```
# Load tidyverse for read_csv
library(tidyverse)
library(raster)
library(exactextractr)
library(stringr)
library(readr)
library(fixest)
library(ncdf4)
library(sf)
library(terra)
library(stars)
library(exactextractr)
library(purrr)

library(dplyr)
library(tidyr)



# These are the data sets I will be using - groundwater recharge data
tn_gw_2020 <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford
Classes and Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/1ef56cc6-cbf1-4389-a111-acedb9aa7481.csv')
tn_gw_2022 <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford
Classes and Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/631bf771-9bcb-400c-954b-2cb6f5fbe149.csv')
tn_gw_2023 <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford
Classes and Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/ae6c22ab-5cdc-457e-b67e-1dcb37dfb9e8.csv')
tn_gw_2024 <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford
Classes and Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/b604e42e-ef13-40a2-a12d-a3a862fdd929.csv')

#area, productivity, and yield reports
apy_report <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/EEcon -
final project/horizontal_crop_vertical_year_report.xls.csv', show_col_types = FALSE)
apy_report_top7 <- read_csv('/Users/velsenthil/Library/CloudStorage/OneDrive-
Stanford/EEcon - final project/horizontal_crop_vertical_year_report - final.xls.csv')


#I will read the Tamil Nadu GeoJSON and NetCDF files later and integrate them into the
district year panel

######################################################################
# cleaning groundwater data
######################################################################


# 1. Groundwater spelling -> APY spelling
gw_name_map <- c(
  "Kallakkurichi"   = "Kallakurichi",
  "Kallakurichchi"  = "Kallakurichi",
  "Kancheepuram"    = "Kanchipuram",
  "Tenkasi"         = "Thenkasi",
  "Thoothukkudi"    = "Thoothukudi",
  "Viluppuram"      = "Villupuram",
  "Sivagangai"      = "Sivaganga",
  "Tiruvallur"      = "Thiruvallur"
)

# 2. Columns that are in MCM (all except district & % fields)
mcm_cols <- c(
  "Monsoon season recharge from rainfall",
  "Monsoon season recharge from other sources",
```

```
  "Non-monsoon season recharge from rainfall",
  "Non-monsoon season recharge from other sources",
  "Total annual groundwater recharge",
  "Total Natural Discharges",
  "Annual Extractable Groundwater Resource",
  "Irrigation - Annual extraction",
  "Industrial - Annual extraction",
  "Domestic - Annual Extraction",
  "Total Annual Extraction",
  "Annual GW Allocation for domestic use (2025)",
  "Net GW availability for future"
)

# 3. Function to clean groundwater datasets
clean_gw <- function(df) {

  # remove last 2 summary rows FIRST
  df <- df %>% slice(1:(n() - 2))

  # rename district to APY spelling
  df <- df %>%
    mutate(
      `Name of District` = `Name of District` %>%
        str_squish() %>%
        str_to_title() %>%
        recode(!!!gw_name_map)
    )

  # add "(MCM)" to MCM columns
  new_names <- names(df)
  for (col in mcm_cols) {
    if (col %in% new_names) {
      names(df)[names(df) == col] <- paste0(col, " (MCM)")
    }
  }

  return(df)
}

# 4. Apply cleaning to all groundwater datasets
tn_gw_2020 <- clean_gw(tn_gw_2020)
tn_gw_2022 <- clean_gw(tn_gw_2022)
tn_gw_2023 <- clean_gw(tn_gw_2023)
tn_gw_2024 <- clean_gw(tn_gw_2024)




## ---- Make all groundwater numeric (except district) --------------------

make_gw_numeric <- function(df) {
  df %>%
    mutate(
      across(-`Name of District`, ~ parse_number(as.character(.)))
    )
}

tn_gw_2020 <- tn_gw_2020 %>% make_gw_numeric()
tn_gw_2022 <- tn_gw_2022 %>% make_gw_numeric()
tn_gw_2023 <- tn_gw_2023 %>% make_gw_numeric()
tn_gw_2024 <- tn_gw_2024 %>% make_gw_numeric()

## ---- Add year and stack into panel -------------------------------------
```

```
tn_gw_2020 <- tn_gw_2020 %>% mutate(year = 2020L)
tn_gw_2022 <- tn_gw_2022 %>% mutate(year = 2022L)
tn_gw_2023 <- tn_gw_2023 %>% mutate(year = 2023L)
tn_gw_2024 <- tn_gw_2024 %>% mutate(year = 2024L)

gw_panel <- bind_rows(tn_gw_2020, tn_gw_2022, tn_gw_2023, tn_gw_2024) %>%
  rename(district = `Name of District`)

## ---- Construct key groundwater variables (no duplicates) --------------

gw_panel <- gw_panel %>%
  mutate(
    # create extraction shares (these are new variables)
    share_irrig = if_else(`Total Annual Extraction (MCM)` > 0,
                          `Irrigation - Annual extraction (MCM)` / `Total Annual
Extraction (MCM)`,
                          NA_real_),
    share_ind = if_else(`Total Annual Extraction (MCM)` > 0,
                        `Industrial - Annual extraction (MCM)` / `Total Annual Extraction
(MCM)`,
                        NA_real_),
    share_dom = if_else(`Total Annual Extraction (MCM)` > 0,
                        `Domestic - Annual Extraction (MCM)` / `Total Annual Extraction
(MCM)`,
                        NA_real_)
  ) %>%
  group_by(district) %>%
