Irrigated areas drive irrigation water withdrawals

R code

Arnald Puy

October 2 2020

Contents

1 Presentation

This document presents the workflow of the paper "Irrigated areas drive irrigation water withdrawals", by A. Puy, E. Borgonovo, S. Lo Piano, S. Levin and A. Saltelli. The abstract is the following:

A sustainable management of global freshwater resources requires producing reliable estimates of the water demanded by irrigated agriculture. This has been attempted by FAO through country surveys and censuses, or through Global Models (GM), which compute irrigation water withdrawals with sub-models on crop types and calendars, evapotranspiration, irrigation efficiencies, weather data and irrigated areas, among others. Here we demonstrate that these strategies err on the side of excess complexity, as the values reported by FAO and outputted by GM are largely driven by irrigated areas only. Modeling irrigation water withdrawals as a function of irrigated areas yields almost the same results in a much cheaper and parsimonious way, while permitting the exploration of all model uncertainties. Our work offers a more robust and transparent approach to compute one of the most important indicators guiding our policies on water security worldwide.

Once all datasets required for the analysis have been acquired from the appropriate sources, the results of the paper should be fully reproducible in any personal computer. Questions about the code or the computational design should be addressed to A. Puy (apuy@princeton.edu; arnald.puy@pm.me).

1.1 Preliminary functions

We start by creating a function to load all required libraries for the analysis in one go. We also set a checkpoint to ensure that our code is fully reproducible for anyone anytime. Finally, we design a short theme function for all the plots that will be produced in the analysis.

```
# LOAD PACKAGES ------------------------------------------------------------

# Function to read in all required packages in one go:
loadPackages <- function(x) {
  for(i in x) {
    if(!require(i, character.only = TRUE)) {
      install.packages(i, dependencies = TRUE)
      library(i, character.only = TRUE)
    }
  }
}

loadPackages(c("data.table", "sensobol", "ncdf4",
               "rworldmap", "sp", "countrycode",
               "IDPmisc", "boot", "parallel", "scales",
               "MASS", "doParallel", "complmrob",
               "mvoutlier", "sandwich", "lmtest", "mice",
               "ggridges", "broom", "naniar", "cowplot",
               "benchmarkme", "tidyverse", "grid", "gridExtra",
               "robustbase", "rsample"))

# SET CHECKPOINT -----------------------------------------------------------
```

```r
dir.create(".checkpoint")

library("checkpoint")

checkpoint("2020-09-08",
           R.version ="3.6.3",
           checkpointLocation = getwd())

# CUSTOM FUNCTION TO DEFINE THE PLOT THEMES ---------------------------------

theme_AP <- function() {
  theme_bw() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          legend.background = element_rect(fill = "transparent",
                                           color = NA),
          legend.key = element_rect(fill = "transparent",
                                    color = NA))
}
```

2 Load datasets

Here we retrieve the datasets needed to conduct the analysis at the national level. We collect irrigation water withdrawal data outputted by eight Global Hydrological Models (GHM) (PCR-GLOBWB, H08, LPJmL, WaterGap, MPI-HM, DBHM), one Land Surface Model (LSM) (VIC) and one Land Earth Surface Model (LESM) (CLM45). We also retrieve irrigation water withdrawal values reported by two FAO-based datasets, Aquastat and Aquastat without missing values (Liu et al. 2016).

For PCR-GLOBWB, H08, LPMjL and WaterGap we use the `.nc` files prepared by Huang et al. (2018). For DBHM, MPI-HM, CLM45 and VIC we use the `.nc` files available in the Inter-Sectoral Impact Model Intercomparison Project (ISIMIP) webpage, https://www.isimip.org. For the FAO-based datasets, we use the irrigation water withdrawal values reported by Aquastat (Frenken and Gillet 2012) and the dataset by Liu et al. (2016), which is similar to Aquastat's but without missing values.

2.1 Define loading functions

Here we define some functions that will be applied later on to read in and clean the datasets just mentioned.

`country_code` unifies the country names across datasets and provides each country with its United Nations code to avoid any confusion during the processing of the data.

`coords2country` transforms the coordinates of the .nc files to the appropriate country names.

`get_nc_data` is tailored to specifically extract irrigation water withdrawal data at the country level from the datasets compiled by Huang et al. (2018). Note that there are calls to `country_code`

and `coors2country` inside the function to transform each longitude and latitude value to a country name.

`open_nc_files` is tailored to specifically extract irrigation water withdrawal data at the country level from the datasets retrieved from ISIMIP. Again, it includes calls to `country_code` and `coors2country`.

```
# CREATE FUNCTIONS ------------------------------------------------------------

# Function to obtain UN code, Continent and Country names
country_code <- function(dt) {
  dt[, `:=` (Code = countrycode(dt[, Country],
                                origin = "country.name",
                                destination = "un"),
             Continent = countrycode(dt[, Country],
                                     origin = "country.name",
                                     destination = "continent"))]
  dt[, Country:= countrycode(dt[, Code],
                             origin = "un",
                             destination = "country.name")]
  setcolorder(dt, c("Country", "Continent", "Code", "Water.Withdrawn"))
  return(dt)
}

## Function to transform longitude and latitude to country.
# It is borrowed from Andy:
# https://stackoverflow.com/questions/14334970/convert-latitude-and-longitude-coordinates-to-c
coords2country = function(points) {
  countriesSP <- rworldmap::getMap(resolution = 'low')
  pointsSP = sp::SpatialPoints(points, proj4string=CRS(proj4string(countriesSP)))
  indices = sp::over(pointsSP, countriesSP)
  indices$ADMIN
  #indices$ISO3 # returns the ISO3 code
  #indices$continent   # returns the continent (6 continent model)
  #indices$REGION   # returns the continent (7 continent model)
}

# Function to load and extract data from .nc files produced
# by Huang et al. 2018
get_nc_data <- function(nc_file) {
  nc <- nc_open(nc_file)
  ww <- ncvar_get(nc, "withd_irr")
  lon <- ncvar_get(nc, "lon")
  lat <- ncvar_get(nc, "lat")
  water <- rowSums(ww[, 469:ncol(ww)]) # Obtain year values for 2010 only
  ww.df <- data.frame(cbind(lon, lat, water))
  countries <- coords2country(ww.df[1:nrow(ww.df), 1:2])
  df <- cbind(countries, ww.df)
  setDT(df)
```

```
  final <- df[, .(Water.Withdrawn = sum(water)), countries]
  setnames(final, "countries", "Country")
  country_code(final)
  out <- na.omit(final[order(Continent)])
  out[, Water.Withdrawn:= Water.Withdrawn / 1000] # From mm to m
  return(out)
}

# Function to load and extract data from .nc files from ISIMIP
open_nc_files <- function(file, dname) {
  ncin <- nc_open(file)
  # get longitude, latitude, time
  lon <- ncvar_get(ncin, "lon")
  lat <- ncvar_get(ncin, "lat")
  # Get variable
  tmp_array <- ncvar_get(ncin, dname)
  # Retrieve last 12 months
  # get last year
  m <- (dim(tmp_array)[3] - 11):dim(tmp_array)[3]
  tmp_slice <- lapply(m, function(m) tmp_array[, , m])
  # create dataframe -- reshape data
  # matrix (nlon*nlat rows by 2 cols) of lons and lats
  lonlat <- as.matrix(expand.grid(lon,lat))
  # vector of `tmp` values
  tmp_vec <- lapply(tmp_slice, function(x) as.vector(x))
  # create dataframe and add names
  tmp_df01 <- lapply(tmp_vec, function(x) data.frame(cbind(lonlat, x)))
  names(tmp_df01) <- m
  da <- lapply(tmp_df01, data.table) %>%
    rbindlist(., idcol = "month") %>%
    na.omit()
  # Convert coordinates to country
  countries <- coords2country(da[1:nrow(da), 2:3])
  df <- cbind(countries, da)
  setDT(df)
  out <- na.omit(df)[, .(Water.Withdrawn = sum(x)), countries]
  out[, Water.Withdrawn:= Water.Withdrawn * 10000]
  return(out)
}
```

2.2 Load GHM and FAO-based datasets

In the next code snippet we load and clean all ten irrigation water withdrawal datasets. We first load the Aquastat dataset (`table4`), then the Aquastat without missing values dataset (`liu.dt`), the `.nc` files from Huang et al. (2018) (PCR-GLOBWB, H08, LPMjL, WaterGap), and the `.nc` files from ISIMIP (DBHM, MPI-HM, CLM45, VIC).

We show at the end of the code chunk that the GHM MPI-HM produced an outlier for Gabon. The

difference with the values outputted by the other GHM is very large and suggests that it is actually a miscalculation. To prevent this outlier from biasing the computations, we transformed it to `NA` in order to impute it later on.

```
# READ IN DATASETS ON IRRIGATION WATER WITHDRAWAL ----------------------------

# Define vector to reorder columns
cols_order <- c("Continent", "Country", "Code", "Water.Dataset", "Water.Withdrawn")

# FAO data (Table 4) ---------------------------
# UNIT IS KM3/YEAR
table4.tmp <- fread("table_4.csv", skip = 3, nrows = 167) %>%
  .[, .(Country, Year, Water.withdrawal)] %>%
  setnames(., "Water.withdrawal", "Water.Withdrawn")

# Extract the selected years
table4.dt <- country_code(table4.tmp[Year %in% 1999:2012])[
  , Water.Dataset:= "Aquastat"][
    , Year:= NULL] %>%
  setcolorder(., cols_order)

# Liu et al. dataset ---------------------------
#UNIT IS 10^9 m3/year = km3/year
liu.dt <- fread("liu.csv")[, .(country, irr)] %>%
  setnames(., c("country", "irr"), c("Country", "Water.Withdrawn")) %>%
  country_code(.) %>%
  .[, Water.Dataset:= "Liu et al. 2016"] %>%
  setcolorder(., cols_order)


# Huang et al datasets --------------------------
names_nc_files <- c("withd_irr_lpjml.nc", "withd_irr_pcrglobwb.nc",
                    "withd_irr_h08.nc", "withd_irr_watergap.nc")
out.nc <- mclapply(names_nc_files, function(x) get_nc_data(x), mc.cores = detectCores() * 0.75)
names(out.nc) <- c("LPJmL", "PCR-GLOBWB", "H08", "WaterGap")
out.final <- rbindlist(out.nc, idcol = "Water.Dataset")

# ISIMIP datasets --------------------------------
files <- list("dbh_wfdei_nobc_hist_varsoc_co2_airrww_global_monthly_1971_2010.nc",
              "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrww_global_monthly_1861_2005.nc",
              "vic_wfdei_nobc_hist_pressoc_co2_airrww_global_monthly_1971_2010.nc")

dname <- "airrww"

isimip.dt <- mclapply(files, function(x) open_nc_files(file = x, dname = dname),
                      mc.cores = detectCores() * 0.75)

names(isimip.dt) <- c("DBHM", "MPI-HM", "VIC")
```

```
# ADD CLM45
CLM45 <- open_nc_files(file = "clm45_gfdl-esm2m_ewembi_historical_2005soc_co2_pirrww_global_mo
                       dname = "pirrww")

CLM45 <- CLM45[, Water.Dataset:= "CLM45"] %>%
  setcolorder(., c("Water.Dataset", "countries", "Water.Withdrawn"))

isimip.final <- rbindlist(isimip.dt, idcol = "Water.Dataset") %>%
  rbind(CLM45) %>%
  setnames(., "countries","Country") %>%
  country_code(.) %>%
  na.omit()

GHM.dt <- rbind(out.final, isimip.final) %>%
  .[order(Country)]

# The value for Gabon by MPI-HM is a clear outlier
GHM.dt[Country == "Gabon"]
```

```
##    Water.Dataset Country Continent Code Water.Withdrawn
## 1:         LPJmL   Gabon    Africa  266    1.644766e-02
## 2:    PCR-GLOBWB   Gabon    Africa  266    1.510275e-02
## 3:           H08   Gabon    Africa  266    1.280429e-02
## 4:      WaterGap   Gabon    Africa  266    1.791356e-02
## 5:          DBHM   Gabon    Africa  266    2.860231e-02
## 6:        MPI-HM   Gabon    Africa  266    1.909227e-08
## 7:           VIC   Gabon    Africa  266    0.000000e+00
## 8:         CLM45   Gabon    Africa  266    0.000000e+00
```

```
# So what we do is to give it a NA
GHM.dt <- GHM.dt[, Water.Withdrawn:= ifelse(Country == "Gabon" &
                                              Water.Dataset == "MPI-HM", NA,
                                            Water.Withdrawn)] %>%
  setcolorder(., cols_order)
```

Next we row-bind the GM and FAO-based datasets, producing the dataset `water.dt`. We then check whether there are duplicated countries resulting from a bad conversion of longitude/latitude to country names. We found that `Cyprus` and `Palestinian Territories` had more than 10 data points, meaning that there were some duplicated values. We computed the mean irrigation water withdrawal for these countries in each water dataset.

```
# CREATE THE FINAL IRRIGATION WATER WITHDRAWAL DATASET -------------------------

water.dt <- rbind(liu.dt, table4.dt, GHM.dt)

# Check if there are duplicated countries
water.dt[, .N, .(Country)][N > 10][, Country]
```

```
## [1] "Cyprus"                  "Palestinian Territories"
```

```
# Compute the mean
water.dt <- water.dt[, .(Water.Withdrawn = mean(Water.Withdrawn)),
                     .(Continent, Country, Code, Water.Dataset)]

# And check again
water.dt[, .N, .(Country)][N > 10][, Country]
```

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

2.3 Load FAO-GMIA dataset

Here we read in the irrigated areas reported by FAO-GMIA. We use the data compiled by Meier, Zabel, and Mauser (2018), and produce the `meier.dt` dataset.

```
# READ IN IRRIGATED AREA DATASET ------------------------------------------

meier.dt <- fread("meier.csv") %>%
  setnames(., "Codes", "Code") %>%
  na.omit() %>%
  .[, .(Country, Continent, Code, `FAO-GMIA`)] %>%
  setnames(., "FAO-GMIA", "Irrigated.Area")
```

2.4 Plot FAO-GMIA against irrigation water withdrawal

In order to obtain the final dataset for the computations, we merge `water.dt` with `meier.dt`. We forced `water.dt` to have the same countries as `meier.dt` (186), and removed Oceanian countries from the simulations due to their small sample size. Finally, we show which countries have missing irrigation water withdrawal values due to the pairing of the datasets, and print the list of countries that show at least one missing value.

```
# MERGE DATASETS ----------------------------------------------------------

full.dt <- water.dt[, merge(.SD, meier.dt, by = c("Country", "Code", "Continent"),
                            all.y = TRUE), Water.Dataset] %>%
  .[!Continent == "Oceania"]

# Show countries with missing values in Water Withdrawal
full.dt[is.na(Water.Withdrawn), ] %>%
  .[, unique(Country), Water.Dataset] %>%
  print(n = Inf)
```

```
##       Water.Dataset                   V1
##  1: Liu et al. 2016          Saint Lucia
##  2: Liu et al. 2016            Swaziland
##  3:        Aquastat               Belize
##  4:        Aquastat Bosnia & Herzegovina
##  5:        Aquastat           Cape Verde
##  6:        Aquastat              Croatia
##  7:        Aquastat                 Cuba
```

```
##  8:        Aquastat          El Salvador
##  9:        Aquastat              Grenada
## 10:        Aquastat               Guyana
## 11:        Aquastat                Haiti
## 12:        Aquastat              Ireland
## 13:        Aquastat              Lebanon
## 14:        Aquastat           Luxembourg
## 15:        Aquastat             Malaysia
## 16:        Aquastat               Panama
## 17:        Aquastat                 Peru
## 18:        Aquastat          Saint Lucia
## 19:        Aquastat                Sudan
## 20:        Aquastat             Suriname
## 21:        Aquastat            Swaziland
## 22:        Aquastat    Trinidad & Tobago
## 23:        Aquastat              Uruguay
## 24:           LPJmL              Grenada
## 25:           LPJmL                Malta
## 26:           LPJmL          Saint Lucia
## 27:           LPJmL           Seychelles
## 28:           LPJmL            Swaziland
## 29:      PCR-GLOBWB            Argentina
## 30:      PCR-GLOBWB               Canada
## 31:      PCR-GLOBWB              Grenada
## 32:      PCR-GLOBWB                Malta
## 33:      PCR-GLOBWB             Portugal
## 34:      PCR-GLOBWB               Russia
## 35:      PCR-GLOBWB          Saint Lucia
## 36:      PCR-GLOBWB           Seychelles
## 37:      PCR-GLOBWB            Swaziland
## 38:             H08              Grenada
## 39:             H08                Malta
## 40:             H08          Saint Lucia
## 41:             H08           Seychelles
## 42:             H08            Swaziland
## 43:        WaterGap              Grenada
## 44:        WaterGap                Malta
## 45:        WaterGap          Saint Lucia
## 46:        WaterGap           Seychelles
## 47:        WaterGap            Swaziland
## 48:            DBHM              Grenada
## 49:            DBHM                Malta
## 50:            DBHM            Mauritius
## 51:            DBHM          Saint Lucia
## 52:            DBHM           Seychelles
## 53:            DBHM            Swaziland
## 54:          MPI-HM              Grenada
## 55:          MPI-HM                Malta
```

```
## 56:          MPI-HM          Saint Lucia
## 57:          MPI-HM           Seychelles
## 58:          MPI-HM            Swaziland
## 59:             VIC              Grenada
## 60:             VIC                Malta
## 61:             VIC          Saint Lucia
## 62:             VIC           Seychelles
## 63:             VIC            Swaziland
## 64:           CLM45           Cape Verde
## 65:           CLM45              Grenada
## 66:           CLM45                Malta
## 67:           CLM45          Saint Lucia
## 68:           CLM45           Seychelles
## 69:           CLM45            Swaziland
##       Water.Dataset                   V1
```

```
# Show unique countries missing
full.dt[is.na(Water.Withdrawn), ] %>%
  .[, unique(Country)]
```

```
##  [1] "Saint Lucia"          "Swaziland"            "Belize"
##  [4] "Bosnia & Herzegovina" "Cape Verde"           "Croatia"
##  [7] "Cuba"                 "El Salvador"          "Grenada"
## [10] "Guyana"               "Haiti"                "Ireland"
## [13] "Lebanon"              "Luxembourg"           "Malaysia"
## [16] "Panama"               "Peru"                 "Sudan"
## [19] "Suriname"             "Trinidad & Tobago"    "Uruguay"
## [22] "Malta"                "Seychelles"           "Argentina"
## [25] "Canada"               "Portugal"             "Russia"
## [28] "Mauritius"
```

Now we plot irrigated areas against irrigation water withdrawals.

```
# PLOT ------------------------------------------------------------------------

full.dt %>%
  na.omit() %>%
  ggplot(., aes(Irrigated.Area, Water.Withdrawn,
                color = Continent)) +
  geom_point(size = 0.4) +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10 ^ (4 * x)),
                labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                labels = trans_format("log10", math_format(10 ^ .x))) +
  labs(x = "Irrigated area (ha)",
       y = expression(paste("Water withdrawal ", " ", "(", 10^9, m^3/year, "", ")"))) +
  facet_wrap(~Water.Dataset, ncol = 5) +
  theme_AP() +
  theme(legend.position = "top",
```

```
        strip.background = element_rect(fill = "white"))
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

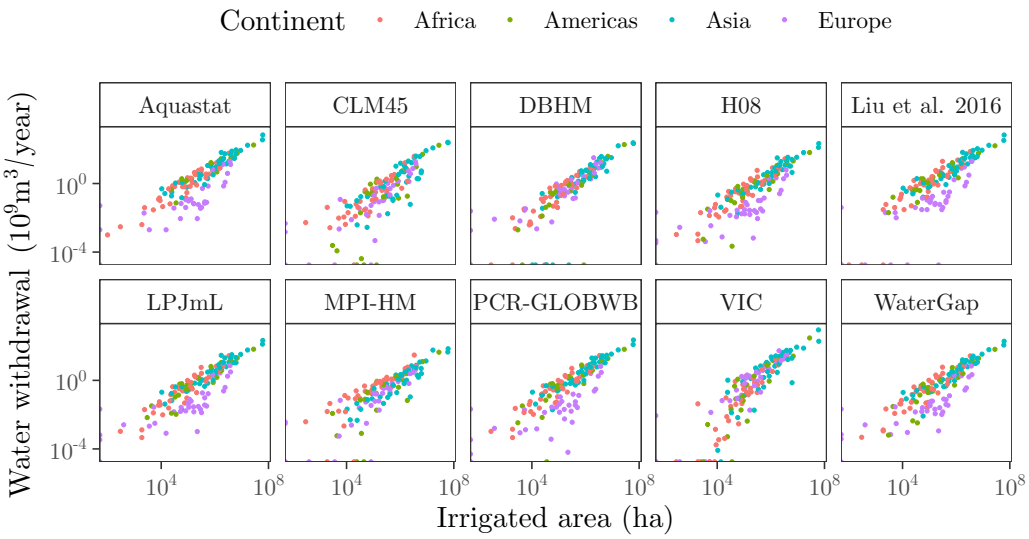


Figure 1: Scatterplots of irrigated areas reported by FAO-GMIA against irrigation water withdrawals. Each dot is a country.

2.5 Check influence of other parameters

In this section we prove that other parameters used by GM to estimate irrigation water withdrawals, shuch as total evapotranspiration, potential evaporation and irrigation efficiency, are computationally irrelevant. The data on potential evaporation and total evapotranspiration is retrieved from ISIMIP, and the data for irrigation efficiency from Rohwer, Gerten, and Lucht (2007).

After normalizing irrigation water withdrawal values by irrigated areas, we create different functions to load the data and create plots of (irrigation water withdrawals / irrigated areas) against each of these parameters.

