Using `date` as index variable.Fitting model for Kericho  
Skipping model for Kericho as all incidences are zero.  
Fitting model for Kiambu  
Skipping model for Kiambu as all incidences are zero.  
Fitting model for Kilifi  
Using `date` as index variable.Fitting model for Kirinyaga  
Skipping model for Kirinyaga as all incidences are zero.  
Fitting model for Kisii  
Skipping model for Kisii as all incidences are zero.  
Fitting model for Kisumu  
Skipping model for Kisumu as all incidences are zero.  
Fitting model for Kitui  
Using `date` as index variable.Fitting model for Kwale  
Using `date` as index variable.Fitting model for Laikipia  
Using `date` as index variable.Fitting model for Lamu  
Skipping model for Lamu as all incidences are zero.  
Fitting model for Machakos  
Using `date` as index variable.Fitting model for Makueni  
Using `date` as index variable.Fitting model for Mandera  
Using `date` as index variable.Fitting model for Marsabit  
Using `date` as index variable.Fitting model for Meru  
Using `date` as index variable.Fitting model for Migori  
Skipping model for Migori as all incidences are zero.  
Fitting model for Mombasa  
Using `date` as index variable.Fitting model for Murang'a  
Skipping model for Murang'a as all incidences are zero.  
Fitting model for Nairobi  
Skipping model for Nairobi as all incidences are zero.  
Fitting model for Nakuru  
Using `date` as index variable.Fitting model for Nandi  
Using `date` as index variable.Fitting model for Narok  
Using `date` as index variable.Fitting model for Nyamira  
Skipping model for Nyamira as all incidences are zero.  
Fitting model for Nyandarua  
Skipping model for Nyandarua as all incidences are zero.  
Fitting model for Nyeri  
Skipping model for Nyeri as all incidences are zero.  
Fitting model for Samburu  
Using `date` as index variable.Fitting model for Siaya  
Using `date` as index variable.Fitting model for Taita Taveta  
Using `date` as index variable.Fitting model for Tana River  
Using `date` as index variable.Fitting model for Tharaka Nithi  
Using `date` as index variable.Fitting model for Trans Nzoia  
Using `date` as index variable.Fitting model for Turkana  
Using `date` as index variable.Fitting model for Uasin Gishu  
Using `date` as index variable.Fitting model for Vihiga  
Using `date` as index variable.Fitting model for Wajir  
Using `date` as index variable.Fitting model for West Pokot  
Using `date` as index variable.

coefficients\_df\_lag3 <- coefficients\_df\_lag3 |>  
 mutate(significant = ifelse(conf\_low \* conf\_high > 0, "Significant", "Not Significant")) |>   
 mutate(across(c(3:8), ~round(., 3)))  
  
lag3\_county\_significant\_individual <- coefficients\_df\_lag3 %>%   
 filter(significant == 'Significant' & variable != '(Intercept)')   
  
write\_csv(lag3\_county\_significant\_individual, "lag3\_county\_significant\_individual.csv")  
  
write\_csv(coefficients\_df\_lag3, "individual\_animal\_incidence\_per\_county\_lag3.csv")

# At lag 0  
indivi\_lag0\_df <- coefficients\_df\_lag0 %>%   
 filter( variable != '(Intercept)' & !is.nan(conf\_low) & !is.na(conf\_low)) %>%   
 filter(conf\_high < 10 & conf\_low > - 10)  
  
  
indivi\_lag0\_plot <- indivi\_lag0\_df %>%   
 #filter(!county %in% c('West Pokot', 'Nandi', 'Elgeyo Marakwet')) %>%   
 ggplot(aes(x = county, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 facet\_wrap(~variable, scales = 'free') +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 0",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none",  
 strip.text = element\_text(color = 'black', hjust = .5, face = 'bold', size = 13)  
 ) +  
 ggtitle('Individual animal incidence at Lag 0')  
indivi\_lag0\_plot



dev.off()

null device   
 1

ggsave(  
 "images/indivi\_lag0\_forest.png",  
 width = 18,  
 height = 10,  
 dpi = 1e3  
)  
  
  
  
# At lag 3  
indivi\_lag3\_df <- coefficients\_df\_lag3 %>%   
 filter( variable != '(Intercept)' & !is.nan(conf\_low) & !is.na(conf\_low)) %>%   
 filter(conf\_high < 10 & conf\_low > - 10)  
  
  
indivi\_lag3\_plot <- indivi\_lag3\_df %>%   
 ggplot(aes(x = county, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 facet\_wrap(~variable, scales = 'free') +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 3",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none",  
 strip.text = element\_text(color = 'black', hjust = .5, face = 'bold', size = 13)  
 ) +  
 ggtitle('Individual animal incidence at Lag 3')  
indivi\_lag3\_plot  
  
dev.off()

null device   
 1

ggsave(  
 "images/indivi\_lag3\_forest.png",  
 width = 18,  
 height = 10,  
 dpi = 1e3  
)