  mutate(
    StageGW_2020 = `Stage of GW extraction (%)`[year == 2020][1],
    dStageGW     = `Stage of GW extraction (%)` - StageGW_2020
  ) %>%
  ungroup()



## ----------- Summary ------------------------##
gw_summary_by_year <- gw_panel %>%
  group_by(year) %>%
  summarise(
    n_districts      = n_distinct(district),
    mean_stage       = mean(`Stage of GW extraction (%)`, na.rm = TRUE),
    sd_stage         = sd(`Stage of GW extraction (%)`, na.rm = TRUE),
    min_stage        = min(`Stage of GW extraction (%)`, na.rm = TRUE),
    max_stage        = max(`Stage of GW extraction (%)`, na.rm = TRUE),
    mean_extract     = mean(`Total Annual Extraction (MCM)`, na.rm = TRUE),
    mean_share_irrig = mean(share_irrig, na.rm = TRUE),
    .groups = "drop"
  )
gw_summary_by_year

## ---- Plot 1: Histogram of Stage of GW extraction by year ---------------

ggplot(gw_panel, aes(x = `Stage of GW extraction (%)`)) +
  geom_histogram(bins = 25) +
  facet_wrap(~ year) +
  labs(
    title = "Distribution of groundwater extraction stress by year",
    x = "Stage of groundwater extraction (percent)",
    y = "Number of districts"
  ) +
  theme_minimal()


## ---- Plot 2: Mean Stage of GW extraction over time --------------------
```

```
mean_stage_time <- gw_panel %>%
  group_by(year) %>%
  summarise(
    mean_stage = mean(`Stage of GW extraction (%)`, na.rm = TRUE),
    .groups = "drop"
  )

ggplot(mean_stage_time, aes(x = year, y = mean_stage)) +
  geom_line() +
  geom_point() +
  labs(
    title = "Average stage of groundwater extraction in Tamil Nadu",
    x = "Year",
    y = "Average stage of extraction (percent)"
  ) +
  theme_minimal()

## ---- Plot 3: GW stress vs irrigation share of extraction --------------

ggplot(gw_panel, aes(x = share_irrig, y = `Stage of GW extraction (%)`)) +
  geom_point(alpha = 0.7) +
  facet_wrap(~ year) +
  labs(
    title = "Groundwater stress and irrigation share of extraction",
    x = "Irrigation share of total extraction",
    y = "Stage of groundwater extraction (percent)"
  ) +
  theme_minimal()


## ---- Plot 4: Change in groundwater stress from 2020 baseline ----------

gw_change_plot <- gw_panel %>%
  filter(!is.na(StageGW_2020)) %>%   # excludes Mayiladuthurai for this plot
  group_by(district) %>%
  summarise(
    dStageGW_2024 = `Stage of GW extraction (%)`[year == 2024] - StageGW_2020
  ) %>%
  arrange(dStageGW_2024)

ggplot(gw_change_plot, aes(x = reorder(district, dStageGW_2024),
                           y = dStageGW_2024)) +
  geom_col(fill = "darkgreen") +
  coord_flip() +
  labs(
    title = "Change in groundwater stress from 2020 to 2024",
    x = "District",
    y = "Δ Stage of GW extraction (%)"
  ) +
  theme_minimal()

## ---- Plot 5: Recharge vs Total Extraction -----------------------------

ggplot(gw_panel, aes(x = `Total annual groundwater recharge (MCM)`,
                     y = `Total Annual Extraction (MCM)`,
                     color = district)) +
  geom_point(size = 3, alpha = 0.7) +
  facet_wrap(~ year) +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
  labs(
    title = "Extraction vs Recharge by District (dashed line = sustainable threshold)",
    x = "Recharge (MCM)",
    y = "Extraction (MCM)"
  ) +
```

```
  theme_minimal() +
  theme(legend.position = "none")


## ---- Plot A: District groundwater stress trends (sorted, labeled) -----

gw_trends <- gw_panel %>%
  arrange(`Stage of GW extraction (%)`) %>%
  mutate(district = factor(district, levels = unique(district)))

ggplot(gw_trends, aes(x = year,
                      y = `Stage of GW extraction (%)`,
                      group = district)) +
  geom_line(alpha = 0.6) +
  geom_point(size = 1.5) +
  facet_wrap(~ district, scales = "free_y", ncol = 6) +
  labs(
    title = "District groundwater stress trajectories (2020–2024)",
    x = "Year",
    y = "Stage of groundwater extraction (%)"
  ) +
  theme_minimal(base_size = 9)

## ---- Plot B: Irrigation share vs normalized extraction ----------------

# temporary normalization using extraction per MCM of recharge
# (a sustainability-adjusted intensity measure)

gw_panel <- gw_panel %>%
  mutate(
    extraction_intensity = `Total Annual Extraction (MCM)` /
      `Total annual groundwater recharge (MCM)`
  )

ggplot(gw_panel, aes(x = share_irrig,
                     y = extraction_intensity)) +
  geom_point(alpha = 0.7, color = "darkblue") +
  facet_wrap(~ year) +
  labs(
    title = "Irrigation dependence vs. extraction intensity",
    x = "Irrigation share of extraction",
    y = "Extraction intensity (Extraction ÷ Recharge)"
  ) +
  theme_minimal()
```

```
#####################################################
## -------------- APY_REPORT cleaning ----------- ##
#####################################################

crop1_col <- names(apy_report)[4]

# Remove State column and ONLY the first metadata row
apy_clean_step1 <- apy_report %>%
  dplyr::select(-1) %>%
  dplyr::slice(-1)