```
# CHECK THE INFLUENCE OF POTENTIAL EVAPOTRANSPIRATION --------------------------

full.dt <- full.dt[, div:= Water.Withdrawn / Irrigated.Area]

# Create function to access the evapotranspiration data from ISIMIP
open_nc_totevap <- function(file, dname) {
ncin <- nc_open(file)
# get longitude, latitude, time
lon <- ncvar_get(ncin, "lon")
lat <- ncvar_get(ncin, "lat")
# Get variable
tmp_array <- ncvar_get(ncin, dname)
m <- (dim(tmp_array)[3] - 11):dim(tmp_array)[3]
tmp_slice <- lapply(m, function(m) tmp_array[, , m])
```

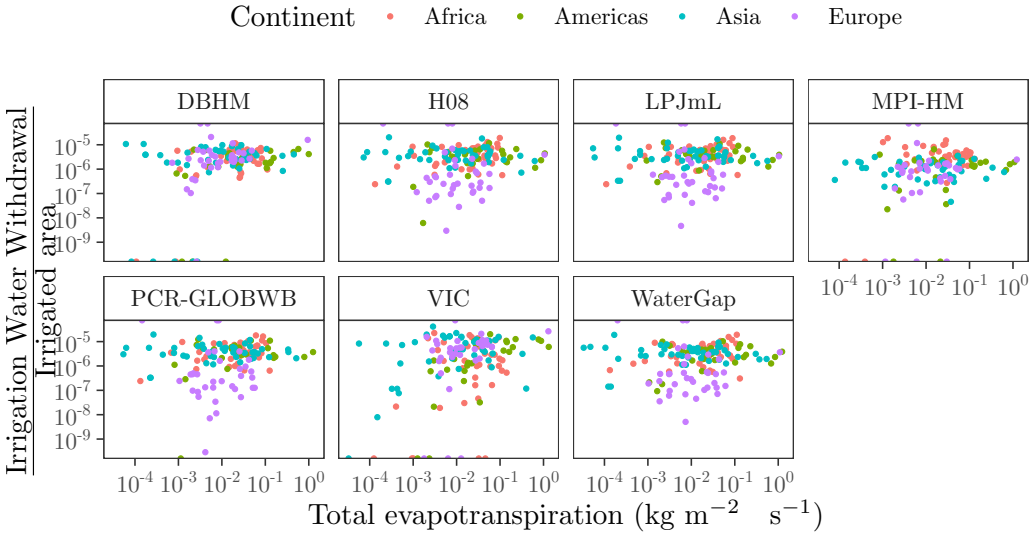
```
lonlat <- as.matrix(expand.grid(lon,lat))
# vector of `tmp` values
tmp_vec <- lapply(tmp_slice, function(x) as.vector(x))
# create dataframe and add names
tmp_df01 <- lapply(tmp_vec, function(x) data.frame(cbind(lonlat, x)))
names(tmp_df01) <- m
da <- lapply(tmp_df01, data.table) %>%
  rbindlist(., idcol = "month") %>%
  na.omit()
# Convert coordinates to country
countries <- coords2country(da[1:nrow(da), 2:3])
dt <- cbind(countries, da)
setDT(dt)
dt <- na.omit(dt)[, .(totevap = sum(x)), countries]
setnames(dt, "countries", "Country")
dt[, `:=` (Code = countrycode(dt[, Country],
                              origin = "country.name",
                              destination = "un"),
           Continent = countrycode(dt[, Country],
                                   origin = "country.name",
                                   destination = "continent"))]
dt[, Country:= countrycode(dt[, Code],
                           origin = "un",
                           destination = "country.name")]
return(dt)
}

# List of files to access
files <- list("watergap2_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc",
              "pcr-globwb_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc",
              "mpi-hm_wfdei_nobc_hist_pressoc_co2_evap_global_monthly_1971_2010.nc",
              "lpjml_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc",
              "h08_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc",
              "vic_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc",
              "dbh_wfdei_nobc_hist_varsoc_co2_evap_global_monthly_1971_2010.nc")

# Retrieve and arrange the data
dname <- "evap"
totevap.dt <- mclapply(files, function(x)
  open_nc_totevap(file = x,
                  dname = dname),
  mc.cores = detectCores() * 0.75)
gm_list <- c("WaterGap", "PCR-GLOBWB", "MPI-HM", "LPJmL", "H08", "VIC", "DBHM")
names(totevap.dt) <- gm_list
```

```
# PLOT EVAPOTRANSPIRATION ----------------------------------------------------
```

```
rbindlist(totevap.dt, idcol = "Water.Dataset") %>%
  na.omit() %>%
  merge(full.dt[Water.Dataset %in% gm_list], ., by = c("Continent", "Country",
                                                       "Code", "Water.Dataset")) %>%
  ggplot(., aes(totevap, div,
                color = Continent)) +
  geom_point(size = 0.6) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  labs(x = "Total evapotranspiration (kg $\\textrm{m}^{-2} \\quad \\textrm{s}^{-1}$)",
       y = "$\\frac{\\textrm{Irrigation Water Withdrawal}}{\\textrm{Irrigated area}}$") +
  facet_wrap(~Water.Dataset, ncol = 4) +
  theme_AP() +
  theme(legend.position = "top",
        strip.background = element_rect(fill = "white"))
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 16 rows containing missing values (geom_point).
```



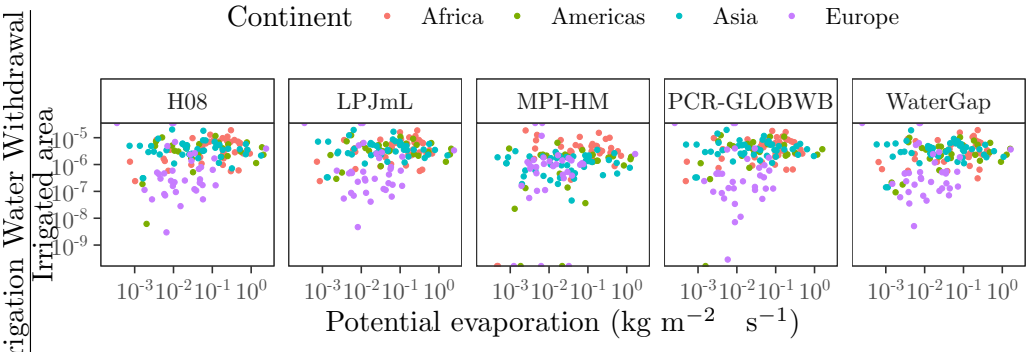

```
# CHECK THE INFLUENCE OF POTENTIAL EVAPORATION ---------------------------------

files <- c("watergap2_wfdei_nobc_hist_pressoc_co2_potevap_global_monthly_1971_2010.nc",
           "h08_wfdei_nobc_hist_pressoc_co2_potevap_global_monthly_1971_2010.nc",
           "pcr-globwb_wfdei_nobc_hist_pressoc_co2_potevap_global_monthly_1971_2010.nc",
           "mpi-hm_wfdei_nobc_hist_pressoc_co2_potevap_global_monthly_1971_2010.nc",
           "lpjml_wfdei_nobc_hist_pressoc_co2_potevap_global_monthly_1971_2010.nc")

# Retrieve and arrange the data
dname <- "potevap"
potevap.dt <- mclapply(files, function(x)
  open_nc_totevap(file = x,
                  dname = dname),
```

```
  mc.cores = detectCores() * 0.75)
gm_list <- c("WaterGap", "H08", "PCR-GLOBWB", "MPI-HM", "LPJmL")
names(potevap.dt) <- gm_list
```

```
# PLOT POTENTIAL EVAPORATION ----------------------------------------------------

rbindlist(potevap.dt, idcol = "Water.Dataset") %>%
  na.omit() %>%
  merge(full.dt[Water.Dataset %in% gm_list], ., by = c("Continent", "Country",
                                                       "Code", "Water.Dataset")) %>%
  ggplot(., aes(totevap, div,
                color = Continent)) +
  geom_point(size = 0.6) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  labs(x = "Potential evaporation (kg $\\textrm{m}^{-2} \\quad  \\textrm{s}^{-1}$)",
       y = "$\\frac{\\textrm{Irrigation Water Withdrawal}}{\\textrm{Irrigated area}}$") +
  facet_wrap(~Water.Dataset, ncol = 5) +
  theme_AP() +
  theme(legend.position = "top",
        strip.background = element_rect(fill = "white"))
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

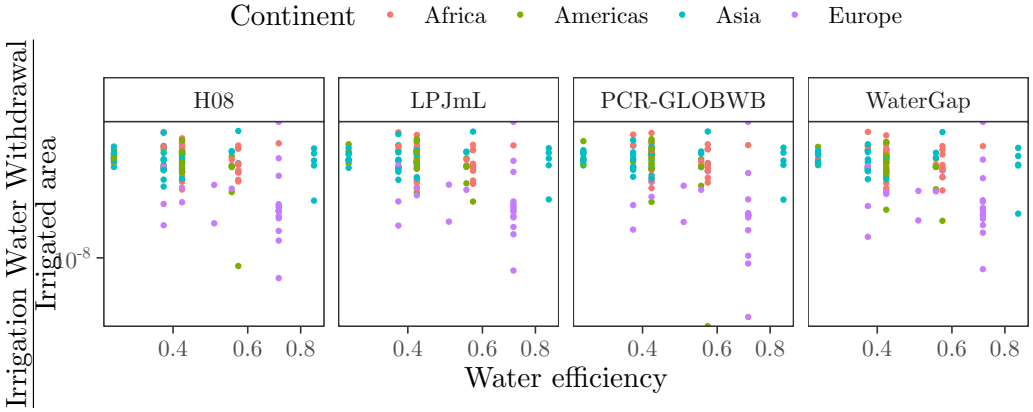
```
## Warning: Removed 10 rows containing missing values (geom_point).
```




```
# CHECK INFLUENCE OF WATER EFFICIENCY -------------------------------------------

# Retrieve the data from Rohwer
rohwer_data <- fread("rohwer_data.csv")

# Plot
merge(full.dt[Water.Dataset %in% c("WaterGap", "H08", "LPJmL",
                                   "PCR-GLOBWB")],
      rohwer_data, by = "Country") %>%
  na.omit() %>%
  ggplot(., aes(Project.efficiency, div,
                color = Continent)) +
```

```
  geom_point(size = 0.6) +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_x_log10(breaks = pretty_breaks(n = 3)) +
  labs(x = "Water efficiency",
       y = "$\\frac{\\textrm{Irrigation Water Withdrawal}}{\\textrm{Irrigated area}}$") +
  facet_wrap(~Water.Dataset, ncol = 4) +
  theme_AP() +
  theme(legend.position = "top",
        strip.background = element_rect(fill = "white"))
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```



3 Creation of the final dataset

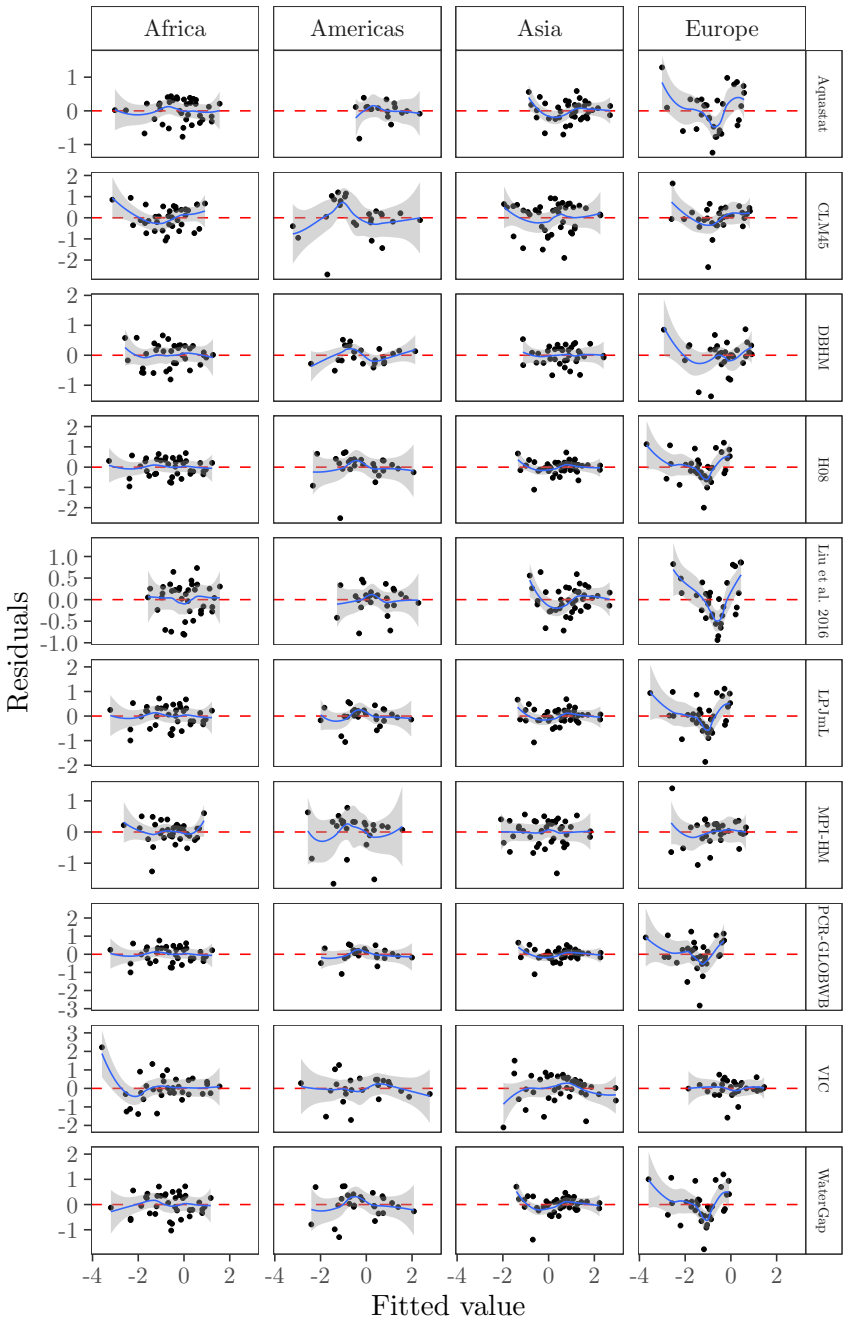
In order to set the ground for the computations, which will be launched hereafter, we now log-transform both irrigation water withdrawals and irrigated area values. Afterwards, we regress irrigated areas against irrigation water withdrawals for each continent and water dataset, and show three diagnostic plots that will help assessing the apropriateness of the methodological approach. Finally, we export all files created to `.csv`.

```
# TRANSFORM DATASET ------------------------------------------------------------

cols <- c("Water.Withdrawn", "Irrigated.Area")
col_names <- c("Continent", "Water.Dataset", "Area.Dataset", "Regression",
               "Imputation.Method", "Iteration")
cols_group <- c("Continent", "Water.Dataset")
full.dt <- full.dt[, (cols):= lapply(.SD, log10), .SDcols = (cols)]
full.dt <- full.dt[, Country:= str_replace(Country, "\\&", "and")]
```

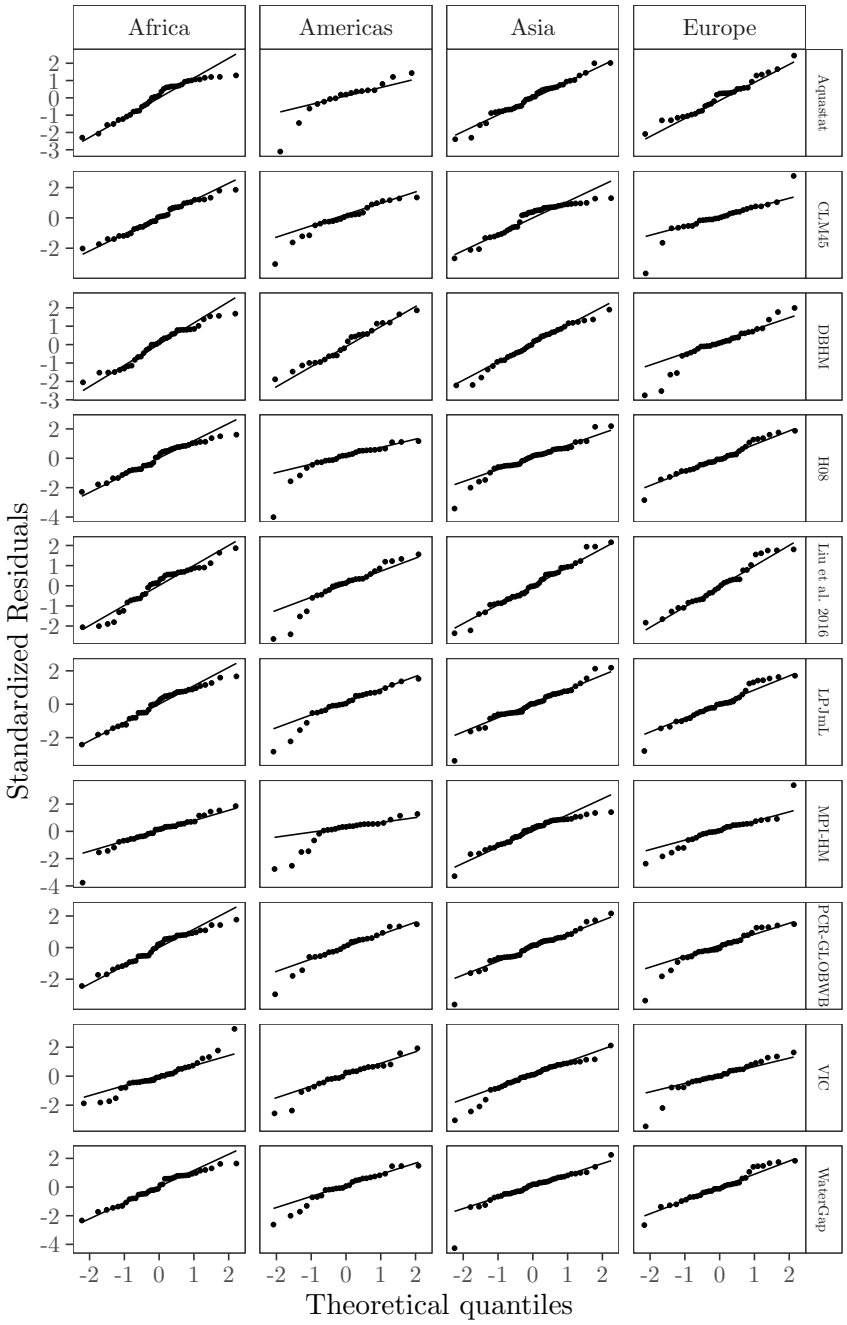
```
# REGRESSION DIAGNOSTICS -------------------------------------------------------

# Conduct regressions
regression.diag <- full.dt %>%
  NaRV.omit() %>%
  group_by(Continent, Water.Dataset) %>%
```

```
  nest() %>%
  mutate(fit = map(.x = data, .f = ~lm(Water.Withdrawn ~ Irrigated.Area, data = .)),
         results = map(fit, glance),
         residuals = map(fit, augment))

# Prepare dataset
diagnostics.dt <- regression.diag %>%
  dplyr::select(Continent, Water.Dataset, residuals) %>%
  unnest(residuals) %>%
  data.table()

# Residuals versus fitted
ggplot(diagnostics.dt, aes(.fitted, .resid)) +
  geom_point(size = 0.5) +
  geom_hline(yintercept = 0, lty = 2, color = "red") +
  geom_smooth(size = 0.5) +
  labs(x = "Fitted value", y = "Residuals") +
  facet_grid(Water.Dataset ~ Continent,
             scales = "free_y") +
  theme_AP() +
  theme(strip.text.y = element_text(size = 5.5),
        strip.background = element_rect(fill = "white"))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



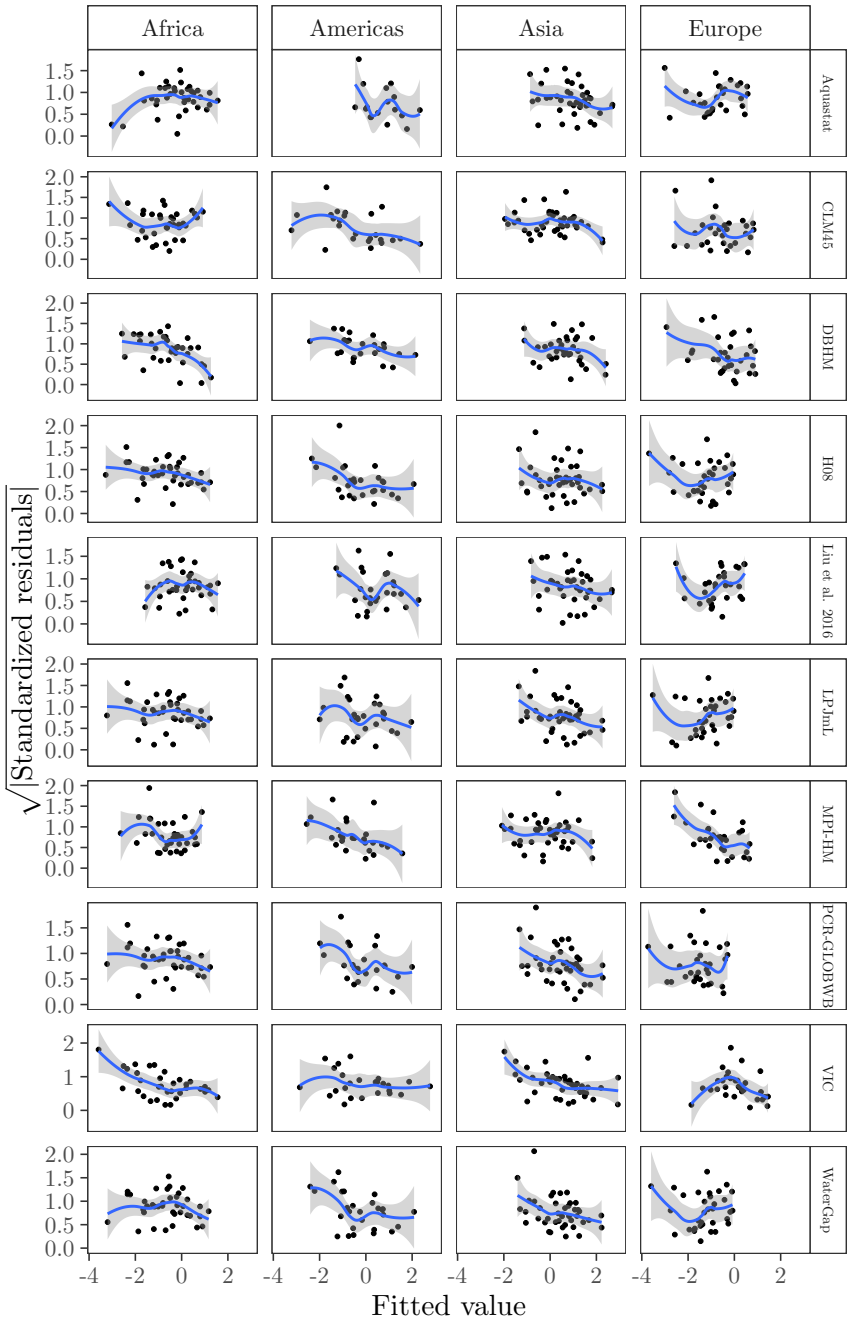

```
# QQ plot
ggplot(diagnostics.dt, aes(sample = .std.resid)) +
  stat_qq(size = 0.5) +
  stat_qq_line() +
  facet_grid(Water.Dataset ~ Continent,
             scales = "free_y") +
  labs(x = "Theoretical quantiles",
       y = "Standardized Residuals") +
  theme_AP() +
  theme(strip.text.y = element_text(size = 5.5),
        strip.background = element_rect(fill = "white"))
```




```
# Scale location plot
ggplot(diagnostics.dt, aes(.fitted, sqrt(abs(.std.resid)))) +
  geom_point(size = 0.5) +
  geom_smooth() +
  facet_grid(Water.Dataset ~ Continent,
             scales = "free_y") +
  labs(x = "Fitted value",
       y = "$\\sqrt{|\\textrm{Standardized residuals}|}$") +
  theme_AP() +
```

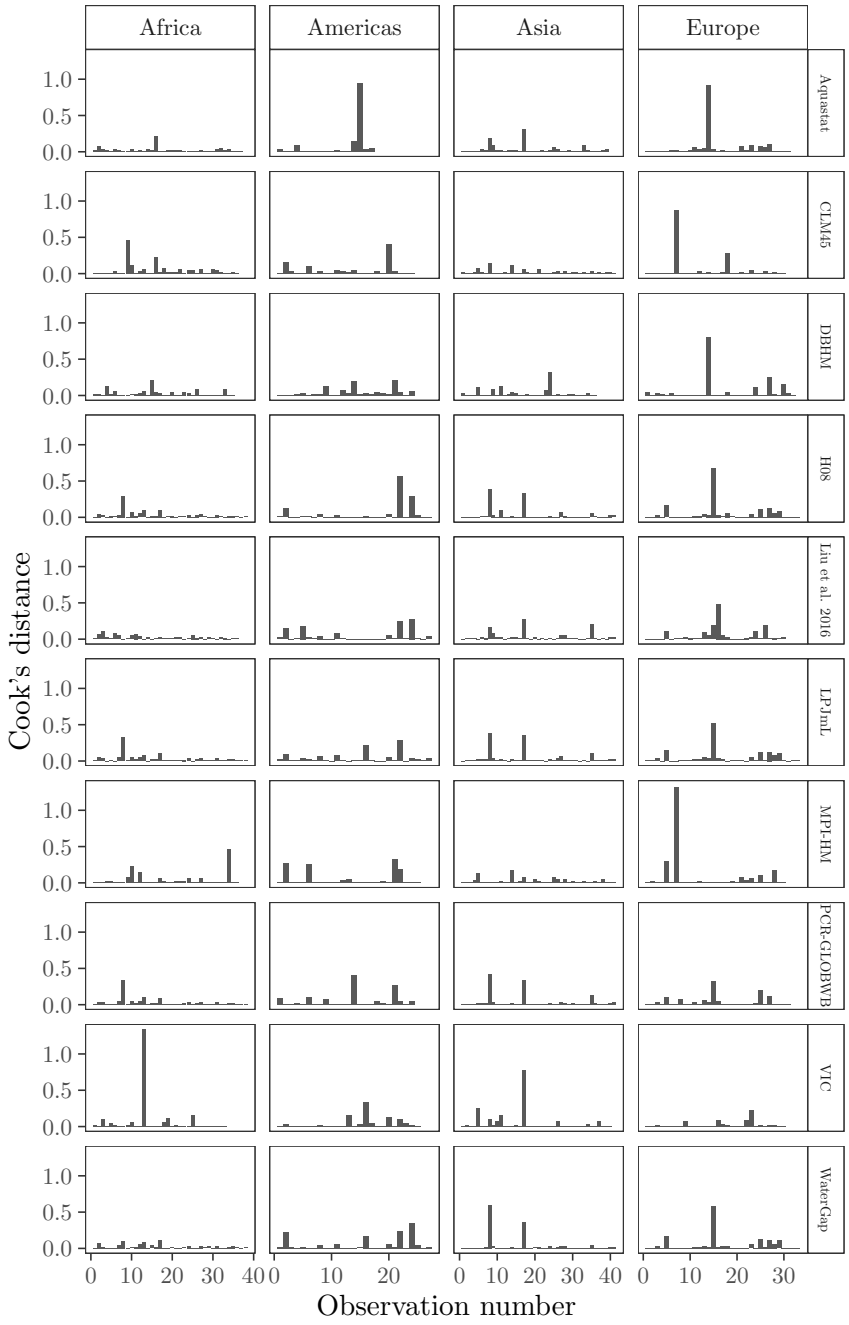
```
  theme(strip.text.y = element_text(size = 5.5),
        strip.background = element_rect(fill = "white"))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
# Cook's distance
ID <- diagnostics.dt[, .N, .(Continent, Water.Dataset)]
diagnostics.dt <- diagnostics.dt[, ID:= unlist(sapply(ID[, N], function(x) 1:x))]
ggplot(diagnostics.dt, aes(ID, .cooksd)) +
  geom_col() +
  facet_grid(Water.Dataset ~ Continent,
```

```
            scales = "free_x") +
  theme_AP() +
  labs(x = "Observation number", y = "Cook's distance") +
  theme(strip.text.y = element_text(size = 5.5),
        strip.background = element_rect(fill = "white"))
```




```
# EXPORT FULL DATASET WITH MISSING VALUES ------------------------------------

fwrite(full.dt, "full.dt.csv")
fwrite(water.dt, "water.dt.csv")
```

4 Missing values

Here we show the fraction of missing values in each GHM and FAO-based dataset.

```
# PLOT PERCENTAGE OF MISSING 2 ------------------------------------------------

full.dt[, sum(is.na(.SD) == TRUE) / .N,
        .(Continent, Water.Dataset),
        .SDcols = "Water.Withdrawn"] %>%
  ggplot(., aes(Continent, V1, fill = Water.Dataset)) +
  geom_bar(stat = "identity",
           position = position_dodge(0.7),
           color = "black") +
  labs(x = "",
       y = "Proportion of missing values") +
  scale_fill_discrete(name = "") +
  theme_AP() +
  theme(legend.position = "top")
```

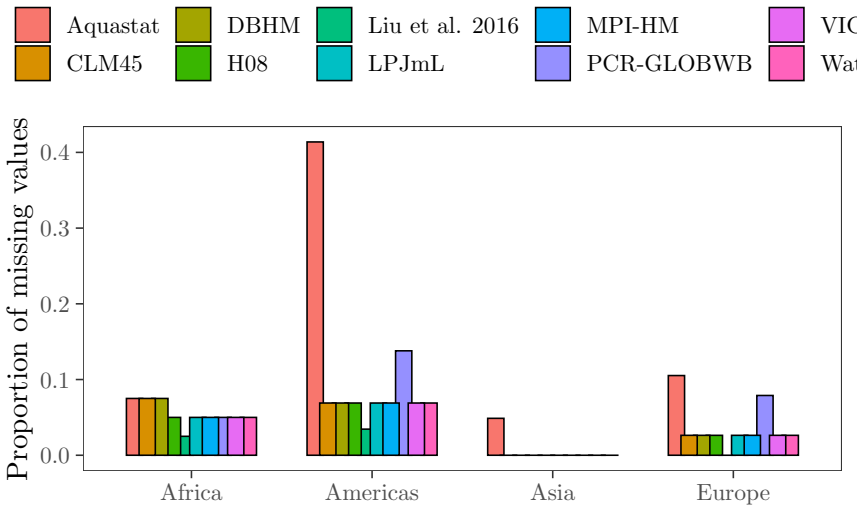


Figure 2: Proportion of missing values in irrigation water withdrawal per dataset.