# Lag 2  
df\_cum\_2\_0 <- df\_cum |>   
 as\_tibble() %>%  
 mutate\_at(vars(animal\_incidence),  
 list( ~ dplyr::lag(., n = 0))) |>  
 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 0  
county\_names2\_0 <- unique(df\_cum\_2\_0$county)  
  
# Create a list to store the models for each county  
models\_list <- list()  
  
# Initialize the data frame for each county  
coefficients\_df2\_lag0 <- data.frame(county = character(),   
 variable = character(),  
 estimate = numeric(),  
 stringsAsFactors = FALSE)  
  
for (county\_name in county\_names2\_0) {  
 message(paste("Fitting model for", county\_name))  
   
   
   
 # Fit the model  
 mod\_county <- fit\_county\_model(county\_name, df\_cum\_2\_0, 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\_lag0 <- bind\_rows(coefficients\_df2\_lag0,   
 mod\_county %>%   
 mutate(county = county\_name,  
 estimate = round(estimate, 3)))  
 }  
}

Fitting model for Uasin Gishu

Using `date` as index variable.  
Fitting model for Nyeri  
  
Using `date` as index variable.  
Fitting model for Kirinyaga  
  
Using `date` as index variable.  
Fitting model for Garissa  
  
Using `date` as index variable.  
Fitting model for Mandera  
  
Using `date` as index variable.  
Fitting model for Mombasa  
  
Using `date` as index variable.  
Fitting model for Kilifi  
  
Using `date` as index variable.  
Fitting model for Tana River  
  
Using `date` as index variable.  
Fitting model for Siaya  
  
Using `date` as index variable.  
Fitting model for Nakuru  
  
Using `date` as index variable.  
Fitting model for Bomet  
  
Using `date` as index variable.  
Fitting model for Kwale  
  
Using `date` as index variable.  
Fitting model for Narok  
  
Using `date` as index variable.  
Fitting model for Makueni  
  
Using `date` as index variable.  
Fitting model for Tharaka Nithi  
  
Using `date` as index variable.  
Fitting model for Meru  
  
Using `date` as index variable.  
Fitting model for Wajir  
  
Using `date` as index variable.  
Fitting model for Baringo  
  
Using `date` as index variable.  
Fitting model for Kericho  
  
Using `date` as index variable.  
Fitting model for Nandi  
  
Using `date` as index variable.  
Fitting model for Kisii  
  
Using `date` as index variable.  
Fitting model for Migori  
  
Using `date` as index variable.  
Fitting model for Murang'a  
  
Using `date` as index variable.  
Fitting model for Busia  
  
Using `date` as index variable.  
Fitting model for Isiolo  
  
Using `date` as index variable.  
Fitting model for Bungoma  
  
Using `date` as index variable.  
Fitting model for Turkana  
  
Using `date` as index variable.  
Fitting model for Embu  
  
Using `date` as index variable.  
Fitting model for Machakos  
  
Using `date` as index variable.  
Fitting model for Samburu  
  
Using `date` as index variable.  
Fitting model for Kitui  
  
Using `date` as index variable.  
Fitting model for Taita Taveta  
  
Using `date` as index variable.  
Fitting model for Trans Nzoia  
  
Using `date` as index variable.  
Fitting model for West Pokot  
  
Using `date` as index variable.  
Fitting model for Kakamega  
  
Using `date` as index variable.  
Fitting model for Nyamira  
  
Using `date` as index variable.  
Fitting model for Vihiga  
  
Using `date` as index variable.  
Fitting model for Laikipia  
  
Using `date` as index variable.  
Fitting model for Kajiado  
  
Using `date` as index variable.  
Fitting model for Marsabit  
  
Using `date` as index variable.  
Fitting model for Elgeyo Marakwet  
  
Using `date` as index variable.  
Fitting model for Homa Bay  
  
Using `date` as index variable.

coefficients\_df2\_lag0 <- coefficients\_df2\_lag0 |>  
 mutate(across(c(3:8), ~round(., 3))) |>   
 mutate(significant = ifelse(conf\_low \* conf\_high > 0, "Significant", "Not Significant")) |>   
 as\_tibble()  
  
lag0\_county\_significant\_combined <- coefficients\_df2\_lag0 %>%   
 filter(significant == 'Significant' & variable != '(Intercept)')   
  
write\_csv(lag0\_county\_significant\_combined,  
 "lag0\_county\_significant\_combined.csv")  
  
  
write\_csv(coefficients\_df2\_lag0, "all\_animal\_incidence\_per\_county\_lag0.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)))  
 }  
}

Fitting model for Tharaka Nithi  
Using `date` as index variable.Fitting model for Turkana  
Using `date` as index variable.Fitting model for Trans Nzoia  
Using `date` as index variable.Fitting model for Nyamira  
Using `date` as index variable.Fitting model for Vihiga  
Using `date` as index variable.Fitting model for Bungoma  
Using `date` as index variable.Fitting model for Marsabit  
Using `date` as index variable.Fitting model for Embu  
Using `date` as index variable.Fitting model for Isiolo  
Using `date` as index variable.

