R Script

The following R Script was used for the calcutations in this research paper:

# Load Packages --------------------------------------------------------------------------------------------------  
  
library(tidyverse)  
library(sf)  
library(tmap)  
library(ggplot2)  
  
?tmap  
  
# Import data -----------------------------------------------------------------------------------------------------  
  
# geographical data  
# download from https://ec.europa.eu/eurostat/web/gisco/geodata/statistical-units/territorial-units-statistics  
# NUTS 2010  
  
sf\_nuts <- read\_sf("data/NUTS/NUTS\_RG\_20M\_2010\_3035.shp/NUTS\_RG\_20M\_2010\_3035.shp")  
  
# survey data  
# Download from https://ess.sikt.no/en/  
  
ESS8 <- read\_csv("data/ESS/ESS8e02\_3/ESS8e02\_3.csv")  
ESS10 <- bind\_rows(read\_csv("data/ESS/ESS10/ESS10.csv"), read\_csv("data/ESS/ESS10SC/ESS10SC.csv"))  
  
# flood data  
# download from https://essd.copernicus.org/articles/16/5145/2024/essd-16-5145-2024-assets.html  
  
# Prepare Geographical Data ---------------------------------------------------------------------------------------  
  
# prepare a dataset with geographical information of europe  
  
nuts <-   
 sf\_nuts |>   
 rename(region = NUTS\_ID) |>   
 dplyr::select(region, CNTR\_CODE, LEVL\_CODE, NAME\_LATN, geometry)  
  
colnames(sf\_nuts)  
class(sf\_nuts)  
  
  
country\_polygons <- nuts |> filter(LEVL\_CODE == 0)  
  
# Prepare survey data ---------------------------------------------------------------------------------------------  
  
# variables  
# impenv, importance to care for environment  
# wrclmch, how worried about climate change  
# ccnthum, CC caused naturally or by humans  
# ccrdprs, personal responsability to reduce CC  
  
# prepare a dataset with survey participants from wave eight and ten  
  
ESS\_prepared <-   
 bind\_rows(ESS10, ESS8) |>   
 filter(essround == 8 | essround == 10) |>   
 filter(cntry %in% ESS8$cntry, # make sure that the country in wave 10 is also in wave 8  
 cntry %in% ESS10$cntry, # make sure that the country in wave 8 is also in wave 10  
 cntry %in% c("GB", "ES", "CH", "AT") # only include countries with sufficient data  
 ) |>   
 mutate(  
 date8 = as.Date(paste(inwdds, inwmms, inwyys, sep = "-"), format = "%d-%m-%Y"),  
 date10 = inwds  
 ) |>  
 mutate(  
 date = if\_else(essround == 8, date8, date10) |> as.Date(),  
 cntry = if\_else(cntry == "GB", "UK", cntry)  
 ) |>  
 left\_join(y = nuts, by = "region") |>  
 dplyr::select(idno, essround, date,   
 cntry, region, LEVL\_CODE, NAME\_LATN, # geographical information  
 wrclmch, # outcome variable  
 lrscale, ccnthum, # control variables  
 geometry) |>  
 mutate(  
 region = if\_else(region == "99999", NA, region),  
 wrclmch = if\_else(wrclmch > 5, NA, wrclmch),  
 lrscale = if\_else(lrscale > 10, NA, lrscale),  
 ccntum = if\_else(ccnthum > 5, NA, ccnthum)  
 ) |>  
 drop\_na(idno, essround, cntry, region) |>  
 mutate(respondent\_id = row\_number()) |>   
 dplyr::select(respondent\_id, essround, date,   
 cntry, region, LEVL\_CODE, NAME\_LATN, # geographical information  
 wrclmch, # outcome variable  
 lrscale, ccnthum, # control variables  
 geometry) |>   
 st\_as\_sf()  
  
df\_survey\_dates <- ESS\_prepared |>  
 as\_tibble() |>  
 group\_by(essround) |>  
 summarise(min\_date = min(date, na.rm = TRUE),  
 mean\_date = mean(date, na.rm = TRUE),  
 max\_date = max(date, na.rm = TRUE))  
  
df\_survey\_dates |> knitr::kable()  
  
# Prepare Floods Data ---------------------------------------------------------------------------------------------  
  
# prepare a dataset of floods from 2016 until 2019 in the areas where people were surveyed  
  
floods\_prepared <-   
 floods\_regions |>   
 mutate(end\_date = make\_date(End\_Y, End\_M, End\_D)) |>   
 filter(between(as.Date(end\_date, format = "%Y-%m-%d"),   
 as.Date("2017-06-18"), # date of the last survey in wave 8  
 as.Date("2021-05-05")), # date of the first survey in wave 10  
 Code %in% ESS\_prepared$cntry # only floods in countries that were surveyed  
 ) |>  
 rename(  
 country = Code,  
 flood\_id = ID,  
 regions = Region2010,  
 references = References  
 ) |>   
 dplyr::select(flood\_id, end\_date, country, regions, references)  
  