Now we impute the missing values for each continent and irrigation water withdrawal dataset. Given the linear trend between water withdrawals and irrigated areas shown in Figure 1, we select three possible regression-based multiple imputation methods: bayesian, without model error and bootstrapped. For each method and missing value, we compute 40 different iterations to account for the uncertainty in the true missing value. The result is the dataset `full.imput`, which only contains complete cases.

```
# IMPUTATION OF MISSING VALUES ------------------------------------------------

# Substitute Inf values for NA
for (j in 1:ncol(full.dt)) set(full.dt, which(is.infinite(full.dt[[j]])), j, NA)

full.dt[, lapply(.SD, function(x) sum(is.infinite(x)))] # Check
```

```
##    Water.Dataset Country Code Continent Water.Withdrawn Irrigated.Area div
## 1:             0       0    0         0               0              0   0
```

```
# Imputation settings
m.iterations <- 40
imputation.methods <- c("norm.boot", "norm", "norm.nob")

# remove div columns to better prepare dataset for mice
full.dt <- full.dt[, div:= NULL]

# Run
imput <- full.dt[, .(Group = lapply(imputation.methods, function(x)
  mice(.SD, m = m.iterations, maxit = m.iterations, method = x, seed = 500,
       print = FALSE))),
  cols_group]

imput <- imput[, Imputation.Method:= rep(imputation.methods, .N /
                                           length(imputation.methods))]

# Extract iterations
imput <- imput[, Datasets:= lapply(Group, function(x)
  lapply(1:m.iterations, function(y) data.table(mice::complete(x, y))))] %>%
  .[, Data:= lapply(Datasets, function(x) rbindlist(x, idcol = "Iteration"))]

# Vector to loop onto
columns_add <- c("Country", "Iteration", "Irrigated.Area", "Water.Withdrawn")
tmp <- as.list(columns_add)
names(tmp) <- columns_add

# Extract columns
for(i in names(tmp)) {
  imput <- imput[, tmp[[i]]:= lapply(.SD, function(x)
    lapply(x, function(y) y[, ..i])), .SDcols = "Data"]
}

# Unlist
full.imput <- imput[, lapply(.SD, unlist),
                    .SDcols = columns_add,
                    .(Continent,Water.Dataset, Imputation.Method)]
```

5 Regressions

The following code snippet computes all possible regressions for all combinations of irrigation water withdrawals datasets (10), imputation methods (3), iterations (40) and regression method (robust/non-robust, 2). These four uncertain factors will be the triggers that we will use in our model to select the r^2 value (see below). In total, we have 10 * 3 * 40 * 2 = 2400 possible regressions and r^2 values for each continent. The result is the `regressions` dataset.

Then we extract `r.squared` values and other statistics from the `regressions` dataset, and create the `results` dataset.

```
# COMPUTE LINEAR REGRESSIONS ------------------------------------------------

# Compute regressions in each combination
# Settings for robust regressions
a1<-lmrob.control()
a1$k.max <- 1000

# Compute regressions
regressions <- full.imput %>%
  group_by(Continent, Water.Dataset, Imputation.Method, Iteration) %>%
  nest() %>%
  mutate(fit = map(.x = data, .f = ~lm(Water.Withdrawn ~ Irrigated.Area, data = .)),
         fit.robust = map(.x = data, .f = ~lmrob(Water.Withdrawn ~ Irrigated.Area, data = .,
                                                 control = a1)),
         results = map(fit, glance),
         results.robust = map(fit.robust, glance),
         residuals = map(fit, augment))
```

```
## Warning: Problem with `mutate()` input `fit.robust`.
## i M-step did NOT converge. Returning unconverged SM-estimate
## i Input `fit.robust` is `map(...)`.
## i The error occurred in group 1004: Continent = "Africa", Water.Dataset = "VIC", Imputation
```

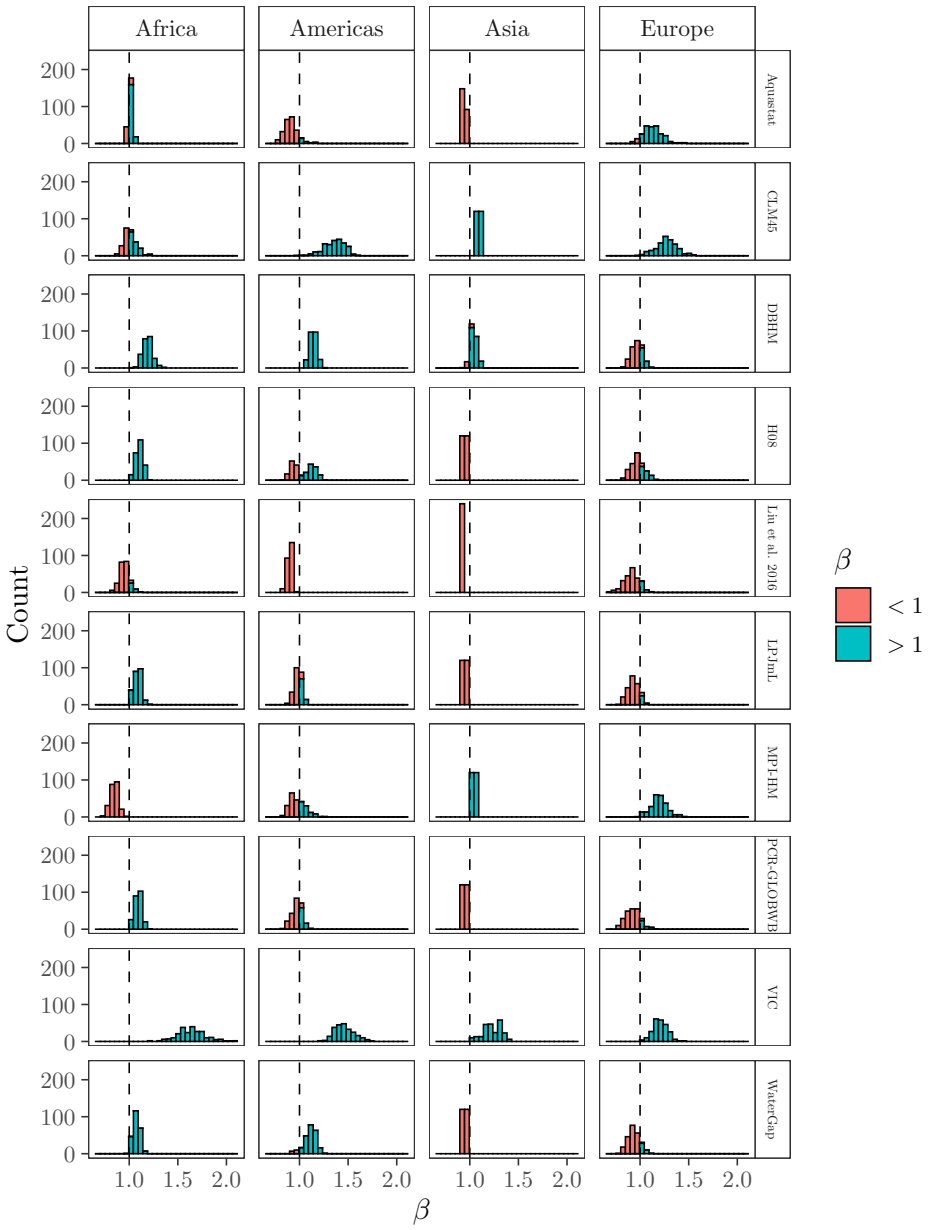
```
## Warning in lmrob.fit(x, y, control, init = init): M-step did NOT converge.
## Returning unconverged SM-estimate
```

```
# EXTRACT R SQUARED ----------------------
# Regular r squared
results <- regressions %>%
  dplyr::select(Continent, Water.Dataset,
                Imputation.Method, Iteration, results,) %>%
  unnest(results) %>%
  data.table() %>%
  .[, Regression:= "Normal"] %>%
  .[, index:= paste(Continent, Water.Dataset, Imputation.Method,
                    Iteration, Regression, sep = "_")]

# Robust r squared
results.robust <- regressions %>%
  dplyr::select(Continent, Water.Dataset,
                Imputation.Method, Iteration, results.robust) %>%
  unnest(results.robust) %>%
  data.table() %>%
  .[, Regression:= "Robust"] %>%
  .[, index:= paste(Continent, Water.Dataset, Imputation.Method,
                    Iteration, Regression, sep = "_")]
```

```
cols_extract <- c("Continent", "Water.Dataset", "Imputation.Method",
                  "Iteration", "Regression", "r.squared",
                  "index")

# Bind regular and robust
all.results <- rbind(results[, ..cols_extract],
                     results.robust[, ..cols_extract])
```

```
# EXTRACT SLOPE (BETA) ------------------------------------------------------

# Extract regular slope
slope <- regressions %>%
  mutate(slope = map(fit, tidy)) %>%
  unnest(slope) %>%
  dplyr::select(Continent, Water.Dataset,
                Imputation.Method, Iteration, term, estimate) %>%
  data.table() %>%
  .[term == "Irrigated.Area"]

# Extract robust slope
slope.robust <- regressions %>%
  mutate(slope = map(fit.robust, tidy)) %>%
  unnest(slope) %>%
  dplyr::select(Continent, Water.Dataset,
                Imputation.Method, Iteration, term, estimate) %>%
  data.table() %>%
  .[term == "Irrigated.Area"]

# Plot
rbind(slope, slope.robust) %>%
  ggplot(., aes(estimate)) +
  geom_histogram(color = "black", aes(fill = estimate > 1)) +
  geom_vline(xintercept = 1, lty = 2) +
  labs(x = "$\\beta$",
       y = "Count") +
  scale_fill_discrete(name = "$\\beta$",
                      labels = c("$<1$", "$>1$")) +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  facet_grid(Water.Dataset~Continent) +
  theme_AP() +
  theme(strip.text.y = element_text(size = 5),
        strip.background = element_rect(fill = "white"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The regressions just conducted allow to obtain the coefficients to predict irrigation water withdrawals from irrigated areas only. To check the efficiency of this approach, we predict irrigation water withdrawal values from the irrigated areas reported by FAO-GMIA (Siebert et al. 2013), and by five other irrigated area products: the IWMI-GIAM (Thenkabail et al. 2009), the GRIPC (Salmon et al. 2015), the Meier dataset (Meier, Zabel, and Mauser 2018), Aquastat (FAO 2016) and FAOSTAT (FAO 2017). We compare the resulting estimates with the irrigation water withdrawal estimates outputted by GHM and with those reported in FAO-based datasets.

Once the predictions are done, we create the `water.quantiles` dataset, which reports the min, median, max and several quantiles to describe the uncertainty in irrigation water withdrawals at the country level.

```
# PREDICT WATER WITHDRAWALS ----------------------------------------------------
```

```
size.gmia <- meier.dt[, .(Country, Continent, Irrigated.Area)] %>%
  .[!Irrigated.Area == 0] %>%
  .[, Irrigated.Area:= log10(Irrigated.Area)] %>%
  .[!Continent == "Oceania"] %>%
  .[order(Continent)]

countries <- split(size.gmia, size.gmia$Continent) %>%
  lapply(., function(x) x[, Country]) %>%
  lapply(., data.table)

areas <- split(size.gmia, size.gmia$Continent) %>%
  lapply(., function(x) x[, Irrigated.Area]) %>%
  lapply(., data.frame) %>%
  lapply(., function(x) setnames(x, "X..i..", "Irrigated.Area"))

tmp.regressions <- regressions %>%
  split(., .$Continent)

out <- out.robust <- list()
for(i in names(tmp.regressions)) {
  out[[i]] <- mutate(tmp.regressions[[i]],
                     pred = map(fit, .f = ~predict(., areas[[i]])))
  out.robust[[i]] <- mutate(tmp.regressions[[i]],
                            pred = map(fit.robust, .f = ~predict(., areas[[i]])))
}

water.predicted <- lapply(out, function(x) {
  select(x, Continent, Water.Dataset, Imputation.Method, Iteration, pred) %>%
  unnest(pred) %>%
  data.table()
})

water.predicted.rob <- lapply(out.robust, function(x) {
  select(x, Continent, Water.Dataset, Imputation.Method, Iteration, pred) %>%
    unnest(pred) %>%
    data.table()
})

out <- out.robust <- list()
for(i in names(water.predicted)) {
  out[[i]] <- water.predicted[[i]][, Country:= rep(countries[[i]][, V1],
                                                   times = nrow(water.predicted[[i]]) /
                                                           nrow(countries[[i]]))]
  out.robust[[i]] <- water.predicted.rob[[i]][, Country:= rep(countries[[i]][, V1],
                                                              times = nrow(water.predicted[[i]]) /
                                                                nrow(countries[[i]]))]
}
```

```
water.predicted <- rbindlist(water.predicted) %>%
  .[, pred:= 10 ^ pred]

water.predicted.rob <- rbindlist(water.predicted.rob) %>%
  .[, pred:= 10 ^ pred]

# Compute quantiles
water.quantiles <- rbind(water.predicted, water.predicted.rob) %>%
  .[, .(min = min(pred),
                                      max = max(pred),
                                      q0.025 = quantile(pred, 0.025),
                                      q0.1 = quantile(pred, 0.1),
                                      q0.25 = quantile(pred, 0.25),
                                      q0.5 = quantile(pred, 0.5),
                                      q0.75 = quantile(pred, 0.75),
                                      q0.975 = quantile(pred, 0.975),
                                      q0.99 = quantile(pred, 0.99),
                                      q1 = quantile(pred, 1),
                                      mean = mean(pred),
                                      median = median(pred)),
                                  .(Continent, Country)] %>%
  .[order(Country, Continent)]

water.quantiles <- water.quantiles[, Country:= str_replace(Country, "\\&", "and")]
```

And now we create a plot to show how our estimates compare with the results yielded by GHM and those reported in FAO-based datasets (Figures 3–6 below).

```
# PLOT PREDICTIONS AGAINST GHM AND FAO OUTPUTS ---------------------------------

water.tmp <- water.dt[Country %in% water.quantiles[, Country]]
Cont <- c("Africa", "Americas", "Asia", "Europe")
gg <- list()
for(i in Cont) {
  gg[[i]] <- water.quantiles[Continent == i] %>%
    ggplot(., aes(reorder(Country, median), median)) +
    geom_point() +
    geom_point(data = water.tmp[Continent == i],
               aes(Country, Water.Withdrawn, color = Water.Dataset)) +
    geom_errorbar(aes(ymin = min,
                      ymax = max)) +
    scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                  labels = trans_format("log10", math_format(10 ^ .x))) +
    scale_color_discrete(name = "Water dataset") +
    labs(y = expression(paste("Water withdrawal ", " ", "(", 10^9, m^3/year, "", ")")),
         x = "") +
    coord_flip() +
```

```
    theme_AP()
}

all.plots <- lapply(1:4, function(x)
  gg[[x]] +
    theme(legend.position = "none") +
    labs(x = "", y = ""))

legend <- get_legend(gg[[1]] + theme(legend.position = "top",
                                     legend.key.size = unit(0.5, 'lines')))
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
bottom1 <- plot_grid(all.plots[[1]], all.plots[[2]], ncol = 2, labels = "auto", align = "hv")
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
all1 <- plot_grid(legend, bottom1, ncol = 1, rel_heights = c(0.1, 1))

grid.arrange(arrangeGrob(all1, bottom = grid::textGrob(
  label = expression(paste("Irrigation water withdrawal ", " ", "(", 10^9, m^3/year, "", ")")))
```

```
bottom2 <- plot_grid(all.plots[[3]], all.plots[[4]], ncol = 2, labels = "auto", align = "hv")
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
all <- plot_grid(legend, bottom2, ncol = 1, rel_heights = c(0.1, 1))

grid.arrange(arrangeGrob(all, bottom = grid::textGrob(
  label = expression(paste("Irrigation water withdrawal ", " ", "(", 10^9, m^3/year, "", ")")))
```

```
# CHECK HOW MANY ESTIMATES FIT WITHIN THE REGRESSION BOUNDS ----------------------

da <- merge(water.tmp, water.quantiles[, .(Continent, Country, min, max)],
            by = c("Continent", "Country")) %>%
  .[, fit:= ifelse(Water.Withdrawn >= min & Water.Withdrawn <= max, TRUE, FALSE)]

# Check which GHM or FAO-based dataset shows more estimates
# beyond or below our predictions
da[, .(N = sum(Water.Withdrawn < min | Water.Withdrawn > max),
       prop = sum(Water.Withdrawn < min | Water.Withdrawn > max) / .N),
   .(Water.Dataset, Continent)]
```