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()  
  
lag3\_county\_significant\_combined <- coefficients\_df2\_lag3 %>%   
 filter(significant == 'Significant' & variable != '(Intercept)')   
  
write\_csv(lag3\_county\_significant\_combined,  
 "lag3\_county\_significant\_combined.csv")  
  
write\_csv(coefficients\_df2\_lag3, "all\_animal\_incidence\_per\_county\_lag3.csv")

# At lag 0  
comb\_lag0\_df <- coefficients\_df2\_lag0 %>%   
 filter( variable != '(Intercept)' & !is.nan(conf\_low) & !is.na(conf\_low))   
  
  
comb\_lag0\_plot <- comb\_lag0\_df %>%   
 filter(county != 'Isiolo') %>%   
 ggplot(aes(x = county, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 0",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 ) +  
 ggtitle('Combined animal incidence at Lag 0')  
  
# At lag 3  
comb\_lag3\_df <- coefficients\_df2\_lag3 %>%   
 filter( variable != '(Intercept)' & !is.nan(conf\_low) & !is.na(conf\_low))   
  
  
comb\_lag3\_plot <- comb\_lag3\_df %>%   
 filter(county != 'Isiolo') %>%   
 ggplot(aes(x = county, y = estimate)) +  
 geom\_point(size = 3) +  
 geom\_errorbar(aes(ymin = conf\_low, ymax = conf\_high), width = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +   
 coord\_flip() +  
 labs(  
 #title = "Incidence Rate Ratios (IRRs) and Confidence Intervals",  
 x = "Variable at lag 3",  
 y = "Estimate"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.y = element\_text(size = 10, color = 'black'),  
 axis.title.x = element\_text(size = 12, color = 'black'),  
 axis.title.y = element\_text(size = 12),  
 plot.title = element\_text(size = 16, hjust = 0.5, face = "bold"),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank(),  
 panel.border = element\_blank(),  
 panel.background = element\_blank(),  
 legend.position = "none"  
 ) +  
 ggtitle('Combined animal incidence at Lag 3')  
  
combined\_forest <- comb\_lag0\_plot + comb\_lag3\_plot  
  
  
dev.off()

null device   
 1

ggsave(  
 "images/combined\_forest.png",  
 width = 18,  
 height = 10,  
 dpi = 1e3  
)

# Lag 2  
df\_2\_0\_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() |>  
 dplyr::select(-.model) |>  
 as\_tibble() |>  
 mutate(term = case\_when(  
 term == "animal\_incidence" ~ "Animal Incidence",  
 TRUE ~ as.character(term)   
 ),  
 variable = term  
 ) |>   
 dplyr::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() |>  
 dplyr::select(-.model) |>  
 as\_tibble() |>  
 mutate(term = case\_when(  
 term == "goat\_incidence" ~ "Goat Incidence",  
 term == "catt\_incidence" ~ "Cattle incidence",  
 TRUE ~ as.character(term)   
 ),  
 variable = term  
 ) |>   
 dplyr::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\_0\_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)  
  
# 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\_0\_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)))  
 }  
}

Fitting model for Baringo

Using `date` as index variable.  
Fitting model for Bomet  
  
Using `date` as index variable.  
Fitting model for Bungoma  
  
Using `date` as index variable.  
Fitting model for Busia  
  
Using `date` as index variable.  
Fitting model for Elgeyo Marakwet  
  
Using `date` as index variable.  
Fitting model for Embu  
  
Using `date` as index variable.  
Fitting model for Garissa  
  
Using `date` as index variable.  
Fitting model for Isiolo  
  
Using `date` as index variable.  
Fitting model for Kakamega  
  
Using `date` as index variable.  
Fitting model for Kericho  
  
Using `date` as index variable.  
Fitting model for Kilifi  
  
Using `date` as index variable.  
Fitting model for Kirinyaga  
  
Using `date` as index variable.  
Fitting model for Kisii  
  
Using `date` as index variable.  
Fitting model for Kitui  
  
Using `date` as index variable.  
Fitting model for Laikipia  
  
Using `date` as index variable.  
Fitting model for Machakos  
  
Using `date` as index variable.  
Fitting model for Makueni  
  
Using `date` as index variable.  
Fitting model for Mandera  
  
Using `date` as index variable.  
Fitting model for Marsabit  
  
Using `date` as index variable.  
Fitting model for Meru  
  
Using `date` as index variable.  
Fitting model for Migori  
  
Using `date` as index variable.  
Fitting model for Mombasa  
  
Using `date` as index variable.  
Fitting model for Murang'a  
  
Using `date` as index variable.  
Fitting model for Nakuru  
  
Using `date` as index variable.  
Fitting model for Nandi  
  
Using `date` as index variable.  
Fitting model for Narok  
  
Using `date` as index variable.  
Fitting model for Nyamira  
  
Using `date` as index variable.  
Fitting model for Nyeri  
  
Using `date` as index variable.  
Fitting model for Siaya  
  
Using `date` as index variable.  
Fitting model for Taita Taveta  
  
Using `date` as index variable.  
Fitting model for Tharaka Nithi  
  
Using `date` as index variable.  
Fitting model for Trans Nzoia  
  
Using `date` as index variable.  
Fitting model for Turkana  
  
Using `date` as index variable.  
Fitting model for Uasin Gishu  
  
Using `date` as index variable.  
Fitting model for Vihiga  
  
Using `date` as index variable.  
Fitting model for West Pokot  
  
Using `date` as index variable.