# Fill district names downward; clean numbering
apy_clean_step2 <- apy_clean_step1 %>%
  mutate(
    District = na_if(District, ""),
    District = na_if(District, " ")
  ) %>%
  tidyr::fill(District, .direction = "down") %>%
  mutate(
    District = str_replace(District, "^[0-9]+\\.\\s*", ""),
    District = str_squish(District),
    District = str_to_title(District)
  )


head(apy_clean_step1, 5)
head(apy_clean_step2, 10)

# drop the "Area / Production / Yield" metadata row
first_crop_col <- names(apy_clean_step2)[3]

apy_data <- apy_clean_step2 %>%
  filter(
    is.na(.data[[first_crop_col]]) |
      !str_detect(.data[[first_crop_col]], regex("^area", ignore_case = TRUE))
  )

# rename triplets as Crop_Area / Crop_Production / Crop_Yield
all_cols  <- names(apy_data)
id_cols   <- c("District", "Year")
data_cols <- setdiff(all_cols, id_cols)

stopifnot(length(data_cols) %% 3 == 0)

new_names <- all_cols
for (i in seq(1, length(data_cols), by = 3)) {
  triplet <- data_cols[i:(i+2)]
  crop    <- data_cols[i]

  new_names[match(triplet[1], all_cols)] <- paste0(crop, "_Area")
  new_names[match(triplet[2], all_cols)] <- paste0(crop, "_Production")
  new_names[match(triplet[3], all_cols)] <- paste0(crop, "_Yield")
}
names(apy_data) <- new_names

# long format: one row per District–Year–Crop, season = "Whole Year"
apy_long_old <- apy_data %>%
  pivot_longer(
    cols = -c(District, Year),
    names_to  = c("crop", "measure"),
    names_sep = "_",
    values_to = "value"
  ) %>%
  pivot_wider(
```

```
    names_from  = measure,
    values_from = value
  ) %>%
  mutate(
    Area       = parse_number(as.character(Area)),
    Production = parse_number(as.character(Production)),
    Yield      = parse_number(as.character(Yield)),
    season     = "Whole Year"
  )



#####################################################
## ------- APY_REPORT_TOP7 seasonal cleaning ----- ##
#####################################################

apy7_raw <- read_csv(
  '/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/EEcon - final
project/horizontal_crop_vertical_year_report - final.xls.csv',
  col_names = FALSE,
  show_col_types = FALSE
)

# Row 1: crop names, Row 2: seasons
header1 <- apy7_raw[1, ]
header2 <- apy7_raw[2, ]

n_cols <- ncol(apy7_raw)
new_names7 <- paste0("X", seq_len(n_cols))

# first three columns: State, District, Year
new_names7[1:3] <- c("State", "District", "Year")

# carry forward last non-empty crop name across triplets
last_crop_name <- NA_character_

for (j in seq(4, n_cols, by = 3)) {
  if (j + 2 > n_cols) break

  crop_name_raw <- as.character(header1[[j]])
  season_val    <- as.character(header2[[j]])

  if (!is.na(crop_name_raw) && str_squish(crop_name_raw) != "") {
    last_crop_name <- crop_name_raw
  }

  crop_name_clean <- str_squish(last_crop_name)
  season_clean    <- str_squish(season_val)

  # skip if we still have no crop
  if (is.na(crop_name_clean) || crop_name_clean == "") next

  # if season missing (eg whole-year crops), label as "Whole Year"
  if (is.na(season_clean) || season_clean == "") {
    season_clean <- "Whole Year"
  }

  new_names7[j]     <- paste(crop_name_clean, season_clean, "Area", sep = "_")
  new_names7[j + 1] <- paste(crop_name_clean, season_clean, "Production", sep = "_")
  new_names7[j + 2] <- paste(crop_name_clean, season_clean, "Yield", sep = "_")
}

# drop first 3 header rows and apply names
apy7_data <- apy7_raw[-c(1:3), ]
```

```
names(apy7_data) <- new_names7

# remove State, clean District
apy7_data <- apy7_data %>%
  dplyr::select(-State) %>%
  dplyr::mutate(
    District = na_if(District, ""),
    District = na_if(District, " ")
  ) %>%
  tidyr::fill(District, .direction = "down") %>%
  dplyr::mutate(
    District = str_replace(District, "^[0-9]+\\.\\s*", ""),
    District = str_squish(District),
    District = str_to_title(District)
  )

# keep only crop_season_measure columns (with two underscores)
id_cols7   <- c("District", "Year")
data_cols7 <- names(apy7_data)[
  !(names(apy7_data) %in% id_cols7) &
    str_count(names(apy7_data), "_") == 2
]

apy_long_seasonal <- apy7_data %>%
  pivot_longer(
    cols      = all_of(data_cols7),
    names_to  = c("crop", "season", "measure"),
    names_sep = "_",
    values_to = "value"
  ) %>%
  pivot_wider(
    names_from  = measure,
    values_from = value
  ) %>%
  mutate(
    Area       = parse_number(as.character(Area)),
    Production = parse_number(as.character(Production)),
    Yield      = parse_number(as.character(Yield))
  )







####################################################
## -------- Combine APY files, drop dups -------- ##
####################################################

apy_long_all <- bind_rows(
  apy_long_old %>% mutate(source = "whole_year"),
  apy_long_seasonal %>% mutate(source = "seasonal")
)

# keep unique rows on the main analytical fields
apy_long_all <- apy_long_all %>%
  distinct(District, Year, crop, season, Area, Production, Yield, .keep_all = TRUE)