```
##       Water.Dataset Continent  N       prop
##  1: Liu et al. 2016    Africa 18 0.46153846
##  2:        Aquastat    Africa 13 0.35135135
```

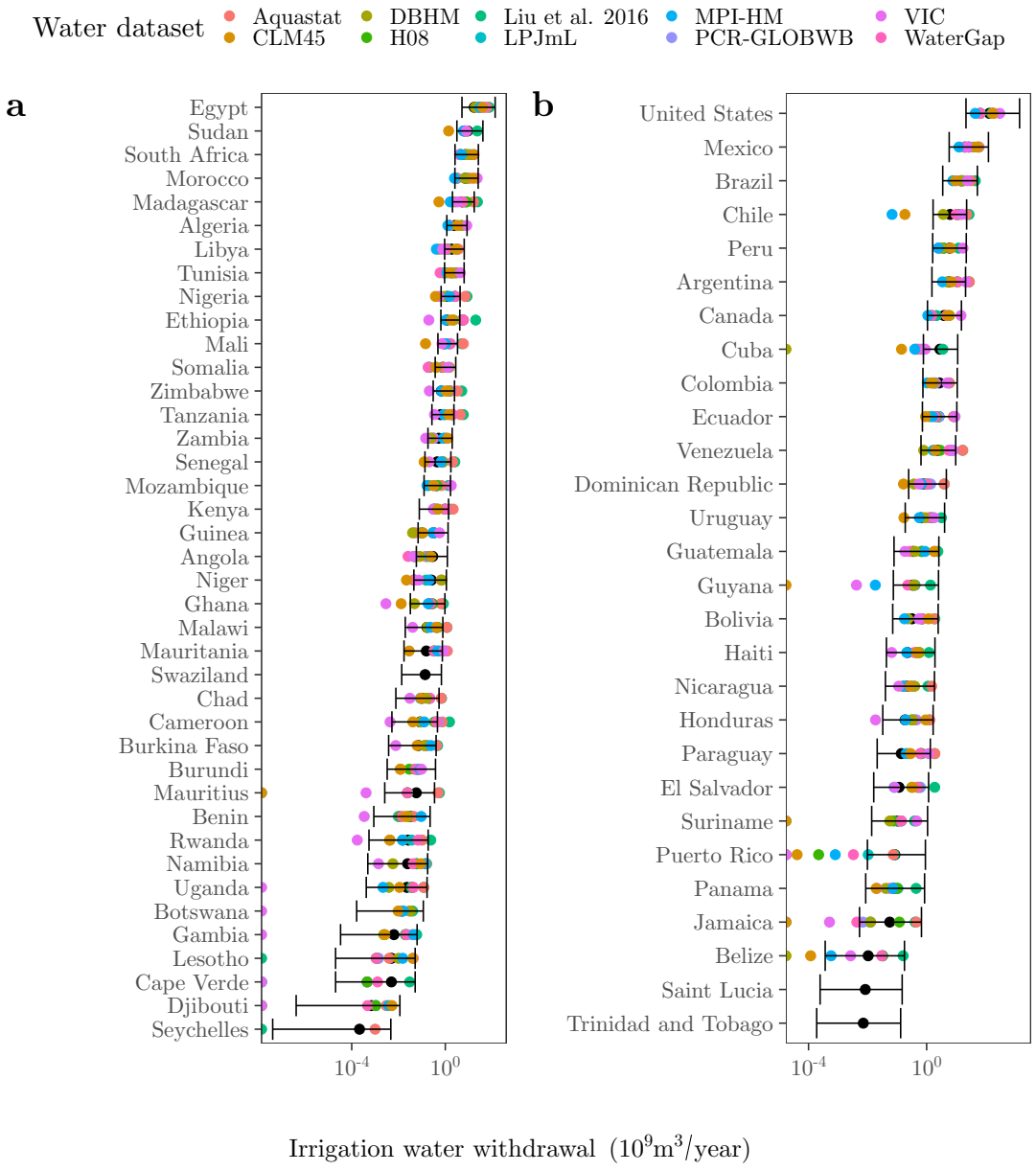


Figure 3: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models (DBHM, Ho8, LPJmL, MPI-HM, PCR-GLOBWB, WaterGap) and FAO-based datasets (Aquastat, Liu et al. 2016).

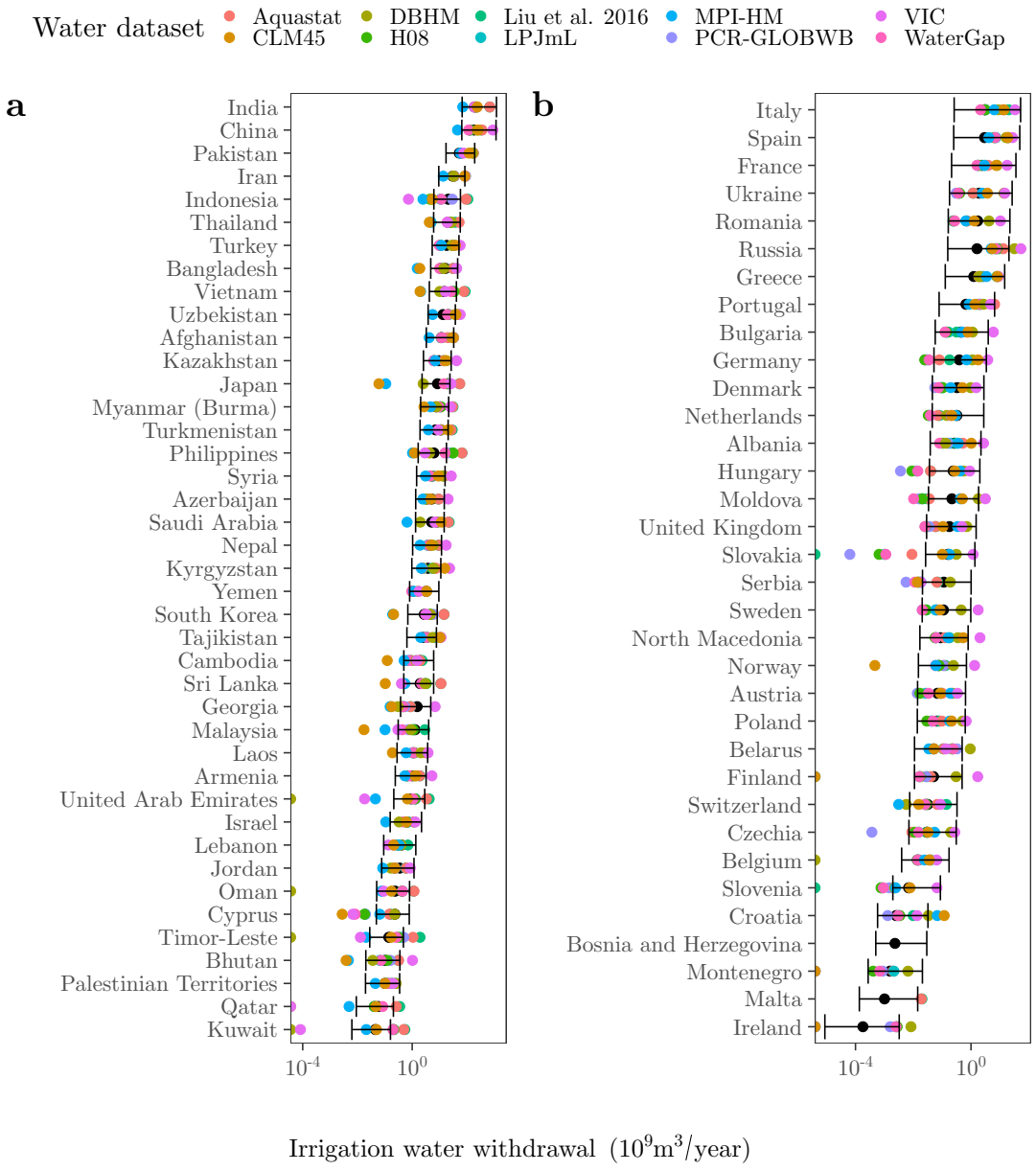
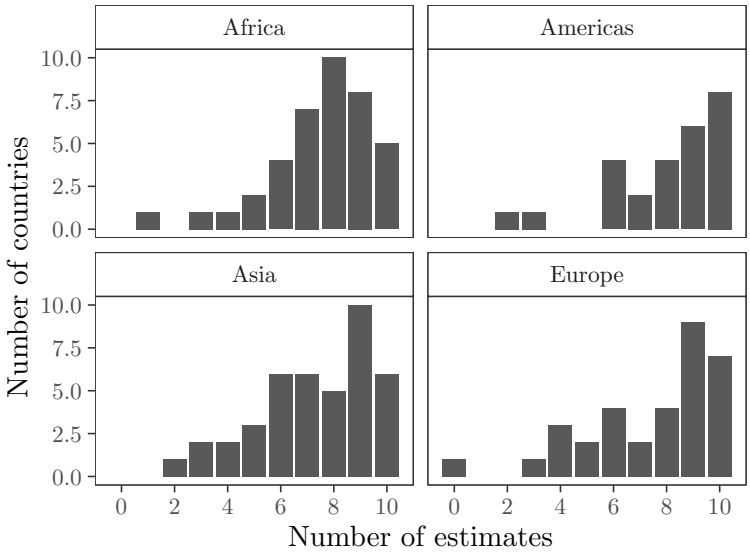


Figure 4: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models (DBHM, Ho8, LPJmL, MPI-HM, PCR-GLOBWB, WaterGap) and FAO-based datasets (Aquastat, Liu et al. 2016).

```
##  3:           LPJmL    Africa  4 0.10526316
##  4:      PCR-GLOBWB    Africa  5 0.13157895
##  5:             H08    Africa  5 0.13157895
##  6:        WaterGap    Africa  5 0.13157895
##  7:            DBHM    Africa  4 0.10810811
##  8:          MPI-HM    Africa  5 0.13157895
##  9:             VIC    Africa 18 0.47368421
## 10:           CLM45    Africa  8 0.21621622
## 11: Liu et al. 2016  Americas  5 0.19230769
## 12:        Aquastat  Americas  4 0.23529412
## 13:           LPJmL  Americas  1 0.03846154
## 14:             H08  Americas  2 0.07692308
## 15:        WaterGap  Americas  3 0.11538462
## 16:            DBHM  Americas  3 0.11538462
## 17:          MPI-HM  Americas  6 0.23076923
## 18:             VIC  Americas  5 0.19230769
## 19:           CLM45  Americas  9 0.34615385
## 20:      PCR-GLOBWB  Americas  2 0.08333333
## 21: Liu et al. 2016      Asia 16 0.39024390
## 22:        Aquastat      Asia 17 0.43589744
## 23:           LPJmL      Asia  3 0.07317073
## 24:      PCR-GLOBWB      Asia  4 0.09756098
## 25:             H08      Asia  4 0.09756098
## 26:        WaterGap      Asia  2 0.04878049
## 27:            DBHM      Asia  7 0.17073171
## 28:          MPI-HM      Asia 16 0.39024390
## 29:             VIC      Asia 23 0.56097561
## 30:           CLM45      Asia 17 0.41463415
## 31: Liu et al. 2016    Europe  6 0.18181818
## 32:        Aquastat    Europe  2 0.06451613
## 33:           LPJmL    Europe  7 0.21875000
## 34:      PCR-GLOBWB    Europe 10 0.33333333
## 35:             H08    Europe  8 0.25000000
## 36:        WaterGap    Europe  8 0.25000000
## 37:            DBHM    Europe  6 0.18750000
## 38:          MPI-HM    Europe  6 0.18750000
## 39:             VIC    Europe 13 0.40625000
## 40:           CLM45    Europe  6 0.18750000
##       Water.Dataset Continent  N       prop
```

```
# Plot bars
prove <- da[, sum(fit), .(Country, Continent)]

ggplot(prove, aes(V1)) +
  geom_bar() +
  facet_wrap(~Continent) +
  theme_AP() +
  labs(x = "Number of estimates",
```

```
       y = "Number of countries") +
  scale_x_continuous(breaks = seq(0, 10, 2)) +
  theme(strip.background = element_rect(fill = "white"))
```




```
# Check how many countries show more than 7 or 10 estimates
# bounded by our predictions
lapply(c(7, 10), function(x)
  prove[, .(Total.countries = .N,
            Bounded = sum(V1 >= x),
            Proportion = sum(V1 >= x) / .N)])
```

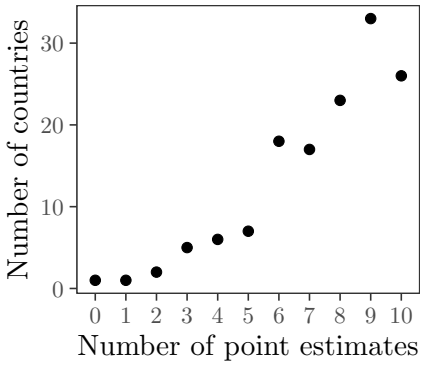
```
## [[1]]
##    Total.countries Bounded Proportion
## 1:             139      99  0.7122302
##
## [[2]]
##    Total.countries Bounded Proportion
## 1:             139      26  0.1870504
```

```
# Which countries do not show any framed estimate, or only 1, 2 or 3
lapply(c(0:3), function(x) prove[V1 == x])
```

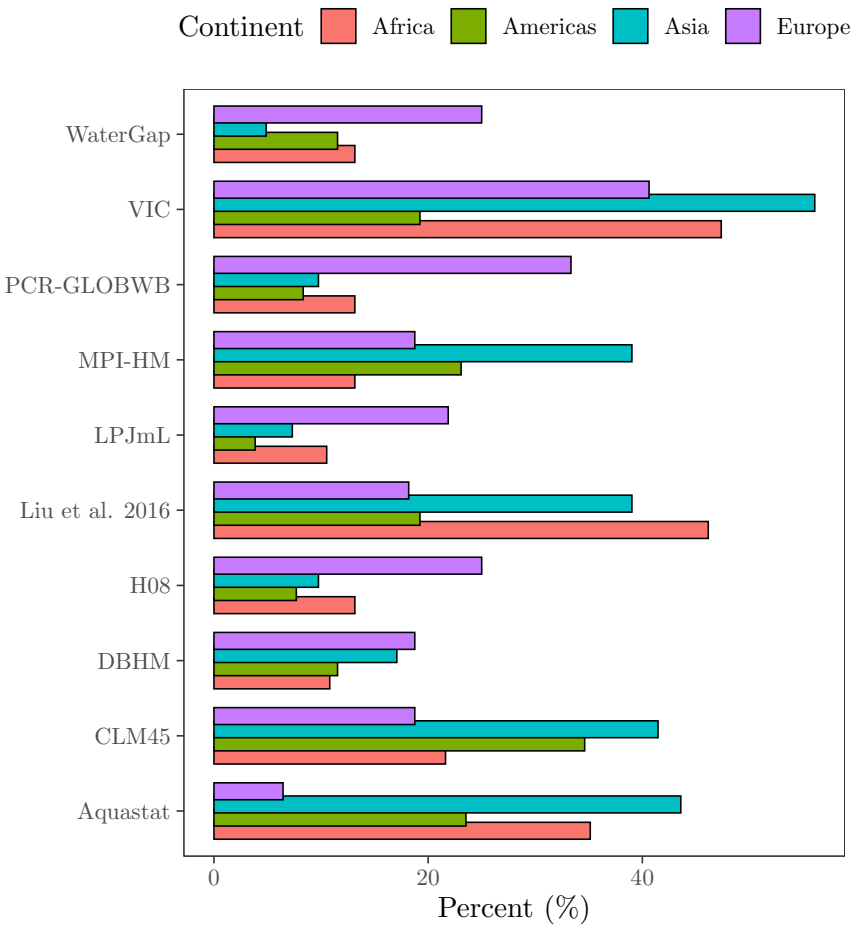
```
## [[1]]
##    Country Continent V1
## 1:   Malta    Europe  0
##
## [[2]]
##       Country Continent V1
## 1: Seychelles    Africa  1
##
## [[3]]
##    Country Continent V1
## 1:    Cuba  Americas  2
```

```
## 2:  Kuwait      Asia  2
##
## [[4]]
##        Country Continent V1
## 1:    Ethiopia    Africa  3
## 2: Puerto Rico  Americas  3
## 3:   Indonesia      Asia  3
## 4: Philippines      Asia  3
## 5:      Serbia    Europe  3
```

```
# PLOT ALL ESTIMATES TOGETHER ------------------------------------------------

data.table(table(prove$V1)) %>%
  .[, V1:= factor(V1, levels = 0:10)] %>%
  ggplot(., aes(V1, N)) +
  geom_point() +
  labs(x = "Number of point estimates",
       y = "Number of countries") +
  theme_AP()
```




```
# CHECK BIASED POINT ESTIMATES BY DATASET -------------------------------------

da[, .(N = sum(Water.Withdrawn < min | Water.Withdrawn > max),
       prop = sum((Water.Withdrawn < min | Water.Withdrawn > max) / .N) * 100),
   .(Water.Dataset, Continent)] %>%
  ggplot(., aes(Water.Dataset, prop, fill = Continent)) +
  geom_bar(stat = "identity",
           position = position_dodge(0.7),
           color = "black") +
  labs(x = "",
       y = "Percent (\\%)") +
  coord_flip() +
  theme_AP() +
  theme(legend.position = "top")
```



6 Lookup table

Our model (see below) will run rowwise, for $i = 1, 2, ..., N$ rows, and it will select an r_i^2 obtained given the conditions defined by three triggers: trigger X_{1_i} will select an GHM or FAO-based product from where to retrieve r^2 values; X_{2_i} the multiple imputation method used to fill missing values, and X_{3_i} the specific iteration from the pool of 40 different iterations computed for each missing value.

In order to speed up these computations, we create a lookup table with all possible r^2 values given all possible combinations between Continent, X_1, X_2 and X_3 (`lookup`). We sort `lookup` by all these elements through the use of the `setkey` function, which increases the speed of binary search.

We finally export all datasets to `.csv` files.

```
# CREATE LOOKUP TABLE ------------------------------------------------------

lookup <- setkey(all.results, index)
```

```
# EXPORT DATASETS ----------------------------------------------------------------

fwrite(full.imput, "full.imput.csv")
fwrite(results, "results.csv")
```

```r
fwrite(lookup, "lookup.csv")
fwrite(water.quantiles, "water.quantiles.csv")
```

7 Run the model

This section codes everything needed to run the simulations and conduct the uncertainty and sensitivity analysis.

7.1 Define settings

Firstly, we define a vector with the name of the continents, a vector with the name of the triggers (X_1, X_2, X_3), the sample size of our base sample matrix (2^{13}) and the order up to which we will compute sensitivity indices (third-order in this case).

```r
# DEFINE THE SETTINGS OF THE SAMPLE MATRIX -------------------------------------

Continents <- c("Africa", "Americas", "Asia", "Europe")

# Create a vector with the name of the columns
parameters <- paste("X", 1:4, sep = "")

# Select sample size
n <- 2 ^ 13

# Define order
order <- "third"
```

7.2 Sample matrix

Now we create the sample matrix, where each row represents a sample point and each column a trigger. We use the function `sobol_matrices` of the `sensobol` package (Puy et al., n.d.). In this case, since we will compute first, second, third and total-order Sobol' indices using the Jansen (1999) estimators, we create an $\mathbf{A}$, a $\mathbf{B}$ matrix, and k $\mathbf{A}_B^{(j)}$ matrices, where all columns come from $\mathbf{A}$ except the j-th, which comes from $\mathbf{B}$, for $k = 3$ (the number of triggers). Overall, this leads to a total computational cost of $C = N(k + 2) = 2^{13}(3 + 2) = 40960$ model runs per continent.

The matrices created with the `sobol_matrices` function use Sobol' Quasi-Random Number sequences (Sobol' 1967, 1976), which are uniformly distributed in $[0, 1]$. We transform these sequences to their appropriate probability distributions in the next code snippet.

```r
# CREATE THE SAMPLE MATRIX -----------------------------------------------------

# Create an A, B and AB matrices for each continent
sample.matrix <- lapply(Continents, function(Continents)
  sobol_matrices(N = n,
                 params = parameters,
                 order = order) %>%
    data.table())
```

```r
# Name the slots, each is a continent
names(sample.matrix) <- Continents

# Name the columns
sample.matrix <- lapply(sample.matrix, setnames, parameters)
```

The code below defines the function `transform_sample_matrix`, which transforms the columns of the sample matrix into their appropriate distributions ($[X_1 \sim \mathcal{DU}(1,8);\ X_2 \sim \mathcal{DU}(1,3);\ X_3 \sim \mathcal{DU}(1,40)]$). We link each discrete number in each distribution with a specific irrigation water withdrawal product, imputation method and iteration (Figure 7).

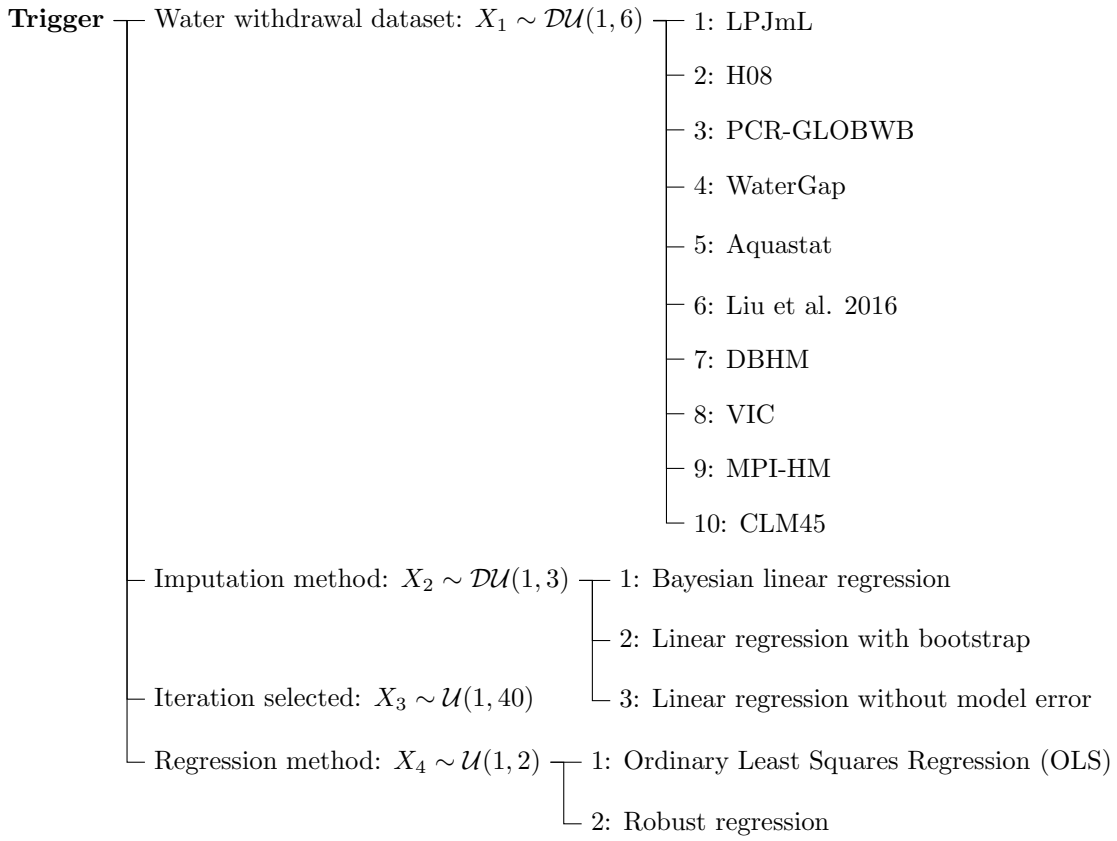


Figure 5: Tree diagram coding the discrete probability distributions of each trigger into each uncertainty level.

```r
# TRANSFORM THE SAMPLE MATRIX ----------------------------------------------------

# Function to transform sample matrix to appropriate distributions
transform_sample_matrix <- function(dt) {
  dt[, X1:= floor(X1 * (10 - 1 + 1)) + 1] %>%
    .[, X1:= ifelse(X1 == 1, "LPJmL",
                    ifelse(X1 == 2, "H08",
                           ifelse(X1 == 3, "PCR-GLOBWB",
                                  ifelse(X1 == 4, "WaterGap",
                                         ifelse(X1 == 5, "Aquastat",
                                                ifelse(X1 == 6, "Liu et al. 2016",
```

```
                                                    ifelse(X1 == 7, "DBHM",
                                                           ifelse(X1 == 8, "VIC",
                                                                  ifelse(X1 == 9, "MPI-HM",
    .[, X2:= floor(X2 * (length(imputation.methods) - 1 + 1)) + 1] %>%
    .[, X2:= ifelse(X2 == 1, imputation.methods[1],
                    ifelse(X2 == 2, imputation.methods[2], imputation.methods[3]))] %>%
    .[, X3:= floor(X3 * (m.iterations - 1 + 1)) + 1] %>%
    .[, X4:= floor(X4 * (2 - 1 + 1)) + 1] %>%
    .[, X4:= ifelse(X4 == 1, "Normal", "Robust")]
}

sample.matrix <- lapply(sample.matrix, transform_sample_matrix)
sample.matrix.dt <- rbindlist(sample.matrix, idcol = "Continent")
```

We then print out the sample matrix to allow the reader understand its organization.

```
# PRINT SAMPLE MATRIX ------------------------------------------------------

print(sample.matrix.dt)
```

```
##         Continent              X1        X2 X3     X4
##      1:    Africa Liu et al. 2016 norm.boot 21 Robust
##      2:    Africa             VIC      norm 31 Normal
##      3:    Africa      PCR-GLOBWB  norm.nob 11 Robust
##      4:    Africa        WaterGap norm.boot 26 Normal
##      5:    Africa          MPI-HM  norm.nob  6 Robust
##     ---
## 524284:    Europe      PCR-GLOBWB  norm.nob 21 Robust
## 524285:    Europe             VIC      norm  1 Normal
## 524286:    Europe Liu et al. 2016  norm.nob 11 Robust
## 524287:    Europe           LPJmL norm.boot 31 Normal
## 524288:    Europe           LPJmL  norm.nob 36 Robust
```

7.3 The model

The model is simply a one-line function that runs in the sample matrix as follows: for the i-th row, it creates a vector with all the conditions set by $X_{1_i}, X_{2_i}, X_{3_i}$, and uses this vector to extract from the lookup table the r_i^2 value according to these conditions.

```
# THE MODEL ----------------------------------------------------------------

model <- function(X) lookup[.(paste0(X[, 1:5], collapse = "_"))][, r.squared]
```

And now we run the model. In order to speed up the simulations, we rely on parallel computing. Once the computation is completed, we create the sample matrix to be used in the sensitivity analysis (`sample.matrix.dt`), the sample matrix to be used in the uncertainty analysis (`AB.dt`), and export the resulting data tables to `.csv.` files.

```
# RUN THE MODEL-------------------------------------------------------------
```

```r
# Set number of cores at 75%
n_cores <- floor(detectCores() * 0.75)

# Create cluster
cl <- makeCluster(n_cores)
registerDoParallel(cl)

# Run model in parallel
r.squared <- foreach(i=1:nrow(sample.matrix.dt),
              .packages = "data.table",
              .combine = "c") %dopar%
  {
    model(sample.matrix.dt[i])
  }

# Stop parallel cluster
stopCluster(cl)
```

```r
# ARRANGE MODEL OUTPUT ------------------------------------------------------

sample.matrix.dt <- cbind(sample.matrix.dt, r.squared)

# Select the A and B matrix only (for uncertainty analysis)
AB.dt <- sample.matrix.dt[, .SD[1:(n * 2)], Continent]

# Export results
fwrite(sample.matrix.dt, "sample.matrix.dt.csv")
fwrite(AB.dt, "AB.dt.csv")
```

8 Uncertainty analysis

In this section we present the results of the uncertainty analysis through a quantile table and different plots.