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)))  
 }  
}

Fitting model for Baringo  
Skipping model for Baringo as all incidences are zero.  
Fitting model for Bomet  
Using `date` as index variable.Fitting model for Bungoma  
Using `date` as index variable.Fitting model for Busia  
Using `date` as index variable.Fitting model for Elgeyo Marakwet  
Using `date` as index variable.Fitting model for Embu  
Using `date` as index variable.Fitting model for Garissa  
Using `date` as index variable.Fitting model for Isiolo  
Using `date` as index variable.Fitting model for Kakamega  
Using `date` as index variable.Fitting model for Kericho  
Using `date` as index variable.Fitting model for Kilifi  
Using `date` as index variable.Fitting model for Kirinyaga  
Using `date` as index variable.Fitting model for Kisii  
Using `date` as index variable.Fitting model for Kitui  
Using `date` as index variable.Fitting model for Laikipia  
Using `date` as index variable.Fitting model for Machakos  
Using `date` as index variable.Fitting model for Makueni  
Using `date` as index variable.Fitting model for Mandera  
Using `date` as index variable.Fitting model for Marsabit  
Using `date` as index variable.Fitting model for Meru  
Using `date` as index variable.Fitting model for Migori  
Using `date` as index variable.Fitting model for Mombasa  
Using `date` as index variable.Fitting model for Murang'a  
Using `date` as index variable.Fitting model for Nakuru  
Using `date` as index variable.Fitting model for Nandi  
Using `date` as index variable.Fitting model for Narok  
Using `date` as index variable.Fitting model for Nyamira  
Using `date` as index variable.Fitting model for Nyeri  
Using `date` as index variable.Fitting model for Siaya  
Using `date` as index variable.Fitting model for Taita Taveta  
Using `date` as index variable.Fitting model for Tharaka Nithi  
Using `date` as index variable.Fitting model for Trans Nzoia  
Using `date` as index variable.Fitting model for Turkana  
Using `date` as index variable.Fitting model for Uasin Gishu  
Using `date` as index variable.Fitting model for Vihiga  
Using `date` as index variable.Fitting model for West Pokot  
Using `date` as index variable.

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)) |>   
 dplyr::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) |>   
 dplyr::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 |>   
 dplyr::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)  
 ) |>   
 dplyr::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  
 ) |>   
 dplyr::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 |>   
 dplyr::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) {  
 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)  
 ) |>   
 dplyr::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) %>%  
 dplyr::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)  
 ) |>   
 dplyr::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) %>%  
 dplyr::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)

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 0

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 1

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 2

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

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)  
 ) |>   
 dplyr::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) %>%  
 dplyr::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)  
 ) |>   
 dplyr::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) %>%  
 dplyr::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)

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 0

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 1

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 2

Warning in eval(family$initialize, rho): non-integer #successes in a binomial  
glm!

Runnning model for lag 3

# Time Series Analysis

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

source('clean\_tseries\_data.R')

Warning: Expecting numeric in I18 / R18C9: got a date

Warning: Expecting numeric in L18 / R18C12: got a date

Warning: Expecting numeric in I19 / R19C9: got a date

Warning: Expecting numeric in L19 / R19C12: got a date

Warning: Expecting numeric in I85 / R85C9: got a date

New names:  
Rows: 282 Columns: 3  
── Column specification  
──────────────────────────────────────────────────────── Delimiter: "," chr  
(2): County, species dbl (1): species\_num  
ℹ Use `spec()` to retrieve the full column specification for this data. ℹ  
Specify the column types or set `show\_col\_types = FALSE` to quiet this message.  
`summarise()` has grouped output by 'date'. You can override using the  
`.groups` argument.  
`summarise()` has grouped output by 'date'. You can override using the  
`.groups` argument.  
• `year` -> `year...10`  
• `year` -> `year...13`

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)

Warning in adf.test(df\_complete1$animal\_incidence): p-value smaller than  
printed p-value

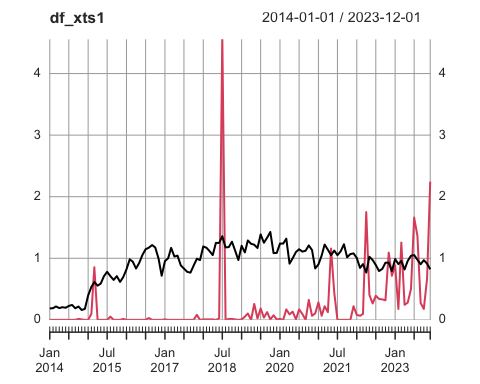
Augmented Dickey-Fuller Test  
  
data: df\_complete1$animal\_incidence  
Dickey-Fuller = -4.0428, Lag order = 4, p-value = 0.01  
alternative hypothesis: stationary