# optional: quick sanity checks
# apy_long_all %>% distinct(crop, season) %>% arrange(crop, season)
# apy_long_all %>% count(District, Year, crop, season) %>% arrange(desc(n))
```

```
apy_long_all



# check there are no exact duplicates
dup_check <- apy_long_all %>%
  distinct(District, Year, crop, season, Area, Production, Yield, .keep_all = FALSE)

stopifnot(nrow(dup_check) == nrow(apy_long_all))


focus_crops <- c(
  "Bajra",
  "Banana",
  "Coconut",
  "Cotton(lint)",
  "Ragi",
  "Rice",
  "Sugarcane"
)

apy_focus <- apy_long_all %>%
  filter(crop %in% focus_crops)




################################################################################
## ---- Parse APY year to match groundwater year ------------------------
################################################################################

apy_long_all <- apy_long_all %>%
  mutate(
    agri_year = readr::parse_number(stringr::str_sub(Year, -4, -1))  # last 4 digits
  )

## ---- Focus on 7 key crops ---------------------------------------------

focus_crops <- c("Bajra",
                 "Banana",
                 "Coconut",
                 "Cotton(lint)", "Cotton (Lint)", "Cotton Lint",
                 "Ragi",
                 "Rice",
                 "Sugarcane")

apy_focus <- apy_long_all %>%
  mutate(crop = stringr::str_squish(crop)) %>%
  filter(crop %in% focus_crops)

## ---- Aggregate to district–year (sum across 7 crops and seasons) -----

apy_dy <- apy_focus %>%
  group_by(District, agri_year) %>%
  summarise(
    area_7crops       = sum(Area, na.rm = TRUE),
    production_7crops = sum(Production, na.rm = TRUE),
    # you can also compute an average yield if you want
    yield_avg_7crops  = production_7crops / area_7crops,
    .groups = "drop"
  )
```

```
## ---- Merge with groundwater panel ------------------------------------

gw_agri_panel <- gw_panel %>%
  left_join(
    apy_dy,
    by = c("district" = "District", "year" = "agri_year")
  )

# quick look
gw_agri_panel %>%
  dplyr::select(district, year, `Stage of GW extraction (%)`, area_7crops,
production_7crops) %>%
  head(15)

# -------------------- Prepare dataset for plots -------------------------

gw_reg_data <- gw_agri_panel %>%
  filter(!is.na(area_7crops)) %>%
  mutate(
    log_area_7crops = log1p(area_7crops),
    StageGW = `Stage of GW extraction (%)`
  ) %>%
  group_by(district) %>%
  mutate(
    StageGW_2020 = StageGW[year == 2020][1],
    dStageGW     = StageGW - StageGW_2020
  ) %>%
  ungroup()


## ---- Check coverage of merged groundwater–agri data -------------------
gw_agri_panel %>%
  group_by(year) %>%
  summarise(
    n_districts_total = n_distinct(district),
    n_with_agri       = n_distinct(district[!is.na(area_7crops)]),
    .groups = "drop"
  )


## ----- 1. COLORED BOXPLOT: AREA IN 7 CROPS BY YEAR -----
ggplot(gw_reg_data, aes(x = factor(year), y = area_7crops, fill = factor(year))) +
  geom_boxplot() +
  scale_fill_brewer(palette = "Set2") +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "Area in 7 Major Crops by Year",
    x = "Year",
    y = "Total Crop Area (hectares)"
  ) +
  theme_minimal()


## ----- 2. COLORED SCATTER: GW STRESS VS AREA -----
ggplot(gw_reg_data, aes(x = `Stage of GW extraction (%)`,
                        y = area_7crops,
                        color = area_7crops)) +
  geom_point(size = 3, alpha = 0.8) +
  facet_wrap(~ year) +
  scale_color_gradient(low = "yellow", high = "red", labels = scales::comma) +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "Groundwater Stress vs Cropped Area (7 Major Crops)",
    x = "Stage of GW Extraction (%)",
```

```
    y = "Crop Area (hectares)"
  ) +
  theme_minimal()
```

```
######################################################################
# RAINFALL: GeoJSON
######################################################################

## 1. Read Tamil Nadu district polygons from GeoJSON ----------------------
tn_geo_path <- '/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes
and Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/TAMIL NADU_DISTRICTS.geojson'

tn_sf_raw <- sf::st_read(tn_geo_path)

tn_sf <- tn_sf_raw %>%
  dplyr::mutate(
    # use the 'dist' column as the canonical district name
    district = dist %>%
      stringr::str_squish() %>%       # remove extra spaces
      stringr::str_to_title() %>%     # consistent capitalization
      dplyr::recode(!!!gw_name_map)   # fix known spelling mismatches
  ) %>%
  dplyr::select(district, geometry) %>%  # keep only what we need
  sf::st_make_valid()

# quick sanity check
tn_sf


######################################################################
```

```
# 1. Your list of rainfall NetCDF files (only the .nc files)
#####################################################################

rain_nc_files <- c(