```r
# COMPUTE QUANTILES AND MEAN ------------------------------------------------

AB.dt[, .(q0.025 = quantile(r.squared, 0.025),
          q0.1 = quantile(r.squared, 0.1),
          q0.25 = quantile(r.squared, 0.25),
          q0.5 = quantile(r.squared, 0.5),
          q0.75 = quantile(r.squared, 0.75),
          q0.975 = quantile(r.squared, 0.975),
          q0.99 = quantile(r.squared, 0.99),
          q1 = quantile(r.squared, 1),
          mean = mean(r.squared),
          median = median(r.squared)), Continent]
```

```
##     Continent     q0.025       q0.1      q0.25       q0.5      q0.75     q0.975
```

```
## 1:    Africa 0.7548035 0.8011230 0.8507476 0.8669698 0.8851338 0.9109871
## 2:  Americas 0.6819216 0.7376662 0.8287603 0.8793270 0.9146256 0.9498162
## 3:      Asia 0.6774377 0.7287752 0.8642012 0.8869090 0.9022522 0.9197321
## 4:    Europe 0.5125560 0.5943295 0.6652576 0.7538706 0.8520642 0.9332680
##        q0.99        q1      mean    median
## 1: 0.9142279 0.9356294 0.8608576 0.8669698
## 2: 0.9578729 0.9810553 0.8581450 0.8793270
## 3: 0.9197321 0.9204432 0.8576442 0.8869090
## 4: 0.9402822 0.9548883 0.7484164 0.7538706
```

```
# PLOT UNCERTAINTY ------------------------------------------------------------

# Plot r2
unc.plot <- ggplot(AB.dt, aes(r.squared)) +
      geom_histogram(color = "black", fill = "white") +
      theme_AP() +
      labs(x = expression(italic(r) ^ 2),
           y = "Density") +
      scale_y_continuous(breaks = pretty_breaks(n = 2)) +
      scale_x_continuous(breaks = pretty_breaks(n = 3)) +
      facet_wrap(~Continent, ncol = 1) +
      theme(panel.spacing.x = unit(4, "mm"),
            strip.background = element_rect(fill = "white"))

unc.plot
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
# PLOT UNCERTAINTY IN EACH GHM AND FAO-BASED DATASET ---------------------------

unc.GHM <- AB.dt %>%
  ggplot(., aes(reorder(X1, r.squared), r.squared, fill = Continent)) +
  geom_boxplot(position = position_dodge(0.6),
               outlier.size = 0.3) +
  theme_AP() +
  labs(y = expression(italic(r)^2),
       x = "") +
  scale_x_discrete(labels = c("PCR-GLOBWB" = expression(bold(PCR-GLOBWB)),
                              "DBHM" = expression(bold(DBHM)),
                              "MPI-HM" = expression(bold(MPI-HM)),
                              "LPJmL" = expression(bold(LPJmL)),
                              "H08" = expression(bold(H08)),
                              "WaterGap" = expression(bold(WaterGap)),
                              "CLM45" = expression(bold(CLM45)),
                              "VIC" = expression(bold(VIC)))) +
  theme(legend.position = "none") +
  coord_flip()

# Get legend
```

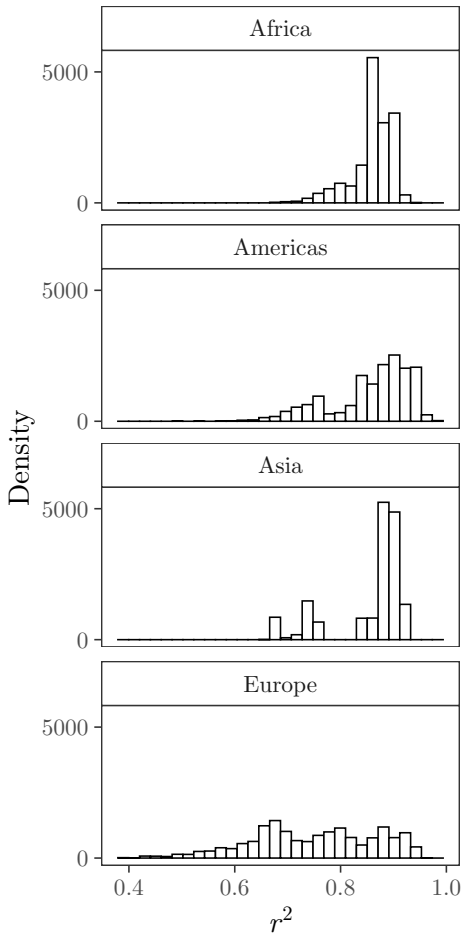


Figure 6: Uncertainty in the empirical distribution of r^2.

```
legend <- get_legend(unc.GHM + theme(legend.position = "top"))
```

```
# MERGE PLOTS ------------------------------------------------------------------

bottom <- plot_grid(unc.plot, unc.GHM, ncol = 2,
                    rel_widths = c(0.5, 1), labels = "auto")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
all <- plot_grid(legend, bottom, ncol = 1, rel_heights = c(0.1, 1))
```

```
# PLOT CUMULATIVE EMPIRICAL DISTRIBUTION FOR R2 ----------------------------------

ggplot(AB.dt, aes(r.squared, colour = Continent)) +
  stat_ecdf() +
  labs(x = "$r^2$",
       y = "y") +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP()
```

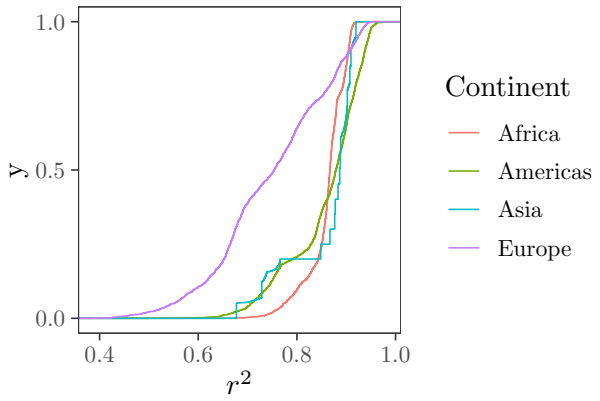


Figure 7: Cumulative empirical distribution for r^2.

9 Sensitivity analysis

And in this last section, we conduct the sensitivity analysis. We first plot the triggers against r^2 values, which allows to map back the model output into each of the triggers' levels. The resulting plot is Figure 10.

```
# PLOT SCATTERPLOTS OF PARAMETERS VS MODEL OUTPUT ---------------------------------

AB.dt <- AB.dt[, X3:= factor(X3, levels = as.factor(1:m.iterations))]

scatter.dt <- melt(AB.dt[, .SD[1:n], Continent],
                   measure.vars = paste("X", 1:4, sep = ""))

# R squared
ggplot(scatter.dt, aes(r.squared, value)) +
```

```
  geom_point(alpha = 0.05, size = 0.5) +
  facet_grid(variable ~ Continent,
             scales = "free_y",
             space = "free_y") +
  labs(y = "",
       x = expression(italic(r)^2)) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  scale_color_manual(values = c("#00BFC4", "#F8766D")) +
  theme_AP() +
  theme(axis.text.y = element_text(angle = 45, hjust = 1, size = 6),
        legend.position = "top")
```

We finally compute the Sobol' indices using the `sobol_indices` function of the sensobol package (Puy et al., n.d.), and bootstrap the results 1000 times. We also plot first, second, third and total-order indices in different plots, and export the results.

```
# SENSITIVITY ANALYSIS ----------------------------------------------------

# Number of bootstrap replicas
R <- 1000

parameters.recoded <- c("$X_1$", "$X_2$", "$X_3$", "$X_4$")

# Sobol' indices for r2
indices <- sample.matrix.dt[, sobol_indices(Y = r.squared,
                                            N = n,
                                            params = parameters.recoded,
                                            first = "jansen",
                                            R = R,
                                            boot = TRUE,
                                            parallel = "multicore",
                                            ncpus = n_cores,
                                            order = order),
                            Continent]
```

```
# PRINT AND EXPORT SENSITIVITY INDICES ------------------------------------

print(indices[sensitivity %in% c("Si", "Ti")])
```

```
##     Continent      original          bias    std.error        low.ci
##  1:    Africa  7.256732e-01 -7.716217e-05 0.0080177609  0.7100358087
##  2:    Africa  8.697334e-03 -2.617602e-05 0.0137915396 -0.0183074109
##  3:    Africa  1.508970e-02  3.935952e-04 0.0151321573 -0.0149623799
##  4:    Africa  3.222859e-03 -1.698043e-04 0.0117636519 -0.0196636704
##  5:    Africa  9.459857e-01 -4.345451e-04 0.0152978884  0.9164369076
##  6:    Africa  1.391249e-01 -3.077090e-04 0.0050858304  0.1294645163
##  7:    Africa  2.097153e-01 -3.689772e-04 0.0064797609  0.1973842283
##  8:    Africa  7.093798e-02  6.471522e-05 0.0028654696  0.0652570465
##  9:  Americas  6.432606e-01 -3.271822e-04 0.0102713447  0.6234562848
```

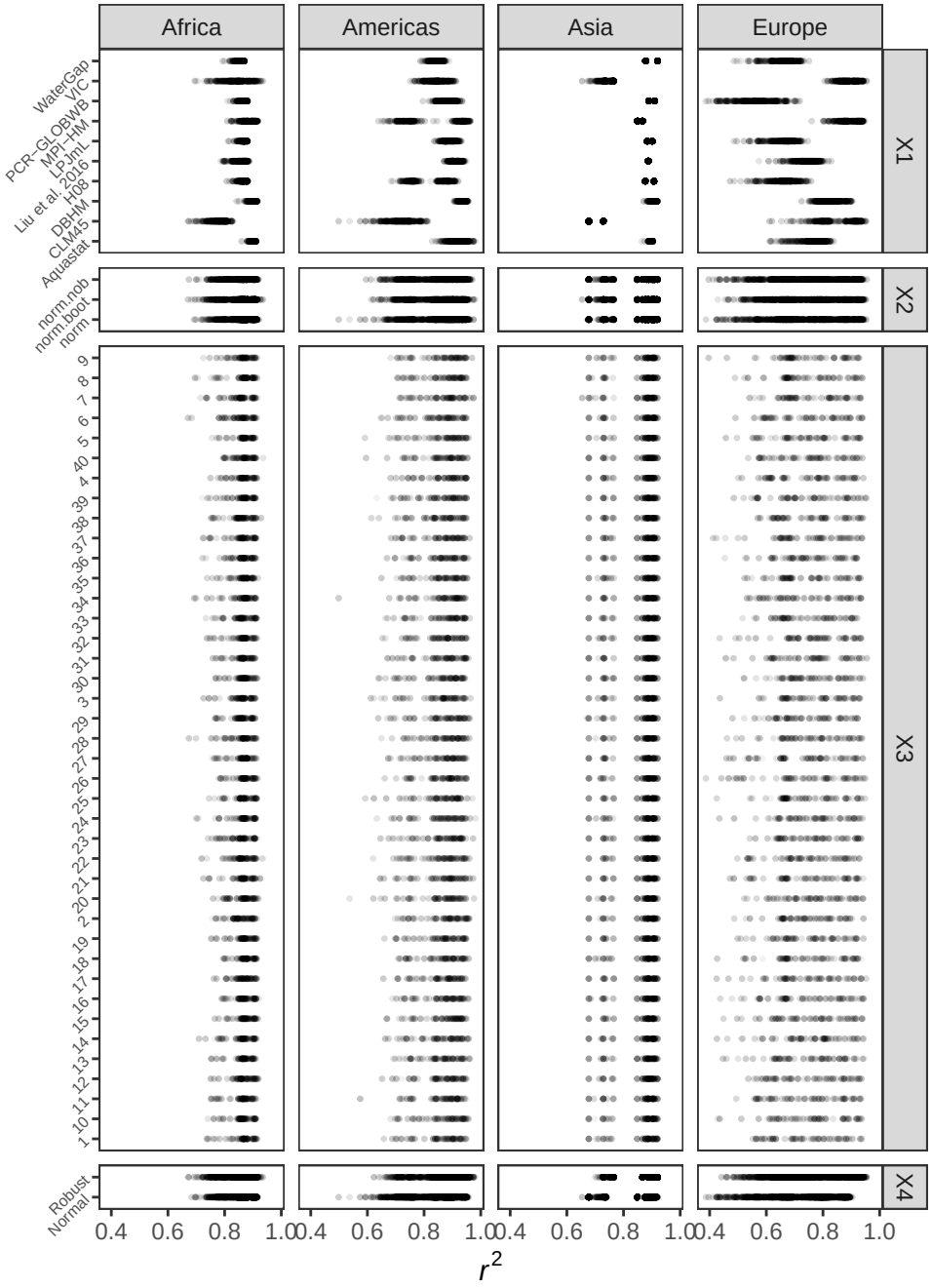


Figure 8: Scatterplots of r^2 against the triggers' levels.

```
## 10:  Americas -3.078453e-03  4.699477e-04 0.0126882751 -0.0284169631
## 11:  Americas  4.798126e-03  4.292432e-04 0.0130117101 -0.0211336005
## 12:  Americas  1.224791e-01  2.823474e-05 0.0119017656  0.0991237869
## 13:  Americas  8.684496e-01 -7.323785e-04 0.0133447674  0.8430267121
## 14:  Americas  6.047752e-02 -7.752331e-05 0.0019249869  0.0567821386
## 15:  Americas  8.752724e-02  6.494698e-05 0.0022975922  0.0829590986
## 16:  Americas  2.843562e-01 -4.501836e-04 0.0095212565  0.2661451028
## 17:      Asia  9.567842e-01 -4.057383e-05 0.0009181736  0.9550251405
## 18:      Asia  5.859341e-05 -1.737139e-05 0.0113488326 -0.0221673384
## 19:      Asia  2.966815e-03 -2.274778e-05 0.0113261886 -0.0192093595
## 20:      Asia  2.267684e-02  1.250063e-04 0.0117248634 -0.0004284776
## 21:      Asia  9.770700e-01 -6.263721e-05 0.0117393245  0.9541239546
## 22:      Asia  3.851782e-03 -1.038419e-05 0.0002856993  0.0033022056
## 23:      Asia  5.923995e-03  4.764772e-06 0.0003360372  0.0052606090
## 24:      Asia  3.762525e-02 -8.699411e-06 0.0007883860  0.0360887393
## 25:    Europe  8.364770e-01 -6.511651e-05 0.0032377090  0.8301963406
## 26:    Europe  3.854289e-03 -4.091515e-04 0.0122824709 -0.0198097600
## 27:    Europe -1.160610e-03 -4.322241e-04 0.0126989555 -0.0256178817
## 28:    Europe  3.292566e-02 -4.929460e-04 0.0113012064  0.0112686483
## 29:    Europe  9.495278e-01  3.879072e-04 0.0124880584  0.9246637327
## 30:    Europe  7.940836e-02  1.127868e-04 0.0021118556  0.0751564079
## 31:    Europe  1.120108e-01  5.109985e-05 0.0026009325  0.1068619583
## 32:    Europe  5.646907e-02  3.053508e-05 0.0015303781  0.0534390531
##     Continent      original          bias    std.error        low.ci
##         high.ci sensitivity parameters
##  1: 0.741464854          Si      $X_1$
##  2: 0.035754431          Si      $X_2$
##  3: 0.044354587          Si      $X_3$
##  4: 0.026448998          Si      $X_4$
##  5: 0.976403528          Ti      $X_1$
##  6: 0.149400605          Ti      $X_2$
##  7: 0.222784424          Ti      $X_3$
##  8: 0.076489481          Ti      $X_4$
##  9: 0.663719216          Si      $X_1$
## 10: 0.021320161          Si      $X_2$
## 11: 0.029871366          Si      $X_3$
## 12: 0.145777851          Si      $X_4$
## 13: 0.895337239          Ti      $X_1$
## 14: 0.064327949          Ti      $X_2$
## 15: 0.091965495          Ti      $X_3$
## 16: 0.303467743          Ti      $X_4$
## 17: 0.958624315          Si      $X_1$
## 18: 0.022319268          Si      $X_2$
## 19: 0.025188484          Si      $X_3$
## 20: 0.045532143          Si      $X_4$
## 21: 1.000141261          Ti      $X_1$
## 22: 0.004422126          Ti      $X_2$
## 23: 0.006577851          Ti      $X_3$
```

```
## 24: 0.039179155          Ti      $X_4$
## 25: 0.842887927          Si      $X_1$
## 26: 0.028336641          Si      $X_2$
## 27: 0.024161109          Si      $X_3$
## 28: 0.055568563          Si      $X_4$
## 29: 0.973616022          Ti      $X_1$
## 30: 0.083434730          Ti      $X_2$
## 31: 0.117057426          Ti      $X_3$
## 32: 0.059438025          Ti      $X_4$
##         high.ci sensitivity parameters
```

```
fwrite(indices, "indices.csv")
```

```
# PLOT UNCERTAINTY AND SOBOL' INDICES ----------------------------------------

bottom <- indices[sensitivity %in% c("Si", "Ti")] %>%
  ggplot(., aes(parameters, original, fill = sensitivity)) +
    geom_bar(stat = "identity",
             position = position_dodge(0.6),
             color = "black") +
    geom_errorbar(aes(ymin = low.ci,
                      ymax = high.ci),
                  position = position_dodge(0.6)) +
    scale_y_continuous(breaks = scales::pretty_breaks(n = 3)) +
    facet_wrap(~Continent,
               ncol = 4) +
    labs(x = "",
         y = "Sobol' index") +
    scale_fill_discrete(name = "Sobol' indices",
                        labels = c(expression(S[italic(i)]),
                                   expression(T[italic(i)]))) +
    theme_AP() +
    theme(legend.position = "top",
          strip.background = element_rect(fill = "white"))

bottom
```

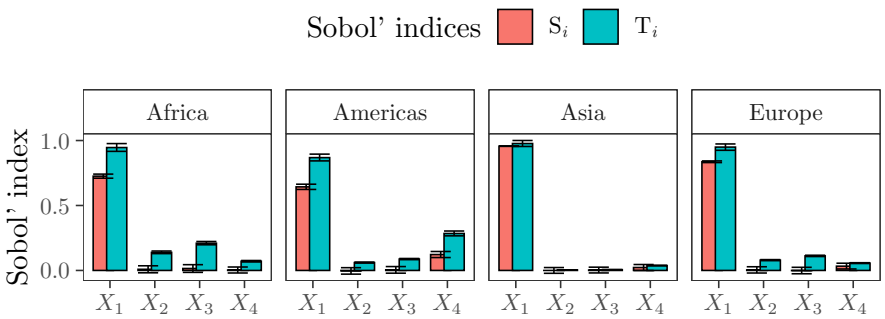


Figure 9: Sobol' indices. S_i and T_i refer respectively to Sobol' first and total order indices. S_i measures the influence of a parameter in the model output, while T_i measures the influence of a parameter jointly with its interactions.

```
# MERGE UNCERTAINTY AND SENSITIVITY ANALYSIS PLOTS ------------------------------

plot_grid(all, bottom, align = "hv", rel_heights = c(0.8, 0.4),
          labels = c("", "c"), ncol = 1)
```

```
## Warning: Graphs cannot be vertically aligned unless the axis parameter is set.
## Placing graphs unaligned.
```

```
## Warning: Graphs cannot be horizontally aligned unless the axis parameter is set.
## Placing graphs unaligned.
```

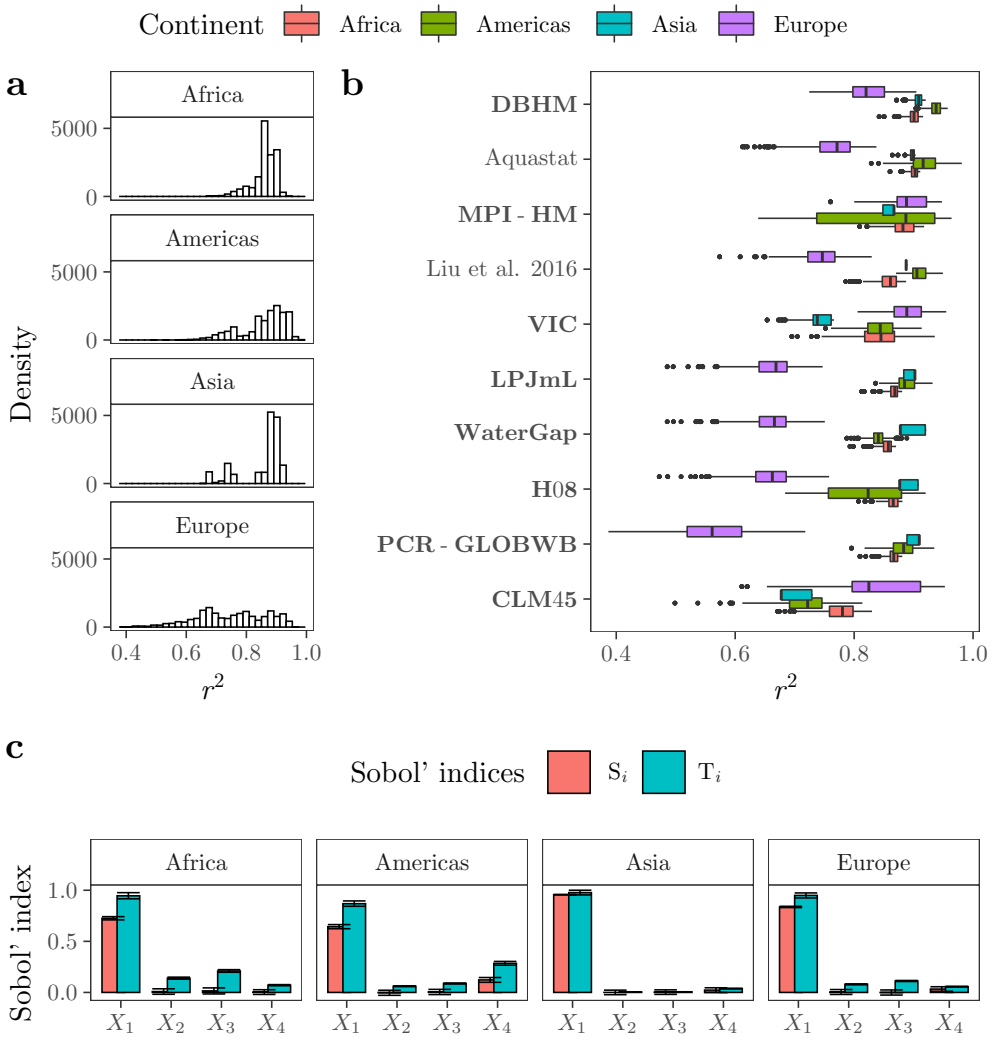


Figure 10: Uncertainty and sensitivity analysis. a) Empirical distribution for r^2 at the continental level. b) Boxplots of r^2 values obtained when regressions where run with GHM (in bold) and FAO-based datasets. c) Sobol' indices. S_i and T_i refer respectively to Sobol' first and total order indices. S_i measures the influence of a parameter in the model output, while T_i measures the influence of a parameter jointly with its interactions.