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, 0)  
 ) |>  
 na.omit() |>   
 distinct(date, .keep\_all = T)  
adf.test(df2.1$human\_incidence)

Augmented Dickey-Fuller Test  
  
data: df2.1$human\_incidence  
Dickey-Fuller = -1.8568, Lag order = 4, p-value = 0.6361  
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

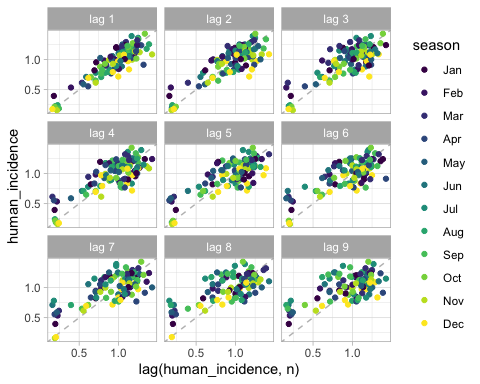
## Testing for several aspects of the time series

### 1. Correlation of human incidence with it’s past values

# Correlation between human incidence and the previous lags  
dd = df\_xts |>  
 data.frame(row.names = NULL) %>%  
 mutate(date = zoo::index(df\_xts) |>  
 zoo::as.yearmon() |>  
 yearmonth()) |>  
 as\_tsibble()

Using `date` as index variable.

dd |>  
 gg\_lag(human\_incidence, geom = "point") +  
 theme\_light()



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

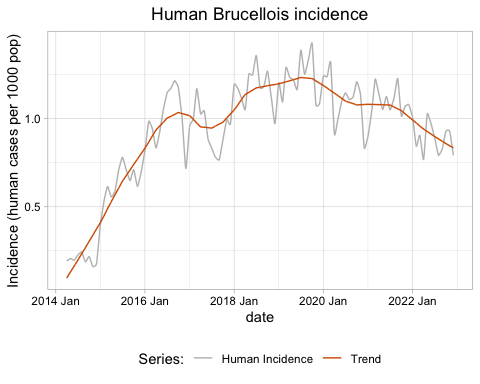
### 3. Decomposing the time series

dcmp <- dd |>  
 model(stl = STL(human\_incidence))

Below is the trend and the time series

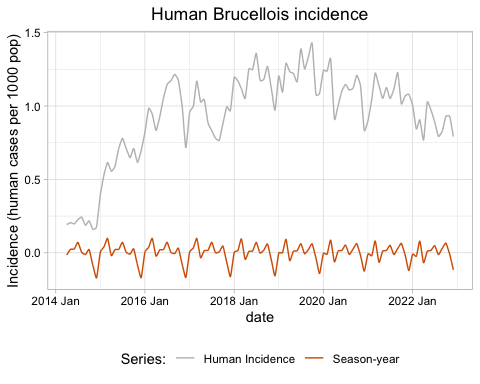
# The trend  
components(dcmp) |>  
 as\_tsibble() |>  
 ggplot(aes(x = date)) +  
 geom\_xspline(aes(y = human\_incidence, colour = "Human Incidence")) +  
 geom\_xspline(aes(y = trend, color = 'Trend')) +  
 labs(y = "Incidence (human cases per 1000 pop)",  
 title = "Human Brucellois incidence") +  
 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:')

Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2 3.4.0.  
ℹ Please use `linewidth` in the `default\_aes` field and elsewhere instead.