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2016_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2017_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2018_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2019_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2020_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2021_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2022_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2023_rfp25.nc",
  "/Users/velsenthil/Library/CloudStorage/OneDrive-Stanford/Stanford Classes and
Notes/Graduate Program/GEP 230 - Environmental Economics/Final
Project/Datasets/rainfall/RF25_ind2024_rfp25.nc"
)

#####################################################################
# 2. Helper function to extract year from filename
#####################################################################

get_year_from_path <- function(path) {
  as.numeric(stringr::str_extract(path, "20\\d{2}"))
}

#####################################################################
# 3. Function: extract district rainfall from one NetCDF file
#####################################################################

extract_rainfall_onefile <- function(nc_path, districts_sf) {
  year <- get_year_from_path(nc_path)
  message("Extracting rainfall for ", year)
  r <- terra::rast(nc_path)
  # Annual total rainfall = sum of all layers
  r_total <- sum(r, na.rm = TRUE)

  # Align CRS
  districts_aligned <- sf::st_transform(districts_sf, terra::crs(r_total))

  # Mean rainfall by district polygon
  vals <- exactextractr::exact_extract(r_total, districts_aligned, "mean")
  tibble(
    district = districts_sf$district,
    year = year,
    rain_mm = vals
  )
}

#####################################################################
```

```
# 4. RAINFALL: GeoJSON + NetCDF → district year panel
######################################################################

rain_panel <- purrr::map_dfr(
  rain_nc_files,
  ~ extract_rainfall_onefile(.x, tn_sf)
)

######################################################################
# 5. Inspect results
######################################################################

print(head(rain_panel, 10))
rain_panel %>%
  group_by(year) %>%
  summarise(
    districts = n_distinct(district),
    mean_rain = mean(rain_mm, na.rm = TRUE),
    sd_rain = sd(rain_mm, na.rm = TRUE)
  )

######################################################################
# 6. Merge into your groundwater + agriculture panel
######################################################################

gw_agri_rain_panel <- gw_agri_panel %>%
  left_join(rain_panel, by = c("district", "year"))

######################################################################
# 7. Quick diagnostic plot
######################################################################

ggplot(gw_agri_rain_panel, aes(x = rain_mm, y = `Stage of GW extraction (%)`)) +
  geom_point(alpha = 0.6) +
  facet_wrap(~ year) +
  theme_minimal() +
  labs(
    title = "Rainfall vs groundwater stress",
    x = "Annual rainfall (mm)",
    y = "GW stress (%)"
  )









######################################################################
# SEASONAL RAINFALL EXTRACTION (Summer, Kharif, Autumn, Rabi, Winter)
######################################################################

get_nc_dates <- function(nc_path, raster_obj) {

  # Try terra first (works for 2024)
  terra_time <- try(terra::time(raster_obj), silent = TRUE)

  if (!inherits(terra_time, "try-error") && !all(is.na(terra_time))) {
    return(as.Date(terra_time))
  }

  # Fallback to ncdf4 (works for 2016–2023)
  nc <- ncdf4::nc_open(nc_path)
```

```
  # TIME may be "TIME" or "time"
  time_var_name <- intersect(c("TIME", "time"), names(nc$dim))
  if (length(time_var_name) == 0) stop("No TIME variable found")

  time_raw <- ncdf4::ncvar_get(nc, time_var_name)
  time_units <- ncdf4::ncatt_get(nc, time_var_name, "units")$value
  ncdf4::nc_close(nc)

  # Parse origin
  origin_str <- stringr::str_extract(time_units, "\\d{1,4}-\\d{1,2}-\\d{1,2}")
  origin_date <- as.Date(origin_str)

  # hours vs days handling
  if (grepl("hours since", time_units)) {
    return(origin_date + time_raw / 24)
  } else if (grepl("days since", time_units)) {
    return(origin_date + time_raw)
  } else {
    stop("Unrecognized TIME unit")
  }
}

######################################################################
# FIXED SEASONAL EXTRACTION USING ncdf4 FOR TIME DIMENSION
######################################################################

extract_seasonal_rainfall_onefile <- function(nc_path, districts_sf) {

  year <- get_year_from_path(nc_path)
  message("Extracting seasonal rainfall for ", year)

  r <- terra::rast(nc_path)
  dates <- get_nc_dates(nc_path, r)

  season_defs <- list(
    summer  = c("03-01", "05-31"),
    kharif  = c("06-01", "09-30"),
    autumn  = c("10-01", "10-31"),
    rabi    = c("11-01", "02-28"),
    winter  = c("12-01", "01-31")
  )

  if (lubridate::leap_year(year)) {
    season_defs$rabi[2] <- "02-29"
  }

  out <- list()

  for (s in names(season_defs)) {

    start_md <- season_defs[[s]][1]
    end_md   <- season_defs[[s]][2]