```
# CHECK SUM OF FIRST-ORDER INDICES ----------------------------------------------

indices[sensitivity == "Si",  sum(original), Continent]
```

```
##    Continent        V1
## 1:    Africa 0.7526831
## 2:  Americas 0.7674593
## 3:      Asia 0.9824864
## 4:    Europe 0.8720964
```

```
# PLOT SOBOL' INDICES (SECOND AND THIRD ORDER) ------------------------------------

indices[sensitivity == "Sij" | sensitivity == "Sijk"] %>%
  .[low.ci > 0] %>%
  .[, sensitivity:= ifelse(sensitivity %in% "Sij", "$S_{ij}$",
                           "$S_{ijk}$")] %>%
  ggplot(., aes(reorder(parameters, original), original, color = Continent)) +
  geom_point(position = position_dodge(0.6)) +
  geom_errorbar(aes(ymax = high.ci, ymin = low.ci),
                position = position_dodge(0.6)) +
  geom_hline(yintercept = 0,
             lty = 2,
             color = "red") +
  facet_grid(~sensitivity,
             scales = "free_x",
             space = "free_x") +
  scale_color_discrete(name = "Continent") +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  labs(x = "",
       y = "Sobol' index") +
  theme_AP()
```

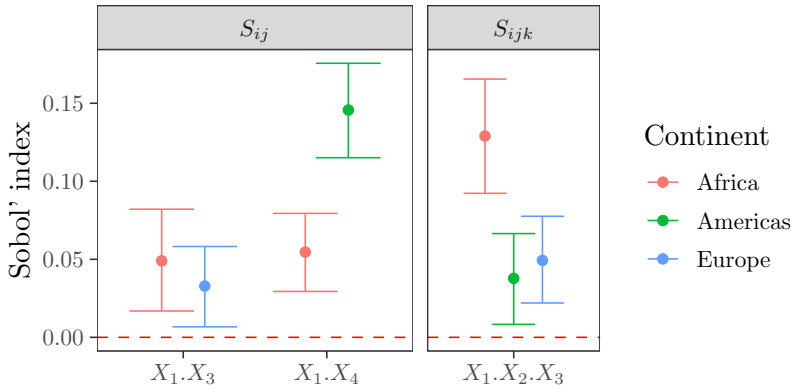
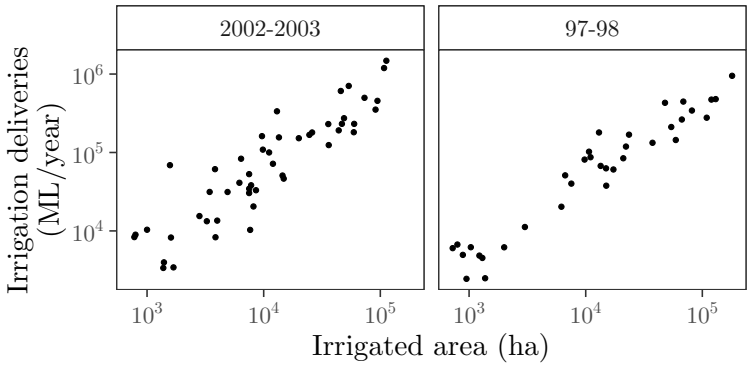


Figure 11: High-order interactions between the triggers. The dots and the errorbars show the mean and the 95% confidence intervals after bootstraping (R=1000).

10 Australian irrigated schemes

```
# RETRIEVE DATA FROM AUSTRALIAN IRRIGATION SCHEMES ------------------------------
australia.scheme <- fread("australia_scheme.csv")

# Filter out NA and rows with 0
australia.dt <- australia.scheme[, lapply(.SD, function(x)
  ifelse(x == 0, NA, x))] %>%
  na.omit() %>%
  .[, Year:= factor(Year, levels = c("97.98", "2002.2003"))] %>%
  .[, Year:= gsub("\\.", "-", Year)]
```

```
# PLOT SCATTERPLOT ---------------------------------------------------------------
a <- ggplot(australia.dt, aes(Irrigated.Area, Water.Withdrawal)) +
  geom_point(size = 0.6) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  facet_grid(~Year) +
  labs(x = "Irrigated area (ha)",
       y = "Irrigation deliveries \n (ML/year)") +
  theme_AP() +
  theme(strip.background = element_rect(fill = "white"))
a
```




```
# COMPUTE REGRESSIONS AND BOOTSTRAP ----------------------------------------------
col_transf <- c("Irrigated.Area", "Water.Withdrawal")
australia.dt[, (col_transf):= lapply(.SD, log10), .SDcols = col_transf]

australia.regressions <- australia.dt %>%
  group_by(Year) %>%
  nest() %>%
  mutate(fit = map(.x = data, .f = ~lm(Water.Withdrawal~ Irrigated.Area,
                                       data = .)),
         results = map(fit, glance),
         residuals = map(fit, augment))

# Retrieve results
australia.results <- australia.regressions  %>%
```

```
  dplyr::select(Year, results) %>%
  unnest(results) %>%
  data.table()

# Retrieve residuals
australia.residuals <- australia.regressions  %>%
  dplyr::select(Year, residuals) %>%
  unnest(residuals) %>%
  data.table()

# Bootstrap
foo <- boot(australia.dt, function(data,indices)
  summary(lm(Water.Withdrawal ~ Irrigated.Area,
             data[indices, ] ))$r.squared, R = 5000)

ci_plot <- boot.ci(foo, type = "all") # Confidence intervals
```

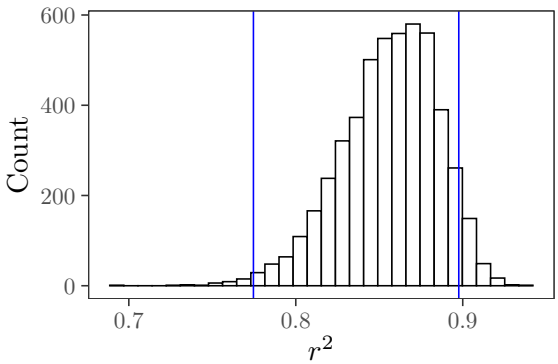
```
## Warning in boot.ci(foo, type = "all"): bootstrap variances needed for
## studentized intervals
```

```
ci_plot
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = foo, type = "all")
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.7997,  0.9132 )   ( 0.8073,  0.9209 )
##
## Level     Percentile            BCa
## 95%   ( 0.7914,  0.9050 )   ( 0.7746,  0.8976 )
## Calculations and Intervals on Original Scale
```

```
b <- data.table(foo$t) %>%
  ggplot(., aes(V1)) +
  geom_histogram(color = "black", fill = "white") +
  geom_vline(xintercept = c(ci_plot$bca[[4]], ci_plot$bca[[5]]), color = "blue") +
  labs(x = "$r^2$",
       y = "Count") +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP()

b
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

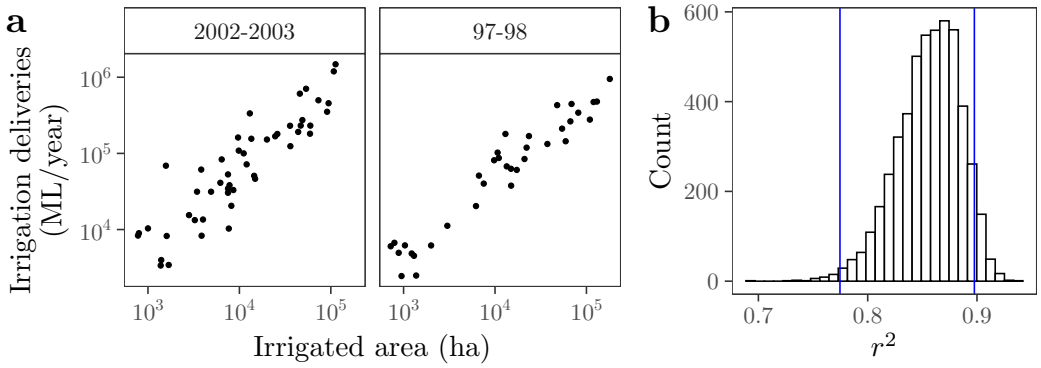



```
# PLOT ALL AUSTRALIAN ------------------------------------------------------------
plot_grid(a, b, labels ="auto", ncol = 2, align = "hv", rel_widths = c(1, 0.65))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Graphs cannot be vertically aligned unless the axis parameter is set.
## Placing graphs unaligned.
```

```
## Warning: Graphs cannot be horizontally aligned unless the axis parameter is set.
## Placing graphs unaligned.
```



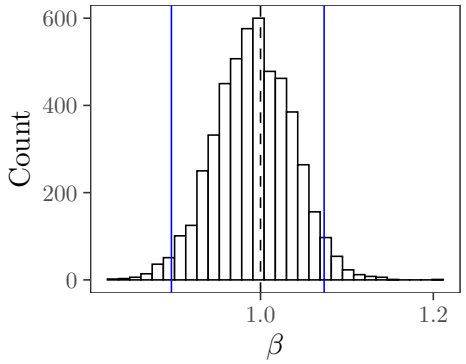

```
# BOOTSTRAP BETA AT THE SCHEME LEVEL ---------------------------------------------

foo.beta <- boot(australia.dt, function(data,indices)
  coef(lm(Water.Withdrawal ~ Irrigated.Area,
            data[indices, ]))[[2]], R = 5000)

# Confidence intervals
ci_plot <- boot.ci(foo.beta, type = "all")
```

```
## Warning in boot.ci(foo.beta, type = "all"): bootstrap variances needed for
## studentized intervals
```

```
# Plot
data.table(foo.beta$t) %>%
  ggplot(., aes(V1)) +
  geom_histogram(color = "black", fill = "white") +
  geom_vline(xintercept = c(ci_plot$bca[[4]], ci_plot$bca[[5]]), color = "blue") +
  geom_vline(xintercept = 1, lty = 2) +
```

```
  labs(x = "$\\beta$",
       y = "Count") +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP()
```

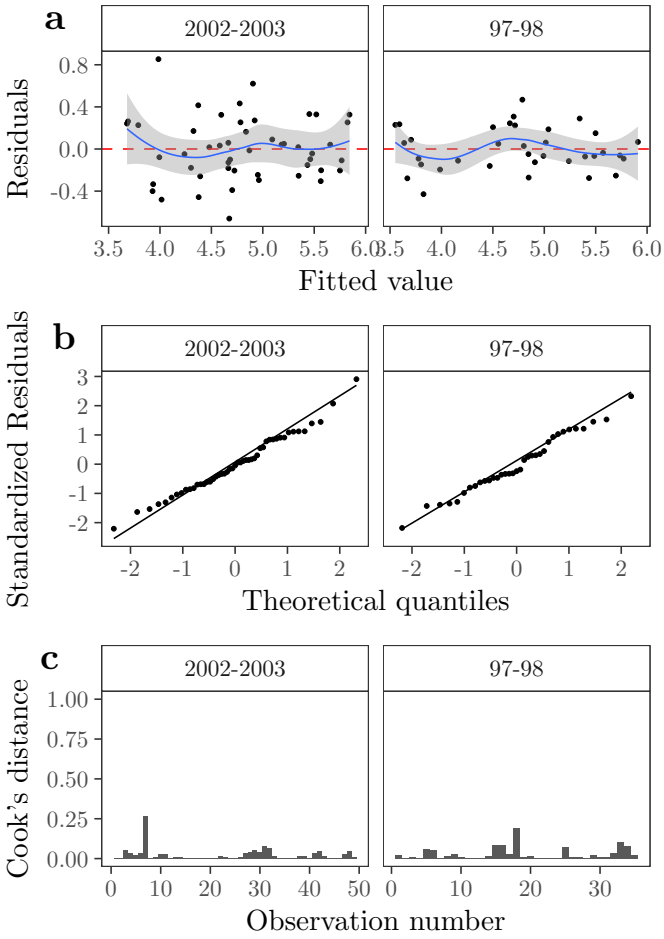
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# REGRESSION DIAGNOSTICS ----------------------------------------------------

# Residuals versus fitted
a <- ggplot(australia.residuals, aes(.fitted, .resid)) +
  geom_point(size = 0.5) +
  geom_hline(yintercept = 0, lty = 2, color = "red") +
  geom_smooth(size = 0.5) +
  labs(x = "Fitted value", y = "Residuals") +
  facet_grid(~Year, scales = "free_y") +
  theme_AP() +
  theme(strip.text.y = element_text(size = 6.4),
        strip.background = element_rect(fill = "white"))

# QQ plot
b <- ggplot(australia.residuals, aes(sample = .std.resid)) +
  stat_qq(size = 0.5) +
  stat_qq_line() +
  facet_grid(~Year, scales = "free_y") +
  labs(x = "Theoretical quantiles",
       y = "Standardized Residuals") +
  theme_AP() +
  theme(strip.text.y = element_text(size = 6.4),
        strip.background = element_rect(fill = "white"))

# Cook's distance
ID <- australia.residuals[, .N, Year]
australia.residuals <- australia.residuals[, ID:= unlist(sapply(ID[, N], function(x) 1:x))]
c <- ggplot(australia.residuals, aes(ID, .cooksd)) +
  geom_col() +
  scale_y_continuous(limits = c(0, 1)) +
```

```
  facet_grid(~Year, scales = "free_x") +
  theme_AP() +
  labs(x = "Observation number", y = "Cook's distance") +
  theme(strip.text.y = element_text(size = 6.4),
        strip.background = element_rect(fill = "white"))

plot_grid(a, b, c, labels = "auto", ncol = 1, align = "hv", hjust = -2.5)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



11 Irrigated area at the cell level

```
# PIXEL LEVEL ------------------------------------------------------------------

ncin <- nc_open("histsoc_landuse-totals_annual_1861_2005.nc")
lon <- ncvar_get(ncin, "lon")
lat <- ncvar_get(ncin, "lat")
tmp_array <- ncvar_get(ncin, "cropland_irrigated")
m <- (dim(tmp_array)[3] - 11):dim(tmp_array)[3]
tmp_slice <- lapply(m, function(m) tmp_array[, , m])
```

```
lonlat <- as.matrix(expand.grid(lon,lat))
tmp_vec <- lapply(tmp_slice, function(x) as.vector(x))
tmp_df01 <- lapply(tmp_vec, function(x) data.frame(cbind(lonlat, x)))
names(tmp_df01) <- m
da <- lapply(tmp_df01, data.table) %>%
  rbindlist(., idcol = "month") %>%
  na.omit() %>%
  .[!x == 0]
Country <- coords2country(da[1:nrow(da), 2:3])
df <- cbind(Country, da)
setDT(df)
df <- setnames(df, c("Var1", "Var2"), c("lon", "lat"))
mirca <- na.omit(df)[, .(area = mean(x)), .(Country, lon, lat)] %>%
  .[order(Country)]

# Function to open the Huang datasets
open_nc_mirca <- function(file, dname) {
  nc <- nc_open(file)
  ww <- ncvar_get(nc, dname)
  lon <- ncvar_get(nc, "lon")
  lat <- ncvar_get(nc, "lat")
  water <- rowSums(ww[, 469:ncol(ww)])
  ww.df <- data.frame(cbind(lon, lat, water))
  setDT(ww.df)
  ww.df <- ww.df[!water == 0]
  Country <- coords2country(ww.df[1:nrow(ww.df), 1:2])
  dt <- cbind(Country, ww.df) %>%
    na.omit() %>%
    .[, water:= water / 1000]
  return(dt)
}

# Function to open ISIMIP datasets
open_nc_mirca_isimip <- function(file, dname) {
  ncin <- nc_open(file)
  # get longitude, latitude, time
  lon <- ncvar_get(ncin, "lon")
  lat <- ncvar_get(ncin, "lat")
  # Get variable
  tmp_array <- ncvar_get(ncin, dname)
  # Retrieve last 12 months
  # get last year
  m <- (dim(tmp_array)[3] - 11):dim(tmp_array)[3]
  tmp_slice <- lapply(m, function(m) tmp_array[, , m])
  # create dataframe -- reshape data
  # matrix (nlon*nlat rows by 2 cols) of lons and lats
  lonlat <- as.matrix(expand.grid(lon,lat))
```

```
  # vector of `tmp` values
  tmp_vec <- lapply(tmp_slice, function(x) as.vector(x))
  # create dataframe and add names
  tmp_df01 <- lapply(tmp_vec, function(x) data.frame(cbind(lonlat, x)))
  names(tmp_df01) <- m
  da <- lapply(tmp_df01, data.table) %>%
    rbindlist(., idcol = "month") %>%
    na.omit()
  # Convert coordinates to country
  Country <- coords2country(da[1:nrow(da), 2:3])
  df <- cbind(Country, da)
  setDT(df)
  out <- na.omit(df)[, .(water = sum(x)), .(Country, Var1, Var2)] %>%
    .[, water:= water * 10000]
  out <- out[order(Country)] %>%
    .[!water == 0] %>%
    setnames(., c("Var1", "Var2"), c("lon", "lat"))
  return(out)
}

# HUANG ET AL DATASETS
names_nc_files <- c("withd_irr_lpjml.nc", "withd_irr_pcrglobwb.nc",
                    "withd_irr_h08.nc", "withd_irr_watergap.nc")
out.nc.mirca <- mclapply(names_nc_files, function(x)
  open_nc_mirca(x, "withd_irr"), mc.cores = detectCores() * 0.75)
names(out.nc.mirca) <- c("LPJmL", "PCR-GLOBWB", "H08", "WaterGap")
out.final.mirca <- rbindlist(out.nc.mirca, idcol = "Water.Dataset")

# ISIMIP DATASETS
files <- list("dbh_wfdei_nobc_hist_varsoc_co2_airrww_global_monthly_1971_2010.nc",
              "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrww_global_monthly_1861_2005.nc",
              "vic_wfdei_nobc_hist_pressoc_co2_airrww_global_monthly_1971_2010.nc")
dname <- "airrww"

isimip.dt.mirca <- mclapply(files, function(x)
  open_nc_mirca_isimip(file = x, dname = dname), mc.cores = detectCores() * 0.75)

names(isimip.dt.mirca) <- c("DBHM", "MPI-HM", "VIC")

# ADD CLM45
CLM45.mirca <- open_nc_mirca_isimip(file = "clm45_gfdl-esm2m_ewembi_historical_2005soc_co2_pir
                                    dname = "pirrww")

CLM45.mirca <- CLM45.mirca[, Water.Dataset:= "CLM45"] %>%
  setcolorder(., c("Water.Dataset", "Country", "lon", "lat", "water"))

final.isimip.mirca <- rbindlist(isimip.dt.mirca, idcol = "Water.Dataset") %>%
```

```
  rbind(CLM45.mirca) %>%
  rbind(out.final.mirca) %>%
  na.omit()

full.isimip <- merge(final.isimip.mirca, mirca, by = c("Country", "lon", "lat"), all.y = TRUE)

full.isimip[, `:=` (Code = countrycode(full.isimip[, Country],
                                       origin = "country.name",
                                       destination = "un"),
                    Continent = countrycode(full.isimip[, Country],
                                            origin = "country.name",
                                            destination = "continent"))]
```

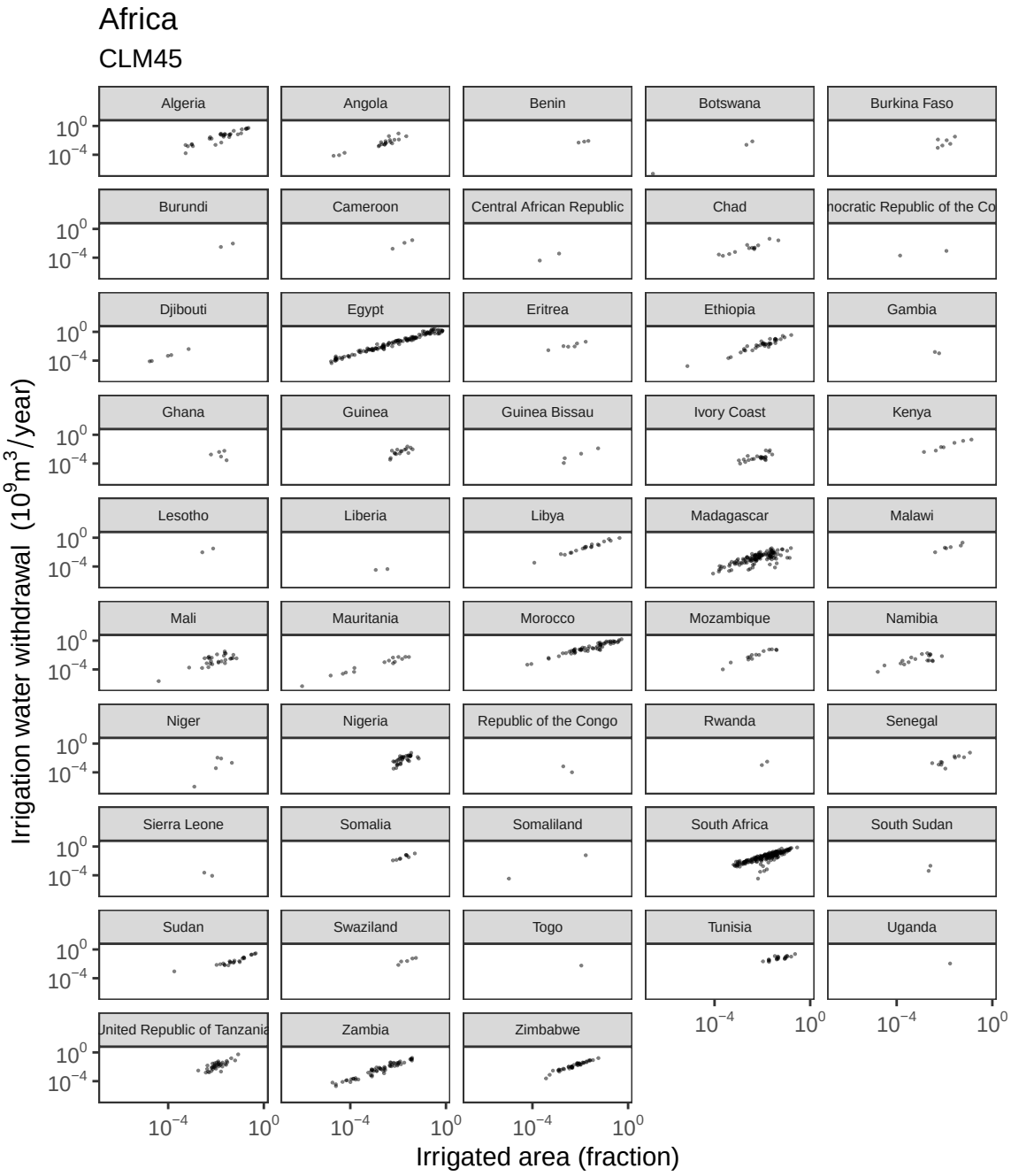
```
## Warning in countrycode(full.isimip[, Country], origin = "country.name", : Some values were n
```

```
## Warning in countrycode(full.isimip[, Country], origin = "country.name", : Some values were n
```

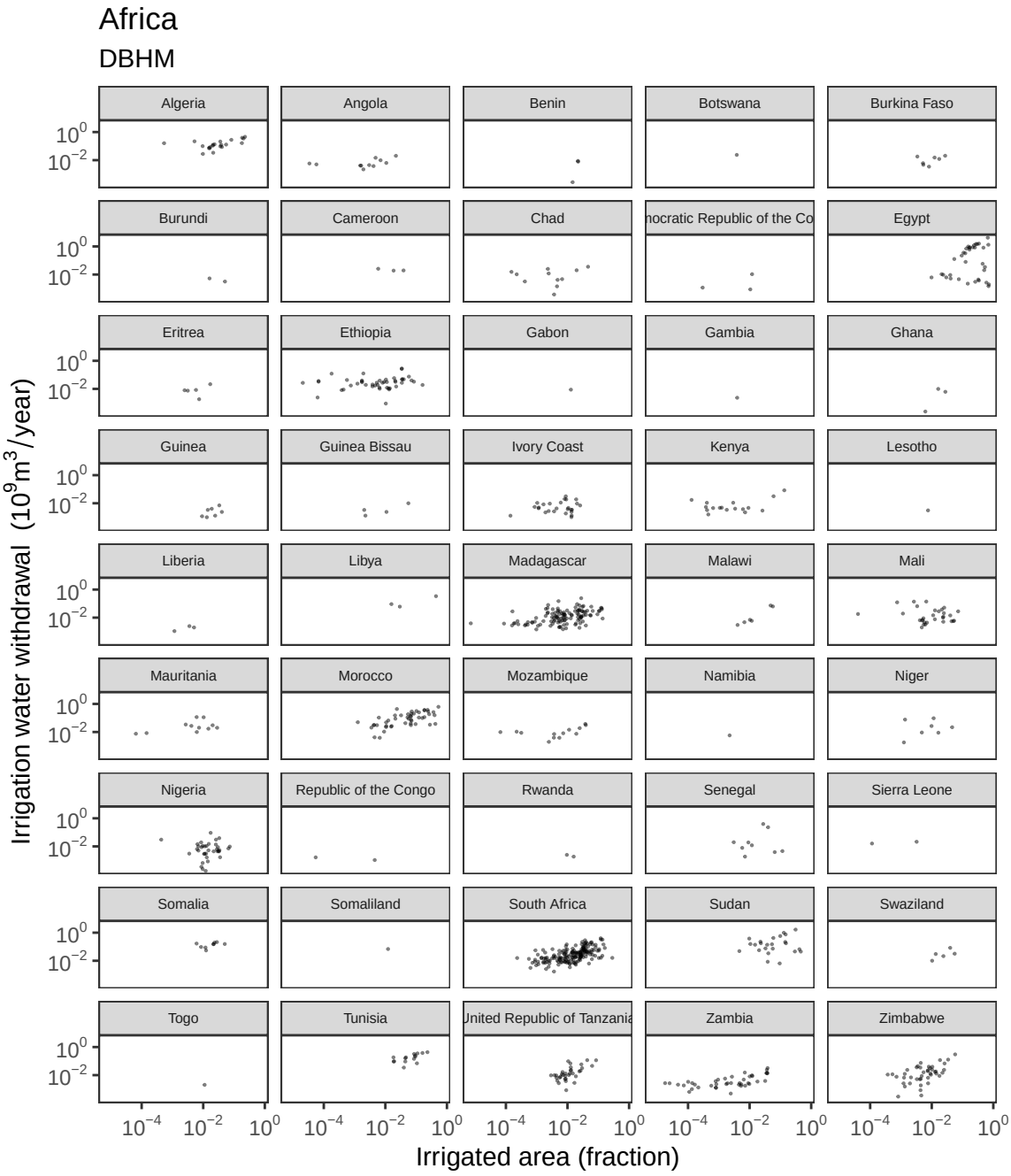
```
full.isimip <- full.isimip[!Continent == "Oceania"]
```

```
# PLOT ------------------------------------------------------------------------

full.isimip.split <- split(full.isimip, full.isimip$Continent) %>%
  lapply(., function(x) split(x, x$Water.Dataset))

gg <- list()
for(i in names(full.isimip.split)) {
  for(j in names(full.isimip.split[[i]])) {
    gg[[i]][[j]] <- ggplot(full.isimip.split[[i]][[j]], aes(area, water)) +
      geom_point(size = 0.1, alpha = 0.5) +
      scale_x_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                    labels = trans_format("log10", math_format(10 ^ .x))) +
      scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                    labels = trans_format("log10", math_format(10 ^ .x))) +
      facet_wrap(~Country,
                 ncol = 5) +
      labs(x = "Irrigated area (fraction)",
           y = expression(paste("Irrigation water withdrawal ", " ", "(", 10^9, m^3/year, "", "
      theme_AP() +
      theme(strip.text.x = element_text(size = 6)) +
      ggtitle(label = names(full.isimip.split[i]),
              subtitle = names(full.isimip.split[[i]][j]))
  }
}

gg
```