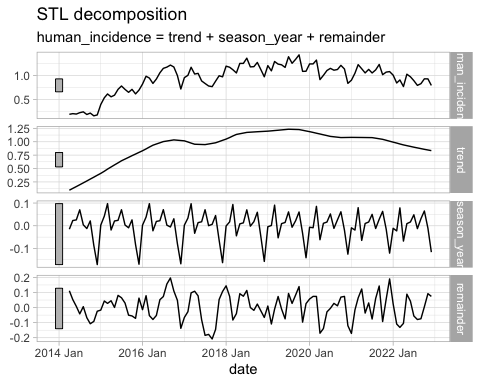
The time series seasonality

# 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:')



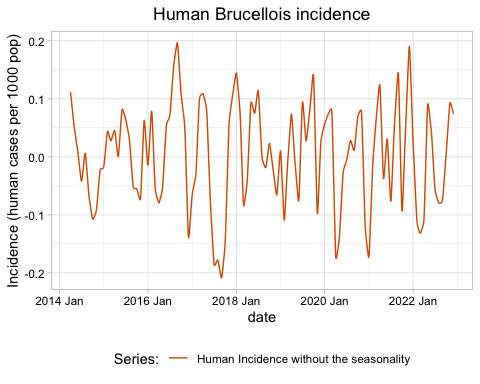
All of them together

# 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" = "#D55E00")) +  
 labs(color = 'Series:')



## Train and testing data

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

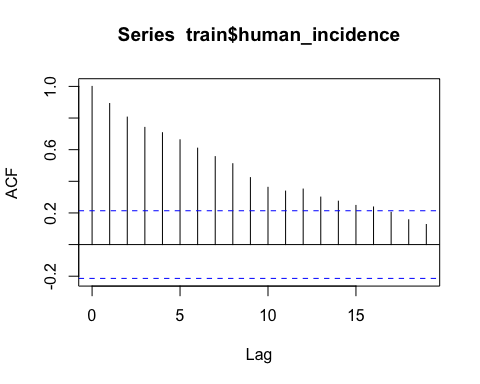
# Splitting ------------------------------------------------------------------------------  
  
# train <- window(ts\_diff, start = as.Date("2014-04-01"), end = "2021-12-01")  
# test <- window(df\_xts, start = as.Date("2022-01-01"), end = as.Date("2022-12-01"))  
  
# We will take 80% for training, 20% for validation.  
nrow(df\_xts) # our data has 105 rows, thus 80% of this is 84 rows for training and 21 for testing.,

[1] 105

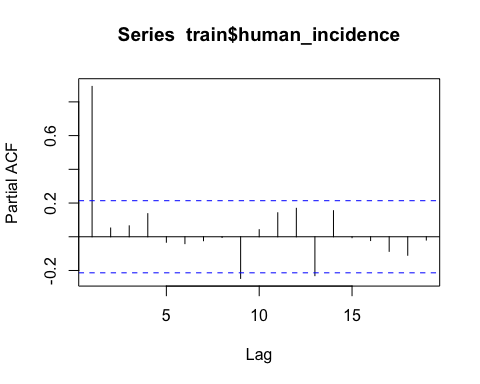
# 0% is upto 2021-03-01 and then the rest for testing  
nrow\_80 <- floor(.8 \* nrow(df\_xts))  
train <- df\_xts[1:nrow\_80,]  
test <- df\_xts[(nrow\_80 + 1):nrow(df\_xts),]

The acf and pacf plot were as follows;

# ACF and PACF plots ---------------------------------------------------------------------  
  
acf(train$human\_incidence)



#> 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()

Using `date` as index variable.

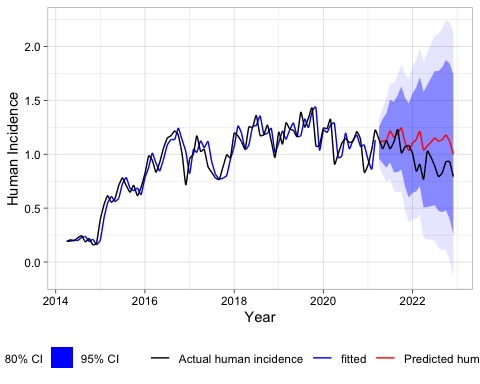
Series: human\_incidence   
Model: ARIMA(0,1,0)(2,0,0)[12]   
  
Coefficients:  
 sar1 sar2  
 0.2471 0.3418  
s.e. 0.1041 0.1158  
  
sigma^2 estimated as 0.01293: log likelihood=61.28  
AIC=-116.56 AICc=-116.25 BIC=-109.3

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() %>%  
 dplyr::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() |>   
 dplyr::select(-V3) |>   
 setNames(c("Lo80", "Hi80"))  
  
  
CI95 <- CI$`95%` |>   
 lapply(FUN = first) |> unlist() |>  
 matrix(ncol = 3, byrow = T) |>   
 as.data.frame() |>   
 dplyr::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"), alpha = 0.1) +  
 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"), alpha = .4) +  
 geom\_xspline(data = forecast\_data\_human\_sarima, aes(date, PointForecast, colour = "Predicted human incidence")) +  
 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() |>  
 dplyr::select(-animal\_incidence)

Using `date` as index variable.

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 × 15  
 .model .type ME RMSE MAE MPE MAPE MASE RMSSE ACF1 winkler pinball  
 <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 "ARIMA… Test -0.150 0.194 0.160 -17.0 18.0 NaN NaN 0.445 1.47 0.0765  
# ℹ 3 more variables: scaled\_pinball <dbl>, 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()

Using `date` as index variable.