    if (s %in% c("rabi", "winter")) {
      start_date <- as.Date(sprintf("%d-%s", year-1, start_md))
      end_date   <- as.Date(sprintf("%d-%s", year, end_md))
    } else {
      start_date <- as.Date(sprintf("%d-%s", year, start_md))
      end_date   <- as.Date(sprintf("%d-%s", year, end_md))
    }

    layer_idx <- which(dates >= start_date & dates <= end_date)

    if (length(layer_idx) == 0) {
```

```
      out[[s]] <- rep(NA_real_, nrow(districts_sf))
      next
    }

    seasonal_raster <- sum(r[[layer_idx]], na.rm = TRUE)
    districts_aligned <- sf::st_transform(districts_sf, terra::crs(seasonal_raster))
    vals <- exactextractr::exact_extract(seasonal_raster, districts_aligned, "mean")

    out[[s]] <- vals
  }

  tibble(
    district = districts_sf$district,
    year = year,
    rain_summer  = out$summer,
    rain_kharif  = out$kharif,
    rain_autumn  = out$autumn,
    rain_rabi    = out$rabi,
    rain_winter  = out$winter
  )
}

######################################################################
# Build FULL seasonal rainfall panel
######################################################################

rain_seasonal_panel <- purrr::map_dfr(
  rain_nc_files,
  ~ extract_seasonal_rainfall_onefile(.x, tn_sf)
)
######################################################################
# Merge with existing rainfall + groundwater + agriculture panel
######################################################################

gw_agri_rain_panel <- gw_agri_rain_panel %>%
  left_join(rain_seasonal_panel, by = c("district", "year"))







######################################################################
# Prepare spatial rainfall data for mapping
######################################################################

tn_rain_sf <- tn_sf %>%
  left_join(rain_panel, by = "district") %>%
  left_join(rain_seasonal_panel, by = c("district", "year"))


year_to_plot <- 2020      # we just took a look at 2020 rainfall data, we can change
depending on what year we want to look at

#spatial rainfall data for       year_to_plot <- 2020
ggplot(
  tn_rain_sf %>% filter(year == year_to_plot)
) +
  geom_sf(aes(fill = rain_mm), color = "white", size = 0.1) +
  scale_fill_viridis_c(option = "C", name = "Rainfall (mm)") +
  labs(
    title = paste("Annual Rainfall in Tamil Nadu –", year_to_plot),
    subtitle = "District-level extraction from IMD 0.25 degree daily rainfall",
```

```
    caption = "Data: IMD gridded rainfall (daily) aggregated to districts"
  ) +
  theme_minimal()



#spatial rainfall data for all years
ggplot(tn_rain_sf) +
  geom_sf(aes(fill = rain_mm), color = NA) +
  scale_fill_viridis_c(option = "C") +
  facet_wrap(~ year) +
  theme_minimal() +
  labs(
    title = "District-level Annual Rainfall Across Years",
    fill = "Rainfall (mm)"
  )









#=============================================================
#STORY 0: Create regression-ready variables
#=============================================================
analysis_df <- gw_agri_rain_panel %>%
  filter(!is.na(area_7crops)) %>%
  mutate(
    extraction_intensity = `Total Annual Extraction (MCM)` / `Total annual groundwater
recharge (MCM)`,
    stress_group = case_when(
      `Stage of GW extraction (%)` < 60 ~ "Low",
      `Stage of GW extraction (%)` < 80 ~ "Medium",
      TRUE ~ "High"
    )
  ) %>%
  group_by(district) %>%
  mutate(
    rain_ann_anom  = rain_mm - mean(rain_mm, na.rm = TRUE),
    rain_kharif_anom = rain_kharif - mean(rain_kharif, na.rm = TRUE),
    StageGW_2020 = `Stage of GW extraction (%)`[year == 2020][1],
    dStageGW = `Stage of GW extraction (%)` - StageGW_2020
  ) %>%
  ungroup()


#=============================================================
#  STORY 1: Rainfall → crop yield
#=============================================================
ggplot(analysis_df, aes(x = rain_mm, y = yield_avg_7crops)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "loess", se = FALSE, color = "blue") +
  labs(
    title = "Effect of Annual Rainfall on Average Yield (7 Crops)",
    x = "Annual rainfall (mm)",
    y = "Yield (tons/ha)"
  ) +
  theme_minimal()

rice_df <- apy_long_all %>%
  filter(crop == "Rice", season == "Kharif") %>%
```

```
  mutate(year = agri_year) %>%
  left_join(rain_seasonal_panel, by = c("District" = "district", "year"))

ggplot(rice_df, aes(x = rain_kharif, y = Yield)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(
    title = "Rice Yield Response to Kharif Rainfall",
    x = "Kharif rainfall (mm)",
    y = "Rice yield (tons/ha)"
  ) +
  theme_minimal()









# ------------------------------------------------------------------
# Clean Rice panel for rainfall x yield plot (Summer + Autumn + Winter)
# ------------------------------------------------------------------

rice_df <- apy_long_all %>%
  mutate(
    crop_clean   = stringr::str_squish(stringr::str_to_title(crop)),
    season_clean = stringr::str_squish(stringr::str_to_title(season)),