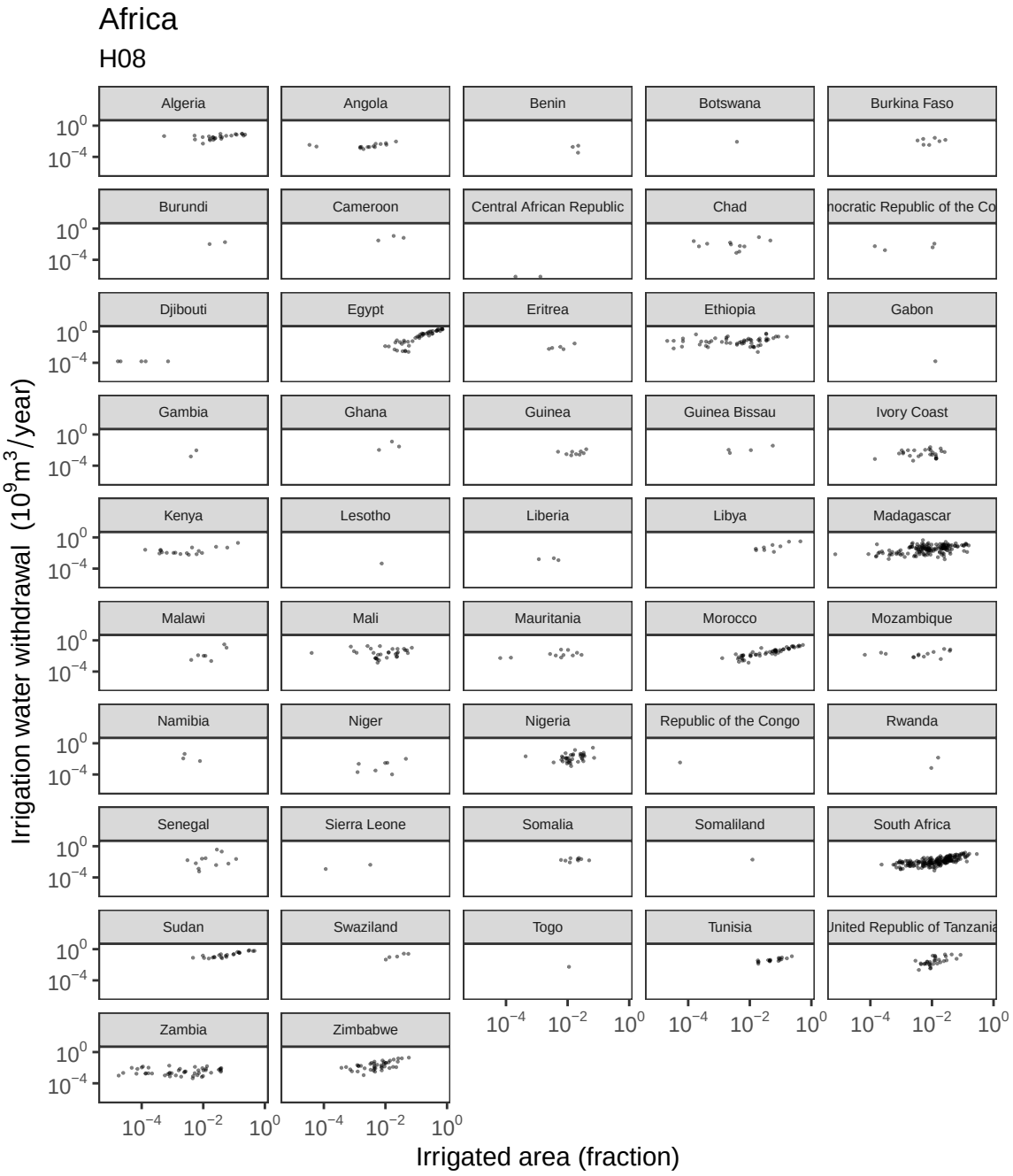
```
## $Africa
## $Africa$CLM45
```


Africa
CLM45
Algeria
Angola
Benin
Botswana
Burkina Faso
Burundi
Cameroon
Central African Republic
Chad
nocratic Republic of the Co
Djibouti
Egypt
Eritrea
Ethiopia
Gambia
Ghana
Guinea
Guinea Bissau
Ivory Coast
Kenya
Lesotho
Liberia
Libya
Madagascar
Malawi
Mali
Mauritania
Morocco
Mozambique
Namibia
Niger
Nigeria
Republic of the Congo
Rwanda
Senegal
Sierra Leone
Somalia
Somaliland
South Africa
South Sudan
Sudan
Swaziland
Togo
Tunisia
Uganda
United Republic of Tanzania
Zambia
Zimbabwe
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

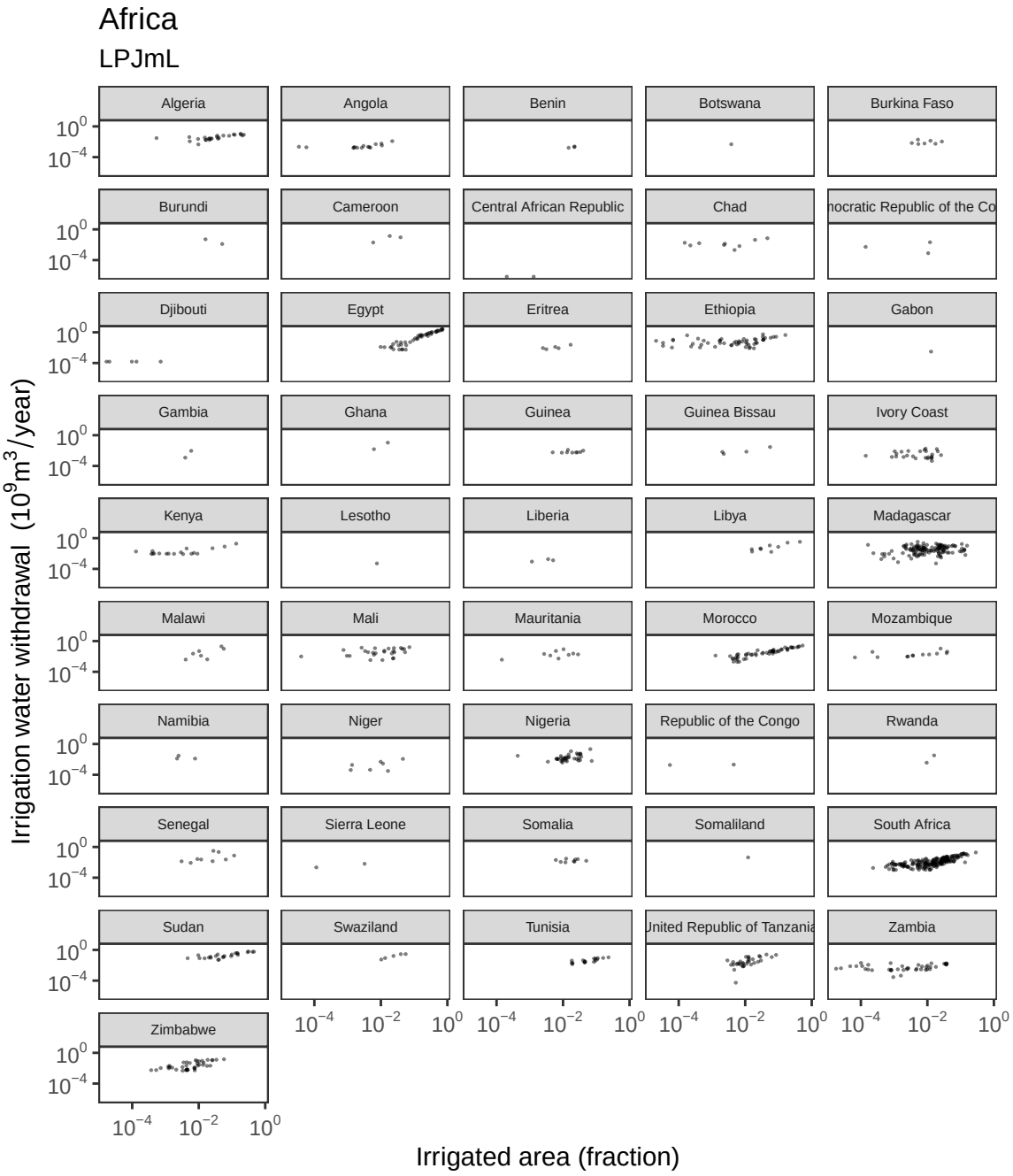
```
##
## $Africa$DBHM
```


Africa
DBHM
Algeria
Angola
Benin
Botswana
Burkina Faso
Burundi
Cameroon
Chad
hocratic Republic of the Co
Egypt
Eritrea
Ethiopia
Gabon
Gambia
Ghana
Guinea
Guinea Bissau
Ivory Coast
Kenya
Lesotho
Liberia
Libya
Madagascar
Malawi
Mali
Mauritania
Morocco
Mozambique
Namibia
Niger
Nigeria
Republic of the Congo
Rwanda
Senegal
Sierra Leone
Somalia
Somaliland
South Africa
Sudan
Swaziland
Togo
Tunisia
United Republic of Tanzania
Zambia
Zimbabwe
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

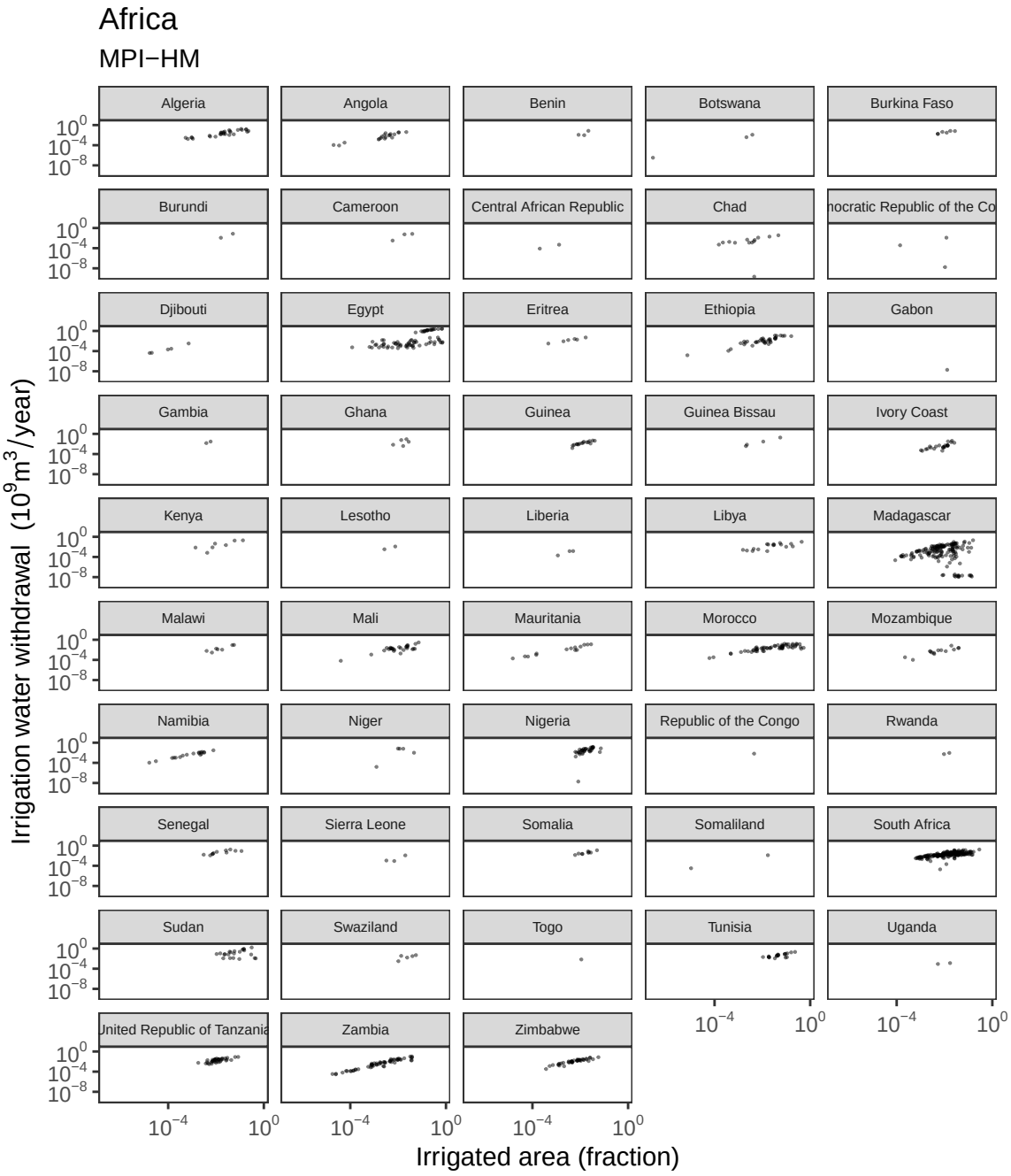
```
##
## $Africa$H08
```



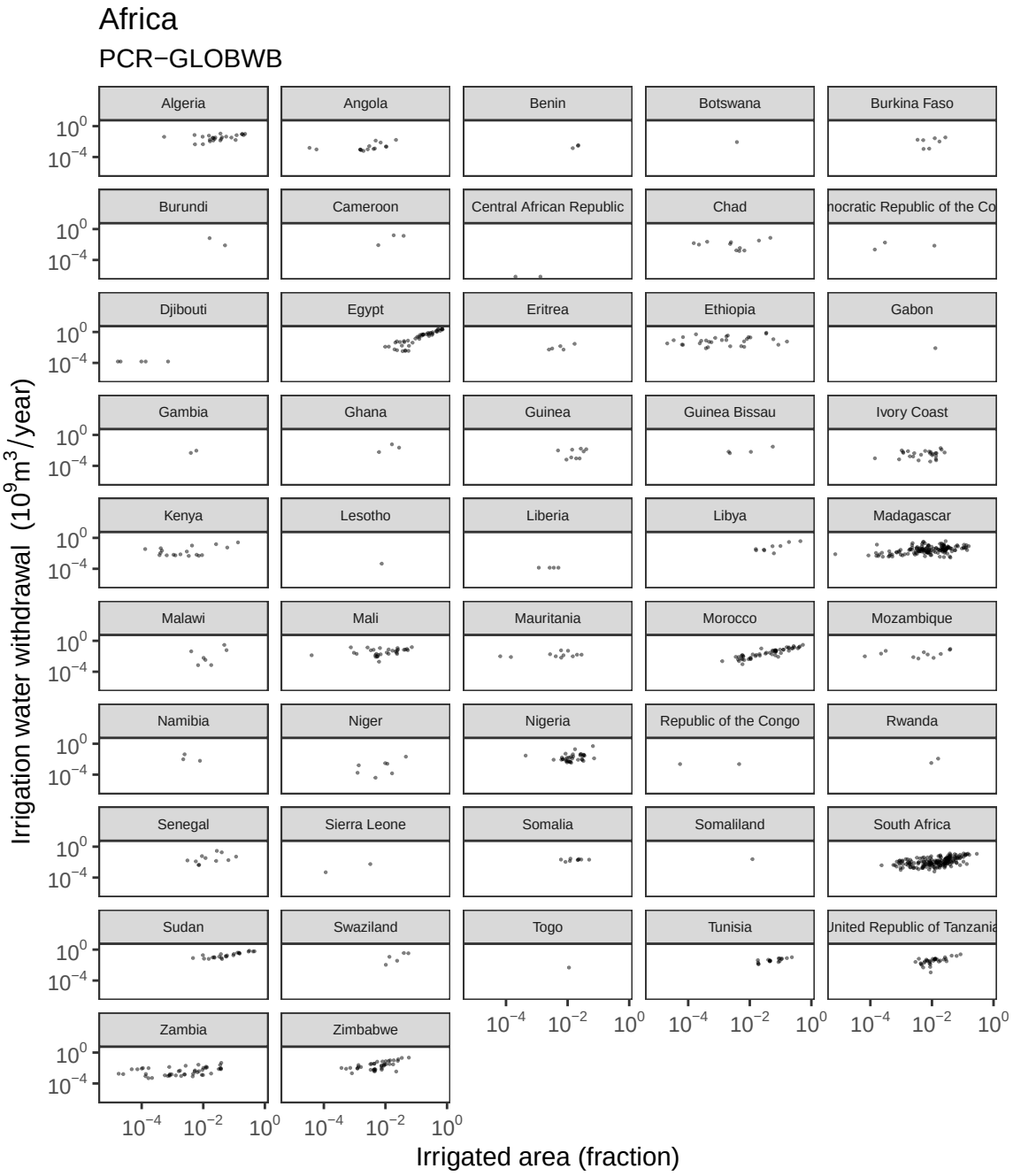

```
##
## $Africa$LPJmL
```



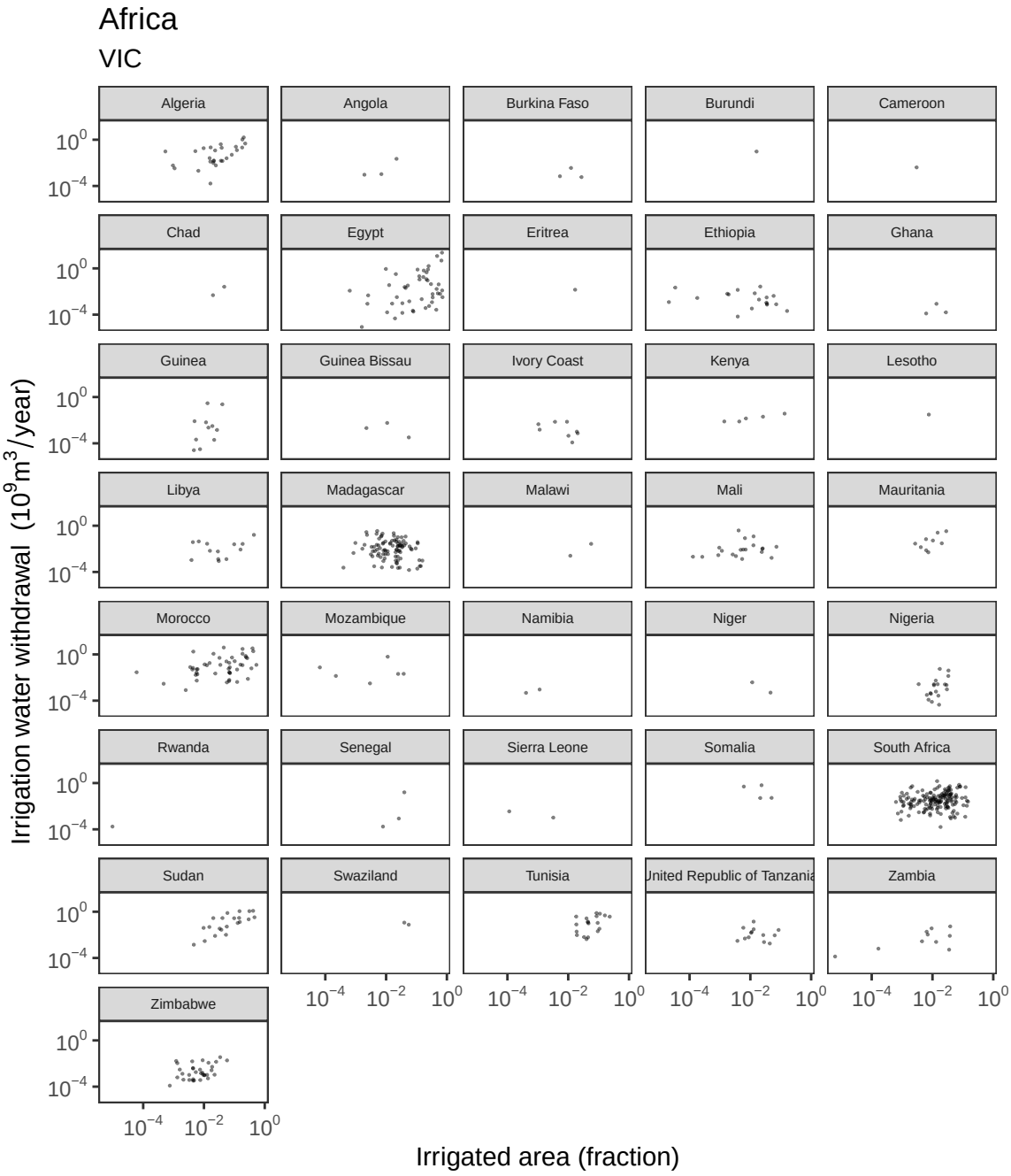

```
##
## $Africa$`MPI-HM`
```



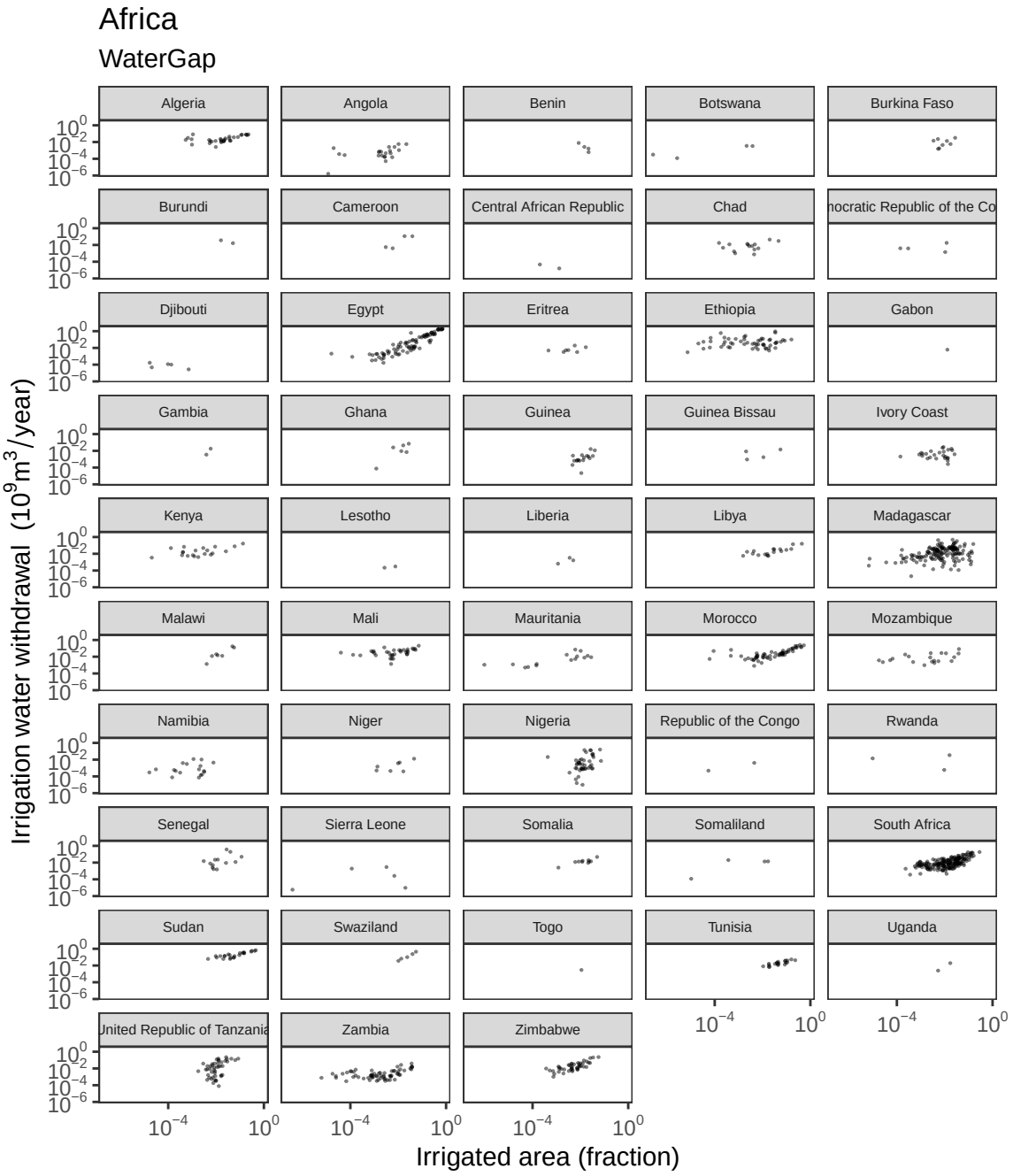

```
##
## $Africa$`PCR-GLOBWB`
```


Africa
PCR−GLOBWB
Algeria
Angola
Benin
Botswana
Burkina Faso
Burundi
Cameroon
Central African Republic
Chad
nocratic Republic of the Co
Djibouti
Egypt
Eritrea
Ethiopia
Gabon
Gambia
Ghana
Guinea
Guinea Bissau
Ivory Coast
Kenya
Lesotho
Liberia
Libya
Madagascar
Malawi
Mali
Mauritania
Morocco
Mozambique
Namibia
Niger
Nigeria
Republic of the Congo
Rwanda
Senegal
Sierra Leone
Somalia
Somaliland
South Africa
Sudan
Swaziland
Togo
Tunisia
United Republic of Tanzania
Zambia
Zimbabwe
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