Series: human\_incidence   
Model: LM w/ ARIMA(0,1,0)(2,0,0)[12] errors   
  
Coefficients:  
 sar1 sar2 animal\_incidence  
 0.2286 0.3516 0.0219  
s.e. 0.1045 0.1159 0.0160  
  
sigma^2 estimated as 0.01281: log likelihood=62.21  
AIC=-116.42 AICc=-115.9 BIC=-106.74

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

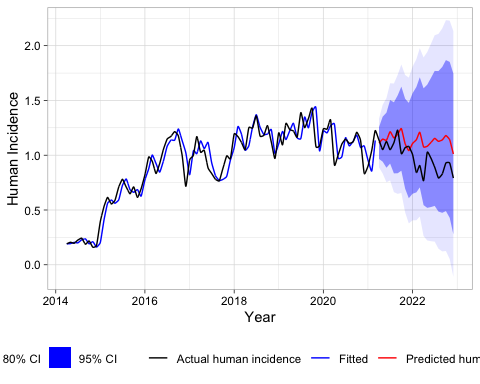
fitted\_covariate <- augment(model\_sarima\_covariates) |>   
 as.data.frame() %>%  
 dplyr::select(date, fitted =.fitted, residuals = .resid) |>   
 mutate(date = ym(date)   
 )   
  
test\_tsibble <- test |>   
 as.data.frame() %>%  
 mutate(date = rownames(.) |> ymd() |> yearmonth()) |>   
 as\_tsibble() |>  
 dplyr::select(-human\_incidence)

Using `date` as index variable.

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() |>   
 dplyr::select(-V3) |>   
 setNames(c("Lo80", "Hi80"))  
  
  
CI\_covariate95 <- CI\_covariate$`95%` |>   
 lapply(FUN = first) |> unlist() |>  
 matrix(ncol = 3, byrow = T) |>   
 as.data.frame() |>   
 dplyr::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"), alpha = 0.1) +  
 geom\_ribbon(data = forecast\_data\_covariate\_sarima, aes(date, ymin = Lo80, ymax = Hi80, fill = "80% CI"), alpha = .4) +  
 geom\_xspline(data = fitted\_covariate, aes(date, y = fitted, color = "Fitted")) +  
 geom\_xspline(data = forecast\_data\_covariate\_sarima, aes(date, PointForecast, colour = "Predicted human incidence")) +  
 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) |>  
 dplyr::select(-.model) %>%  
 as\_tibble() %>%  
 group\_by(term)   
  
  
# Accuracy  
test\_acc2 <- test |>  
 as.data.frame() %>%  
 mutate(date = rownames(.) |> ymd() |> yearmonth()) |>  
 as\_tsibble() |>  
 dplyr::select(-animal\_incidence)

Using `date` as index variable.

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 × 15  
 .model .type ME RMSE MAE MPE MAPE MASE RMSSE ACF1 winkler pinball  
 <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 "ARIMA… Test -0.156 0.200 0.166 -17.8 18.6 NaN NaN 0.437 1.45 0.0767  
# ℹ 3 more variables: scaled\_pinball <dbl>, 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") |>   
 dplyr::select(-.model, -.type) |>   
 dplyr::select(model, RMSE, MAE, MAPE)  
  
model\_sarima\_covariate\_accuracy2 <- model\_sarima\_covariate\_accuracy |>   
 mutate(model = "Model with Covariate") |>   
 dplyr::select(-.model, -.type) |>   
 dplyr::select(model, RMSE, MAE, MAPE)  
  
metrics\_df <- rbind(model\_sarima\_accuracy2, model\_sarima\_covariate\_accuracy2)  
  
kableExtra::kable(metrics\_df)

| model | RMSE | MAE | MAPE |
| --- | --- | --- | --- |
| Model without Covariate | 0.1939374 | 0.1604385 | 17.97294 |
| Model with Covariate | 0.1998684 | 0.1658678 | 18.60326 |

write.csv(metrics\_df, "metrics\_df\_lag0.csv", row.names = F)  
  
  
#> We have saved the metrics df for several training set 70-30, 75-25, 80-20

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;

# 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()

Using `date` as index variable.

Series: human\_incidence   
Model: LM w/ ARIMA(0,1,2)(2,0,0)[12] errors   
  
Coefficients:  
 ma1 ma2 sar1 sar2 animal\_incidence  
 -0.1758 -0.2066 0.1464 0.2872 0.0155  
s.e. 0.0963 0.0939 0.0957 0.0977 0.0167  
  
sigma^2 estimated as 0.01319: log likelihood=78.69  
AIC=-145.38 AICc=-144.52 BIC=-129.52

Then, the forecasted data is curated as follows;

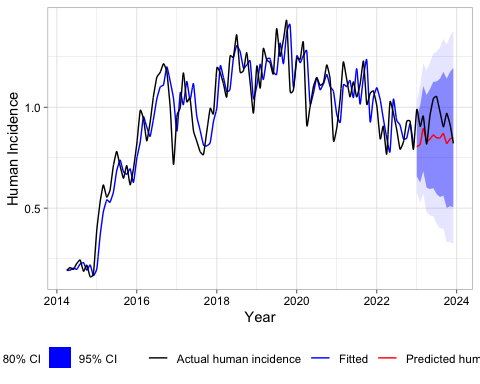
fitted\_full <- augment(full.model) |>   
 as.data.frame() %>%  
 dplyr::select(date, fitted =.fitted, residuals = .resid) |>   
 mutate(date = ym(date)   
 )   
  
original\_data2 <- df\_xts |>  
 rbind(forecast.df) |>   
 as.data.frame() %>%  
 mutate(date = row.names(.) |> ymd())  
  
test\_tsibble <- forecast.df |>   
 as.data.frame() %>%  
 mutate(date = rownames(.) |> ymd() |> yearmonth()) |>   
 as\_tsibble() |>  
 dplyr::select(-human\_incidence)

Using `date` as index variable.