    district     = stringr::str_squish(stringr::str_to_title(District)),
    year         = agri_year
  ) %>%
  # keep only rice and seasons that actually appear for rice
  filter(
    crop_clean == "Rice",
    season_clean %in% c("Summer", "Autumn", "Winter")
  ) %>%
  group_by(district, year) %>%
  summarise(
    rice_area  = sum(Area, na.rm = TRUE),
    rice_prod  = sum(Production, na.rm = TRUE),
    rice_yield = rice_prod / rice_area,
    .groups = "drop"
  ) %>%
  # merge with seasonal rainfall: use three relevant rainfall periods
  left_join(
    rain_seasonal_panel %>%
      dplyr::select(district, year,
             rain_summer,
             rain_autumn,
             rain_winter),
    by = c("district", "year")
  ) %>%
  mutate(
    # rainfall during rice-growing stages
    rice_rain_total = rain_summer + rain_autumn + rain_winter
  )


ggplot(
  rice_df %>% filter(!is.na(rice_yield), !is.na(rice_rain_total)),
```

```
  aes(x = rice_rain_total, y = rice_yield)
) +
  geom_point(alpha = 0.6, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "red") +
  labs(
    title = "Rice Yield Response to Growing-Season Rainfall",
    subtitle = "Rainfall = Summer + Autumn + Winter",
    x = "Growing-season rainfall (mm)",
    y = "Rice yield (tons per hectare)"
  ) +
  theme_minimal()

rice_df_long <- rice_df %>%
  dplyr::select(district, year, rice_yield, rain_summer, rain_autumn, rain_winter) %>%
  pivot_longer(
    cols = starts_with("rain_"),
    names_to = "season",
    values_to = "rain_mm"
  ) %>%
  mutate(season = recode(season,
                         rain_summer = "Summer Rainfall",
                         rain_autumn = "Autumn Rainfall",
                         rain_winter = "Winter Rainfall"))

ggplot(
  rice_df_long %>% filter(!is.na(rain_mm), !is.na(rice_yield)),
  aes(x = rain_mm, y = rice_yield, color = season)
) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ season, scales = "free_x") +
  labs(
    title = "Season-specific Rainfall Effects on Rice Yield",
    x = "Seasonal rainfall (mm)",
    y = "Rice yield (tons/ha)"
  ) +
  theme_minimal()




ggplot(
  rice_df %>% filter(!is.na(rice_yield)),
  aes(x = year, y = rice_yield, group = district)
) +
  geom_line(alpha = 0.4) +
  geom_point(size = 1) +
  labs(
    title = "Rice Yield Trajectories Across Districts",
    x = "Year",
    y = "Rice yield (tons/ha)"
  ) +
  theme_minimal()



rice_sensitivity <- rice_df %>%
  filter(!is.na(rice_rain_total), !is.na(rice_yield)) %>%
  group_by(district) %>%
  summarise(
    sensitivity = coef(lm(rice_yield ~ rice_rain_total))[2]
  ) %>%
  arrange(sensitivity)
```

```
ggplot(rice_sensitivity, aes(x = reorder(district, sensitivity), y = sensitivity)) +
  geom_col(fill = "darkgreen") +
  coord_flip() +
  labs(
    title = "District-level Sensitivity of Rice Yield to Rainfall",
    x = "District",
    y = "Yield response (tons/ha per mm rainfall)"
  ) +
  theme_minimal()
```

```
gw_final <- gw_agri_rain_panel %>%
  dplyr::mutate(
    district = stringr::str_squish(stringr::str_to_title(district)),
    year = as.integer(year),

    StageGW = `Stage of GW extraction (%)`,
    log_area_7crops = log1p(area_7crops),
    log_prod_7crops = log1p(production_7crops),

    yield_avg_7crops = ifelse(area_7crops > 0, production_7crops / area_7crops, NA_real_),

    extraction_mcm = `Total Annual Extraction (MCM)`,
    recharge_mcm   = `Total annual groundwater recharge (MCM)`,

    extraction_intensity = ifelse(recharge_mcm > 0, extraction_mcm / recharge_mcm,
NA_real_)
  ) %>%
  dplyr::filter(!is.na(district), !is.na(year)) %>%
  dplyr::arrange(district, year)
```

```
#Table 1. Descriptive Statistics Table

vars_for_table <- c(
  "StageGW", "dStageGW",
  "extraction_mcm", "recharge_mcm", "extraction_intensity",
  "rain_mm", "rain_summer", "rain_kharif", "rain_autumn", "rain_rabi", "rain_winter",
  "area_7crops", "production_7crops", "yield_avg_7crops"
)

desc_table <- gw_final %>%
  dplyr::select(dplyr::all_of(c("district", "year", vars_for_table))) %>%
  tidyr::pivot_longer(cols = dplyr::all_of(vars_for_table),
                      names_to = "variable",
                      values_to = "value") %>%
  dplyr::group_by(variable) %>%
  dplyr::summarise(
    N = sum(!is.na(value)),
    Mean = mean(value, na.rm = TRUE),
    SD = sd(value, na.rm = TRUE),
    Min = min(value, na.rm = TRUE),
    Median = median(value, na.rm = TRUE),
    Max = max(value, na.rm = TRUE),
    .groups = "drop"
  )
```

```
desc_table

desc_table_fmt <- desc_table %>%
  dplyr::mutate(across(where(is.numeric), ~ round(.x, 3)))

desc_table_fmt




#6.A. Core regression in Fixest
gw_reg <- gw_final %>%
  dplyr::filter(year %in% c(2020, 2022, 2023, 2024)) %>%
  dplyr::filter(!is.na(StageGW), !is.na(rain_mm)) %>%
  dplyr::filter(!is.na(area_7crops), !is.na(production_7crops))

m1 <- feols(log_area_7crops ~ StageGW + rain_mm | district + year,
            data = gw_reg, cluster = "district")

m2 <- feols(log_prod_7crops ~ StageGW + rain_mm | district + year,