  
# Visualize Survey Regions and Floods ---------------------------------------------------------------------------------------------------  
  
ESS\_prepared\_regions <-   
 ESS\_prepared |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
map <- tm\_basemap("OpenStreetMap") +  
 tm\_shape(floods\_prepared$geometry) +  
 tm\_polygons(fill = "blue", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(ESS\_prepared\_regions$geometry) +  
 tm\_polygons(fill = "red", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1)  
  
tmap\_mode("view")  
print(map)  
  
  
# Spatial Merge ---------------------------------------------------------------------------------------------------  
  
sf\_ESS\_prepared\_grouped <-   
 ESS\_prepared |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
containment\_check\_old <-  
 st\_intersects(sf\_ESS\_prepared\_grouped, floods\_prepared)  
  
# Shrink each polygon by 10 kilometers (to avoid overlaps to neighbouring regions)  
sf\_ESS\_shrunk <- st\_buffer(sf\_ESS\_prepared\_grouped, dist = -10000)  
  
containment\_check <-   
 st\_intersects(sf\_ESS\_shrunk, floods\_prepared)  
  
ESS\_intersects <-   
 st\_join(sf\_ESS\_shrunk, floods\_prepared, join = st\_intersects) |>   
 distinct(region, .keep\_all = TRUE) |> # to flag a region as flooded only one flood is necessary  
 as\_tibble()  
  
ESS\_prepared\_with\_flood\_info <-   
 left\_join(ESS\_prepared, ESS\_intersects, by = "region") |>   
 mutate(flood = if\_else(!is.na(flood\_id), 1, 0)) |>   
 dplyr::select(respondent\_id, essround, date, cntry, region, LEVL\_CODE, NAME\_LATN,   
 wrclmch, lrscale, ccnthum, geometry.x, flood) |>   
 rename(geometry = geometry.x)  
  
  
  
# Visualize Treatment Group & Control Group -----------------------------------------------------------------------  
  
treatment\_region\_before <-   
 ESS\_prepared\_with\_flood\_info |>   
 filter(flood == 1,  
 essround == 8) |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
treatment\_region\_after <-   
 ESS\_prepared\_with\_flood\_info |>   
 filter(flood == 1,  
 essround == 10) |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
control\_region\_before <-   
 ESS\_prepared\_with\_flood\_info |>   
 filter(flood == 0,  
 essround == 8) |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
control\_region\_after <-   
 ESS\_prepared\_with\_flood\_info |>   
 filter(flood == 0,  
 essround == 10) |>   
 group\_by(region, geometry) |>   
 summarise(count = n())  
  
map <- tm\_basemap("OpenStreetMap") +  
 tm\_shape(floods\_prepared$geometry) +  
 tm\_polygons(fill = "blue", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(treatment\_region\_before$geometry) +  
 tm\_polygons(fill = "red", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(treatment\_region\_after$geometry) +  
 tm\_polygons(fill = "green", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(control\_region\_before$geometry) +  
 tm\_polygons(fill = "purple", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(control\_region\_after$geometry) +  
 tm\_polygons(fill = "yellow", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1) +  
 tm\_shape(country\_polygons$geometry) +  
 tm\_polygons(fill = "orange", fill\_alpha = 0.5) +  
 tm\_borders(col = "black", lwd = 1)  
  
tmap\_mode("plot")  
print(map)  
  
  
# Code the Treatment Variable & Outcome Variable -------------------------------------------------------------------  
  
data <-   
 ESS\_prepared\_with\_flood\_info |>   
 as\_tibble() |>   
 mutate(time = if\_else(essround == 8, 0, 1)) |>   
 rename(  
 treatment\_variable = flood,  
 outcome\_variable = wrclmch  
 # control\_variable = lrscale  
 ) |>   
 mutate(  
 outcome\_variable = factor(outcome\_variable, ordered = TRUE)  
 # control\_variable = factor(control\_variable, ordered = TRUE)  
 )  
  
# Check Pre-Treatment Characteristics -----------------------------------------------------------------------------  
  
  
pre\_treatment\_data <- data |>  
 filter(time == 0) # Keep only rows from the pre-treatment period  
  
ggplot(pre\_treatment\_data, aes(x = factor(treatment\_variable), y = outcome\_variable)) +  
 geom\_boxplot() +  
 labs(x = "Treatment Group", y = "Outcome Variable", title = "Pre-Treatment Distribution")  
ggplot(pre\_treatment\_data, aes(x = outcome\_variable, fill = factor(treatment\_variable))) +  
 geom\_density(alpha = 0.5) +  
 labs(x = "Outcome Variable", fill = "Treatment Group", title = "Pre-Treatment Density Plot")  
  