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() |>   
 dplyr::select(-V3) |>   
 setNames(c("Lo80", "Hi80"))  
  
  
CI\_full95 <- CI\_full$`95%` |>   
 lapply(FUN = first) |> unlist() |>  
 matrix(ncol = 3, byrow = T) |>   
 as.data.frame() |>   
 dplyr::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"), alpha = 0.1) +  
 geom\_ribbon(data = forecast\_data\_full\_sarima, aes(date, ymin = Lo80, ymax = Hi80, fill = "80% CI"), alpha = .4) +  
 geom\_xspline(data = fitted\_full, aes(date, y = fitted, color = "Fitted")) +  
 geom\_xspline(data = forecast\_data\_full\_sarima, aes(date, PointForecast, colour = "Predicted human incidence")) +  
 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) |>  
 dplyr::select(-.model) %>%  
 as\_tibble() %>%  
 group\_by(term)  
knitr::kable(full.model\_coefficients)

| term | estimate | std.error | statistic | p.value |
| --- | --- | --- | --- | --- |
| ma1 | -0.1757870 | 0.0962505 | -1.8263488 | 0.0706664 |
| ma2 | -0.2065749 | 0.0938788 | -2.2004416 | 0.0299877 |
| sar1 | 0.1463870 | 0.0956684 | 1.5301508 | 0.1290155 |
| sar2 | 0.2872090 | 0.0977332 | 2.9387052 | 0.0040604 |
| animal\_incidence | 0.0155322 | 0.0167198 | 0.9289659 | 0.3550579 |

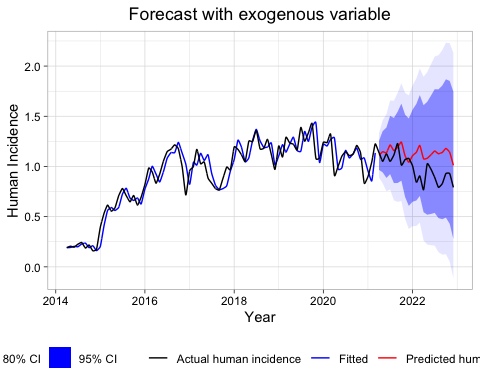
The results of the forecasted values versus the actual, together with the upper 95% CI were as follows;

# 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()) |>   
 dplyr::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 | 0.804 | 0.988 | 0.579 | 1.029 |
| 2023 Feb | 0.818 | 0.905 | 0.526 | 1.110 |
| 2023 Mar | 0.894 | 0.955 | 0.571 | 1.217 |
| 2023 Apr | 0.830 | 0.818 | 0.478 | 1.182 |
| 2023 May | 0.842 | 0.958 | 0.464 | 1.220 |
| 2023 Jun | 0.862 | 1.041 | 0.459 | 1.265 |
| 2023 Jul | 0.849 | 1.052 | 0.422 | 1.275 |
| 2023 Aug | 0.848 | 0.977 | 0.400 | 1.297 |
| 2023 Sep | 0.869 | 0.903 | 0.399 | 1.338 |
| 2023 Oct | 0.821 | 0.969 | 0.332 | 1.311 |
| 2023 Nov | 0.842 | 0.913 | 0.333 | 1.351 |
| 2023 Dec | 0.850 | 0.821 | 0.322 | 1.377 |

## Supplementary Plots

# Supplementary plots --------------------------------------------------------------------  
  
  
forecast\_plot\_covariate\_sarima\_title <- forecast\_plot\_covariate\_sarima +  
 ggtitle("Forecast with exogenous variable")  
forecast\_plot\_covariate\_sarima\_title



dev.off()

null device   
 1

ggsave(  
 "images/forecast\_plot\_covariate\_sarima\_title.png",  
 width = 13,  
 height = 6,  
 dpi = 1e3  
)  
  
require(patchwork)  
all <- forecast\_plot\_human\_sarima/forecast\_plot\_covariate\_sarima  
all  
dev.off()

null device   
 1

ggsave(  
 "images/all\_without\_title\_sarima.png",  
 width = 13,  
 height = 10,  
 dpi = 1e3  
)  
  
forecast\_plot\_human\_sarima\_title <- forecast\_plot\_human\_sarima +  
 ggtitle("Forecast without exogenous variable")  
forecast\_plot\_human\_sarima\_title  
all\_title <- forecast\_plot\_human\_sarima\_title/forecast\_plot\_covariate\_sarima\_title  
all\_title  
dev.off()

null device   
 1

ggsave(  
 "images/all\_with\_title\_sarima.png",  
 width = 13,  
 height = 10,  
 dpi = 1e3  
)