            data = gw_reg, cluster = "district")

m3 <- feols(yield_avg_7crops ~ StageGW + rain_mm | district + year,
            data = gw_reg, cluster = "district")

etable(m1, m2, m3, se = "cluster")


#6.B. Lagged groundwater stress robustness check
gw_reg_lag <- gw_reg %>%
  dplyr::group_by(district) %>%
  dplyr::arrange(year, .by_group = TRUE) %>%
  dplyr::mutate(StageGW_lag = dplyr::lag(StageGW, 1)) %>%
  dplyr::ungroup()

m1_lag <- feols(log_area_7crops ~ StageGW_lag + rain_mm | district + year,
                data = gw_reg_lag, cluster = "district")

m2_lag <- feols(log_prod_7crops ~ StageGW_lag + rain_mm | district + year,
                data = gw_reg_lag, cluster = "district")

m3_lag <- feols(yield_avg_7crops ~ StageGW_lag + rain_mm | district + year,
                data = gw_reg_lag, cluster = "district")

etable(m1_lag, m2_lag, m3_lag, se = "cluster")



#6.C. Seasonal rainfall controls
m_season <- feols(
  yield_avg_7crops ~ StageGW + rain_summer + rain_kharif + rain_autumn + rain_rabi +
rain_winter | district + year,
  data = gw_reg, cluster = "district"
)

etable(m3, m_season, se = "cluster")


#6.D.1: Changes in yield versus changes in groundwater stress, within district
gw_delta <- gw_reg %>%
  dplyr::group_by(district) %>%
  dplyr::mutate(
    yield_2020 = yield_avg_7crops[year == 2020][1],
    Stage_2020 = StageGW[year == 2020][1],
```

```
    d_yield = yield_avg_7crops - yield_2020,
    d_stage = StageGW - Stage_2020
  ) %>%
  dplyr::ungroup()

ggplot(gw_delta, aes(x = d_stage, y = d_yield)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = TRUE) +
  facet_wrap(~ year) +
  theme_minimal() +
  labs(
    title = "Within district changes in yield versus groundwater stress",
    x = "Change in Stage of groundwater extraction relative to 2020",
    y = "Change in average yield relative to 2020"
  )


#6.D.2: Rainfall vs. Yeild by Year
ggplot(gw_reg, aes(x = rain_mm, y = yield_avg_7crops)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = TRUE) +
  facet_wrap(~ year) +
  theme_minimal() +
  labs(
    title = "Rainfall and agricultural yield",
    x = "Annual rainfall (mm)",
    y = "Average yield across seven major crops"
  )







#Section 4: Figure 2
mean_stage_time <- gw_panel %>%
  group_by(year) %>%
  summarise(
    mean_stage = mean(`Stage of GW extraction (%)`, na.rm = TRUE),
    .groups = "drop"
  )

ggplot(mean_stage_time, aes(x = year, y = mean_stage)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  labs(
    title = "Average Groundwater Stress in Tamil Nadu Over Time",
    x = "Year",
    y = "Mean Stage of Groundwater Extraction (%)"
  ) +
  theme_minimal()

ggplot(gw_panel, aes(x = year, y = `Stage of GW extraction (%)`, group = district)) +
  geom_line(alpha = 0.5) +
  facet_wrap(~ district, scales = "free_y", ncol = 6) +
  labs(
    title = "District-Level Groundwater Stress Trajectories",
    x = "Year",
    y = "Stage of Groundwater Extraction (%)"
  ) +
  theme_minimal(base_size = 9)


ggplot(gw_reg_data,
       aes(x = StageGW, y = area_7crops)) +
```

```
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ year) +
  labs(
    title = "Groundwater Stress and Cropped Area",
    x = "Stage of Groundwater Extraction (%)",
    y = "Total Cropped Area (7 Crops)"
  ) +
  theme_minimal()


# ------------------------------------------
# Figure 4: Rice yield vs seasonal rainfall
# ------------------------------------------

ggplot(rice_df, aes(x = rice_rain_total, y = rice_yield)) +
  geom_point(alpha = 0.6, color = "darkgreen") +
  geom_smooth(method = "lm", se = TRUE, color = "black") +
  facet_wrap(~ year) +
  scale_x_continuous(labels = scales::comma) +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "Rice Yield and Seasonal Rainfall",
    subtitle = "Rainfall aggregated over summer, autumn, and winter growing periods",
    x = "Seasonal rainfall during rice growing periods (mm)",
    y = "Rice yield (tonnes per hectare)"
  ) +
  theme_minimal()




rice_season_long <- apy_long_all %>%
  mutate(
    crop_clean   = str_to_title(str_squish(crop)),
    season_clean = str_to_title(str_squish(season)),
    district     = str_to_title(str_squish(District)),
    year         = agri_year
  ) %>%
  filter(crop_clean == "Rice",
         season_clean %in% c("Summer", "Autumn", "Winter")) %>%
  left_join(
    rain_seasonal_panel,
    by = c("district", "year")
  ) %>%
  mutate(
    season_rain = case_when(
      season_clean == "Summer"  ~ rain_summer,
      season_clean == "Autumn"  ~ rain_autumn,
      season_clean == "Winter"  ~ rain_winter
    )
  )

ggplot(rice_season_long,
       aes(x = season_rain, y = Yield)) +
  geom_point(alpha = 0.6, color = "darkgreen") +
  geom_smooth(method = "lm", se = TRUE, color = "black") +
  facet_wrap(~ season_clean) +
  labs(
    title = "Rice Yield Response to Seasonal Rainfall",
    x = "Seasonal rainfall (mm)",
    y = "Rice yield (tonnes per hectare)"
```

```
) +
theme_minimal()
```