  
  
# Calculate the DiD Interaction Term ------------------------------------------------------------------------------  
  
data <- data %>%  
 mutate(interaction = time \* treatment\_variable)  
  
  
# Run the Regression ----------------------------------------------------------------------------------------------  
  
# Logit Model  
  
  
  
logit\_model <- glm(outcome\_variable ~ time + treatment\_variable + interaction,   
 data = data,   
 family = "binomial")  
summary(logit\_model)  
  
library(stargazer)  
library(webshot)  
  
# Save regression table as an HTML file  
html\_file <- "output/logit\_regression\_table.html"  
stargazer(logit\_model, type = "html", out = html\_file, column.labels = "Climate Change Concern",  
 covariate.labels = c("Time", "Treatment", "Time \* Treatment"))  
  
# Convert HTML to PNG  
png\_file <- "output/logit\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
# Ordered Logit Model  
  
library(MASS)  
  
ordered\_logit\_model <- polr(outcome\_variable ~ time + treatment\_variable + interaction,   
 data = data,   
 method = "logistic")  
summary(ordered\_logit\_model)  
  
# Save regression table as an HTML file  
html\_file <- "output/ordered\_logit\_regression\_table.html"  
stargazer(ordered\_logit\_model, type = "html", out = html\_file, column.labels = "Climate Change Concern",  
 covariate.labels = c("Time", "Treatment", "Time \* Treatment"))  
  
# Convert HTML to PNG  
png\_file <- "output/ordered\_logit\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
  
# Robustness Checks -----------------------------------------------------------------------------------------------  
  
# 1. Multinomial Logit Model  
  
library(nnet)  
mnl\_model <- multinom(outcome\_variable ~ time + treatment\_variable + interaction, data = data)  
summary(mnl\_model)  
  
# Save regression table as an HTML file  
html\_file <- "output/multinom\_regression\_table.html"  
stargazer(mnl\_model, type = "html", out = html\_file)  
  
# Convert HTML to PNG  
png\_file <- "output/multinom\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
# 2. Parallel Trends Assumption Check  
  
library(MASS)  
  
# Subset only pre-treatment data (time = 0)  
pre\_treatment\_data <- subset(data, time == 0)  
  
# Run an ordered logit model to test for pre-treatment differences  
placebo\_model <- polr(outcome\_variable ~ treatment\_variable, data = pre\_treatment\_data, method = "logistic")  
  
# Show results  
summary(placebo\_model)  
  
# Save regression table as an HTML file  
html\_file <- "output/placebo\_model\_regression\_table.html"  
stargazer(placebo\_model, type = "html", out = html\_file)  
  
# Convert HTML to PNG  
png\_file <- "output/placebo\_model\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
  
  
# 3. Heterogeneous Treatment Effects  
  
heterogeneity\_model1 <- glm(outcome\_variable ~ time + treatment\_variable + interaction \* lrscale,   
 family = "binomial", data = data)  
summary(heterogeneity\_model1)  
  
# Save regression table as an HTML file  
html\_file <- "output/heterogeneity\_model1\_regression\_table.html"  
stargazer(heterogeneity\_model1, type = "html", out = html\_file)  
  
# Convert HTML to PNG  
png\_file <- "output/heterogeneity\_model1\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
heterogeneity\_model2 <- glm(outcome\_variable ~ time + treatment\_variable + interaction \* ccnthum,   
 family = "binomial", data = data)  
summary(heterogeneity\_model2)  
  
# Save regression table as an HTML file  
html\_file <- "output/heterogeneity\_model2\_regression\_table.html"  
stargazer(heterogeneity\_model2, type = "html", out = html\_file)  
  
# Convert HTML to PNG  
png\_file <- "output/heterogeneity\_model2\_regression\_table.png"  
webshot(html\_file, png\_file)  
  
# 4. Visualization of Treatment Effects  
  
# Outcome distribution before vs. after treatment  
ggplot(data, aes(x = outcome\_variable, fill = as.factor(time))) +  
 geom\_bar(position = "dodge") +  
 labs(title = "Outcome Distribution Before & After Treatment",  
 fill = "Time") + # Renaming legend title  
 scale\_fill\_manual(values = c("0" = "blue", "1" = "red"),   
 labels = c("0" = "Before", "1" = "After")) + # Updating legend labels  
 facet\_wrap(facets = ~treatment\_variable,   
 labeller = as\_labeller(c("0" = "Control Group", "1" = "Treatment Group"))) # Updating facet labels  
  
  
ggsave("output/outcome\_distribution\_before\_and\_after.png")