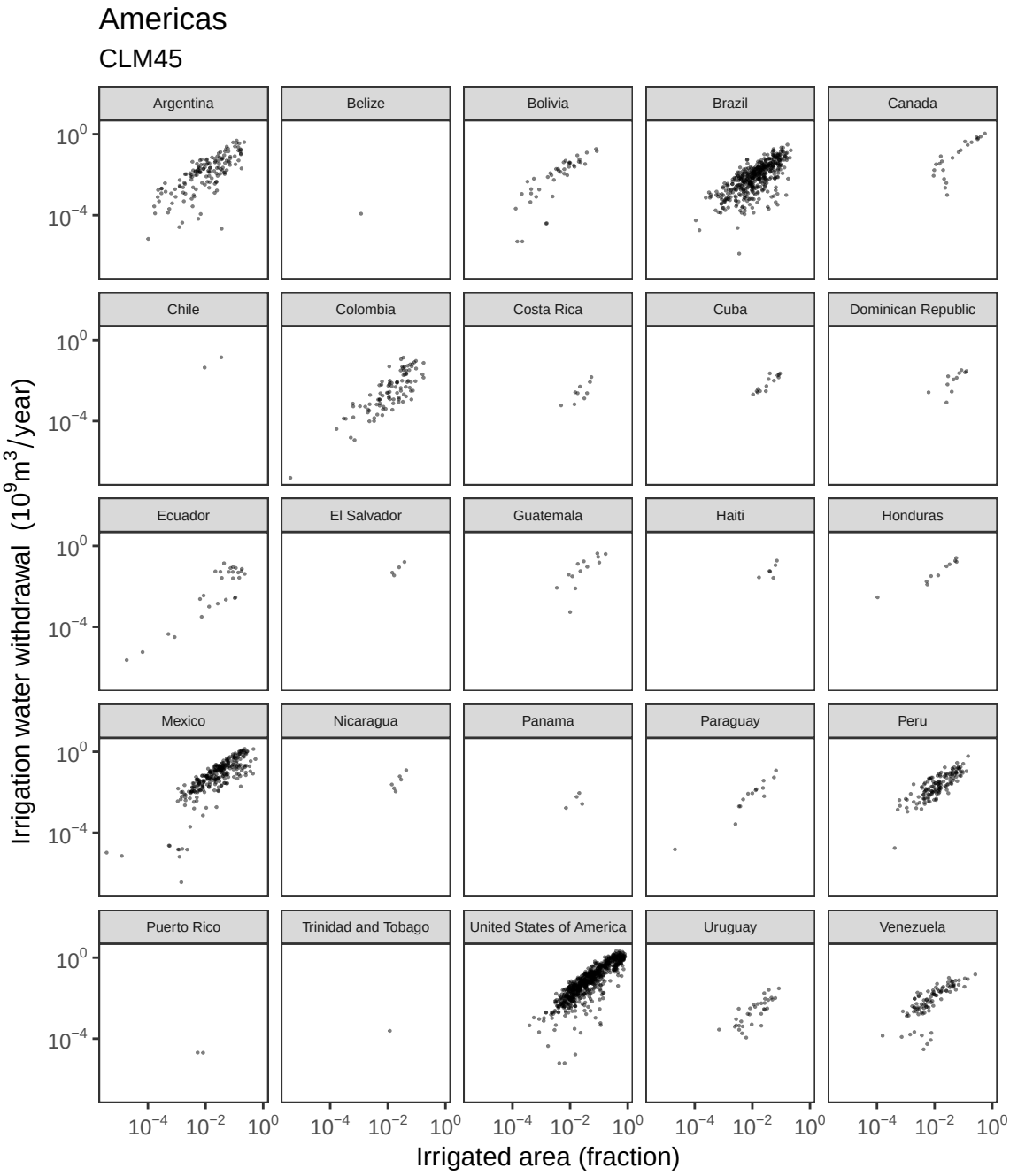
```
##
## $Africa$VIC
```



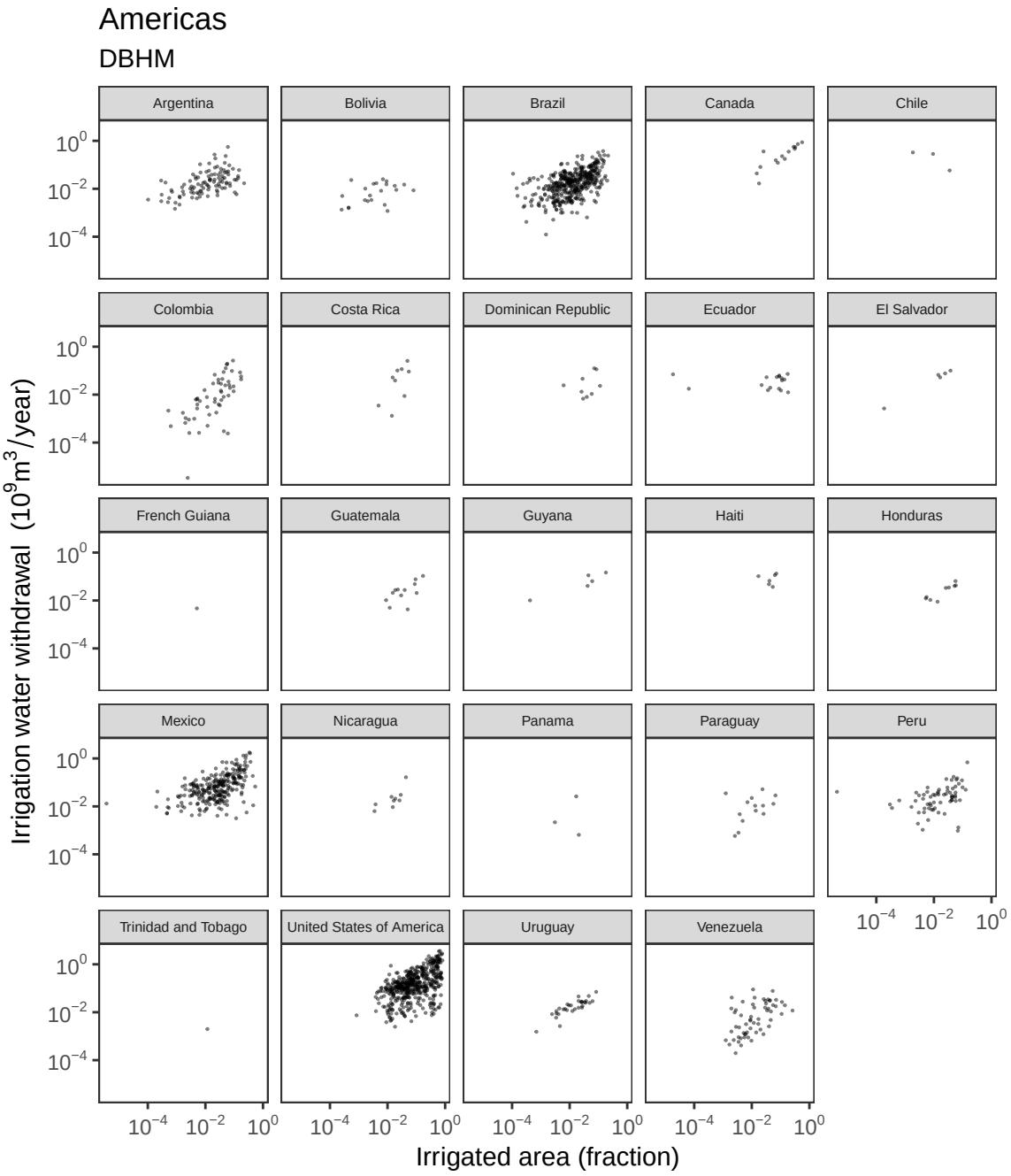

```
##
## $Africa$WaterGap
```




```
##
##
## $Americas
## $Americas$CLM45
```

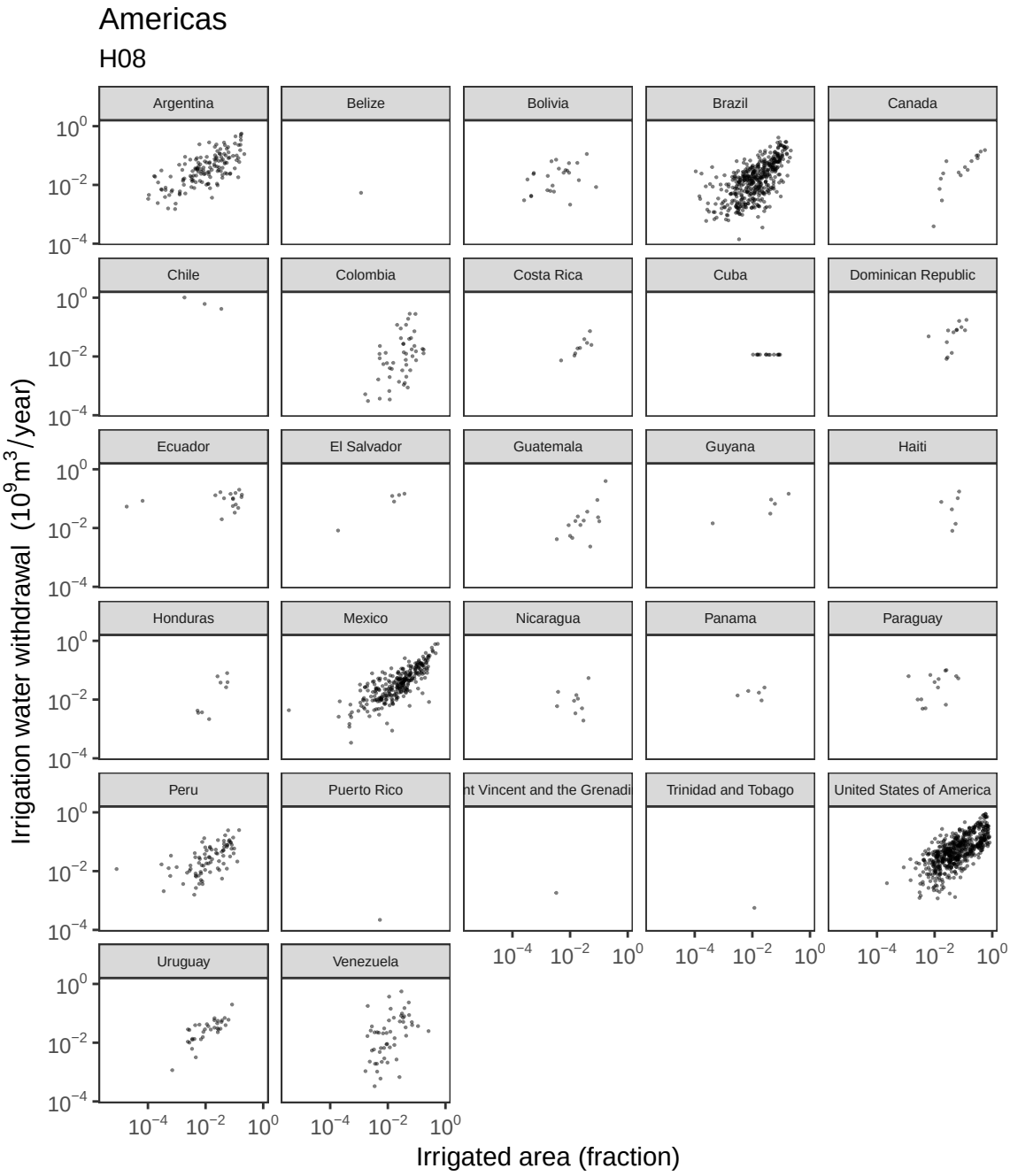



```
##
## $Americas$DBHM
```

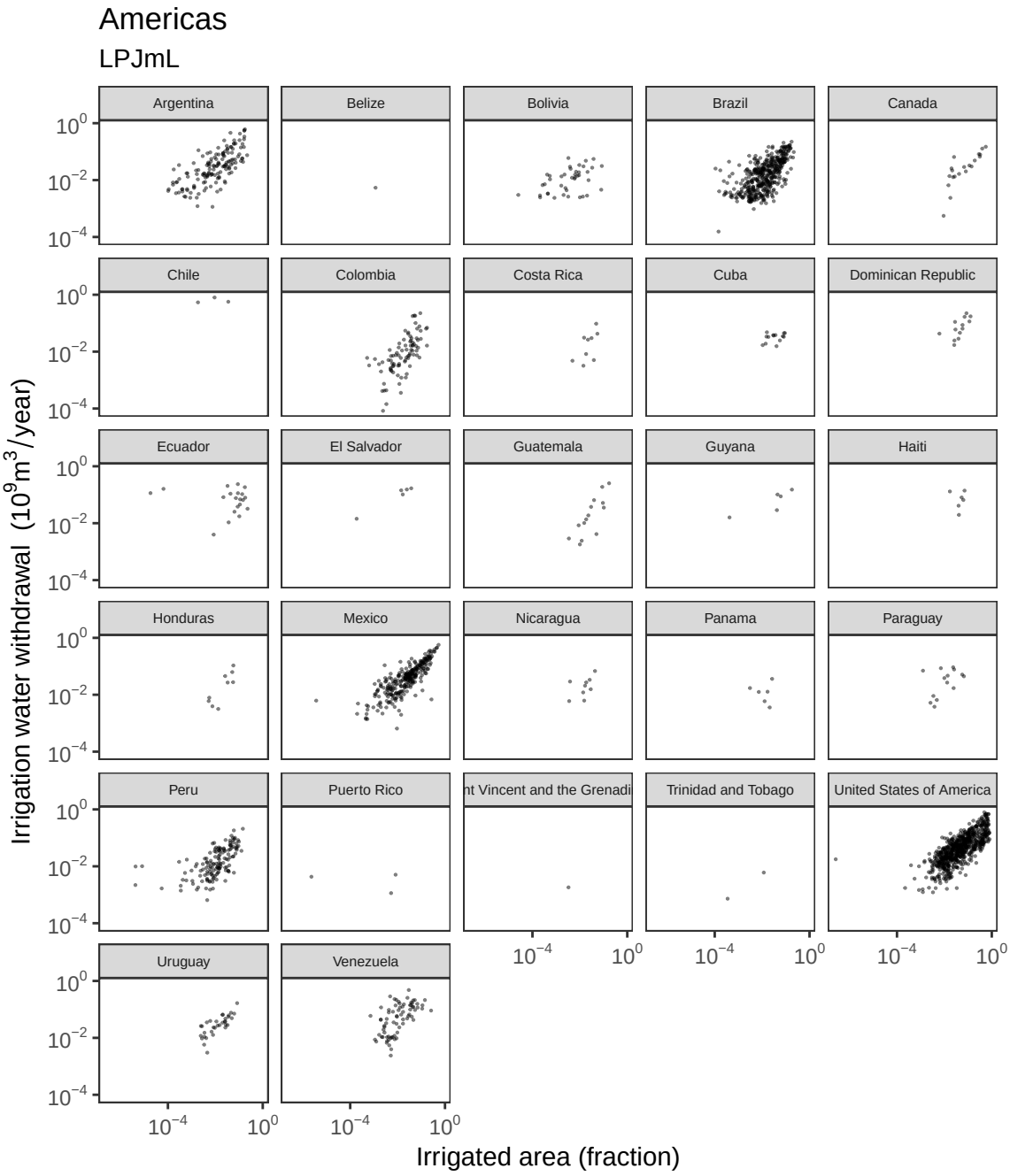


##

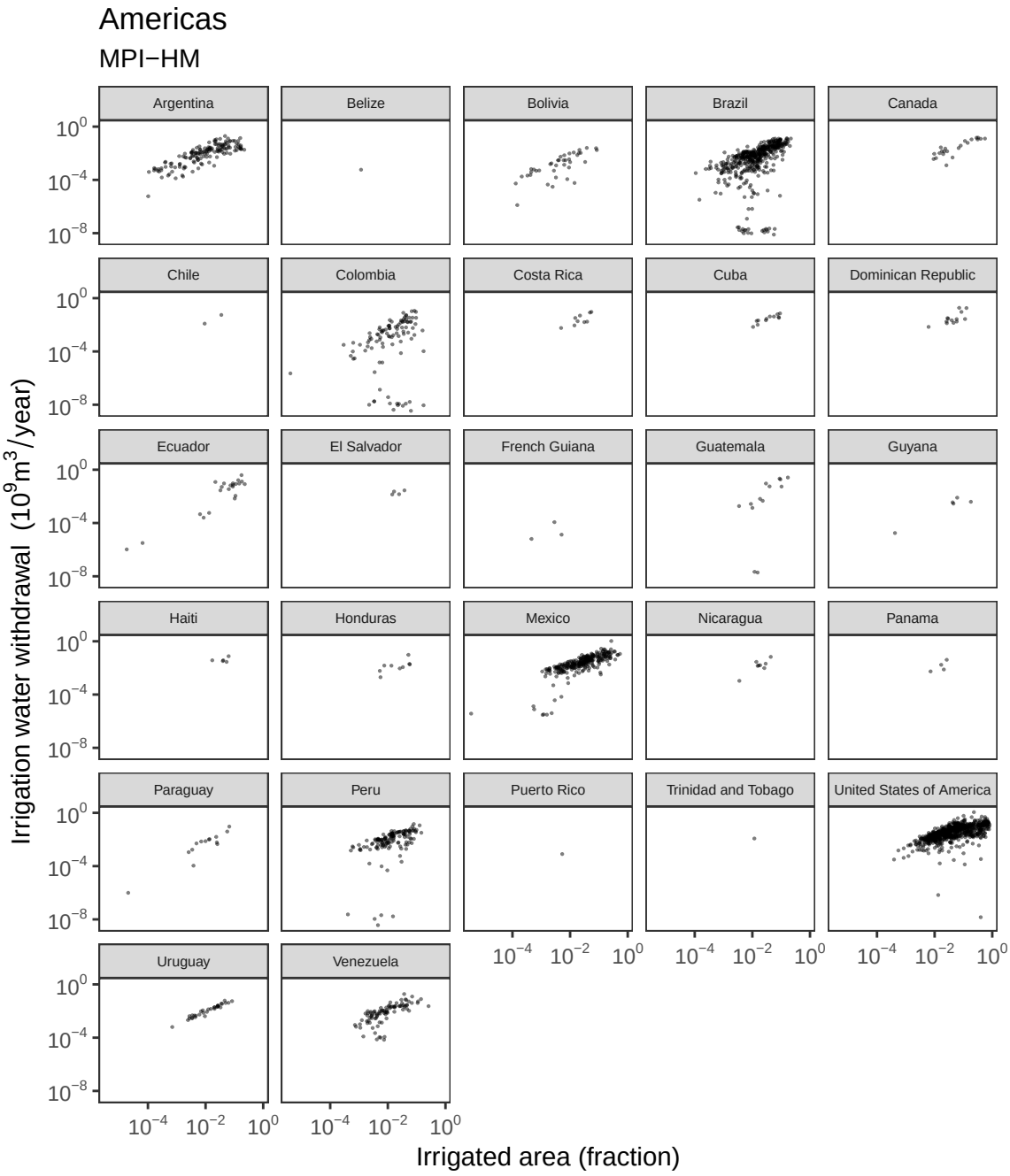
$Americas$H08



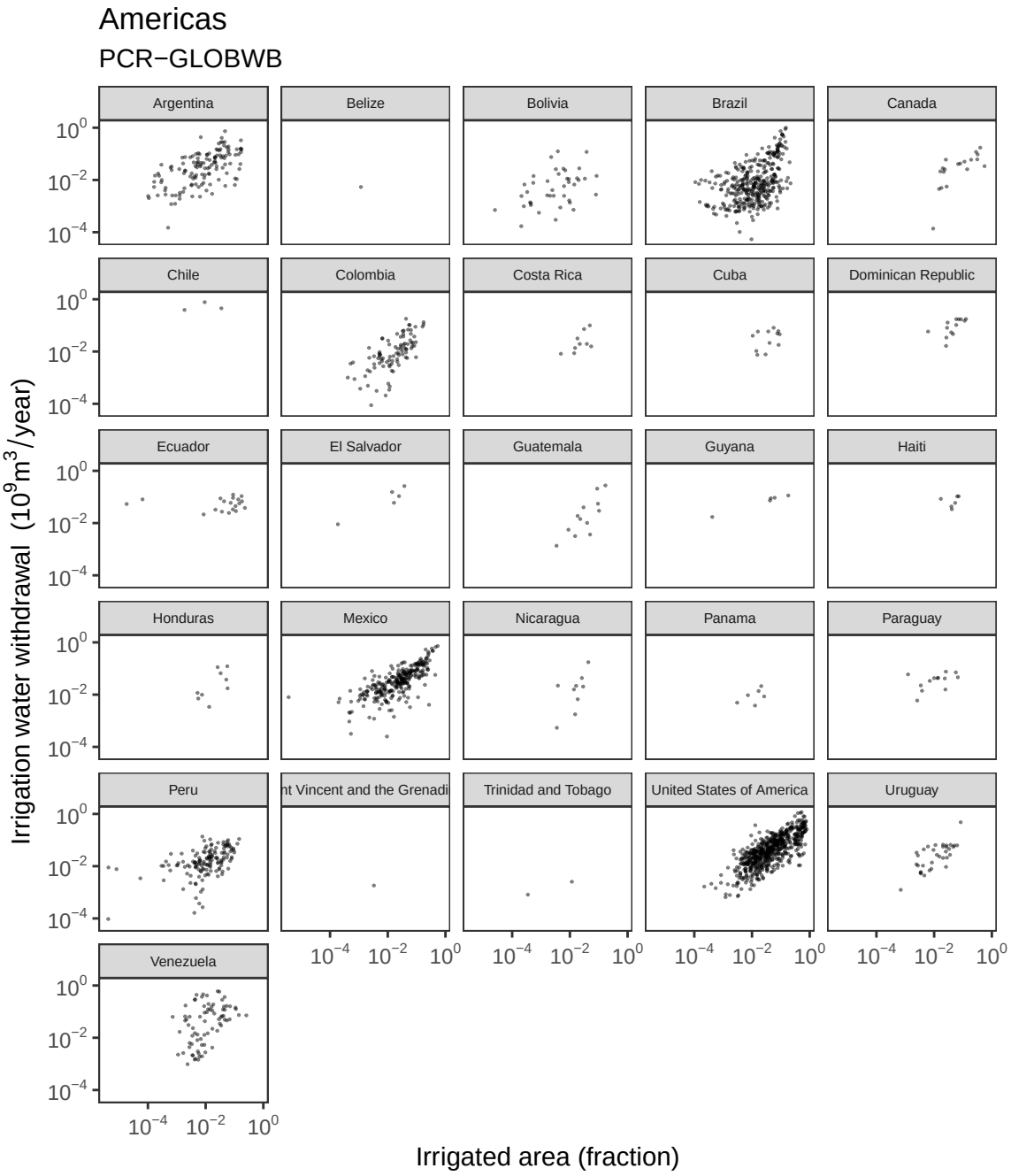

```
##
## $Americas$LPJmL
```



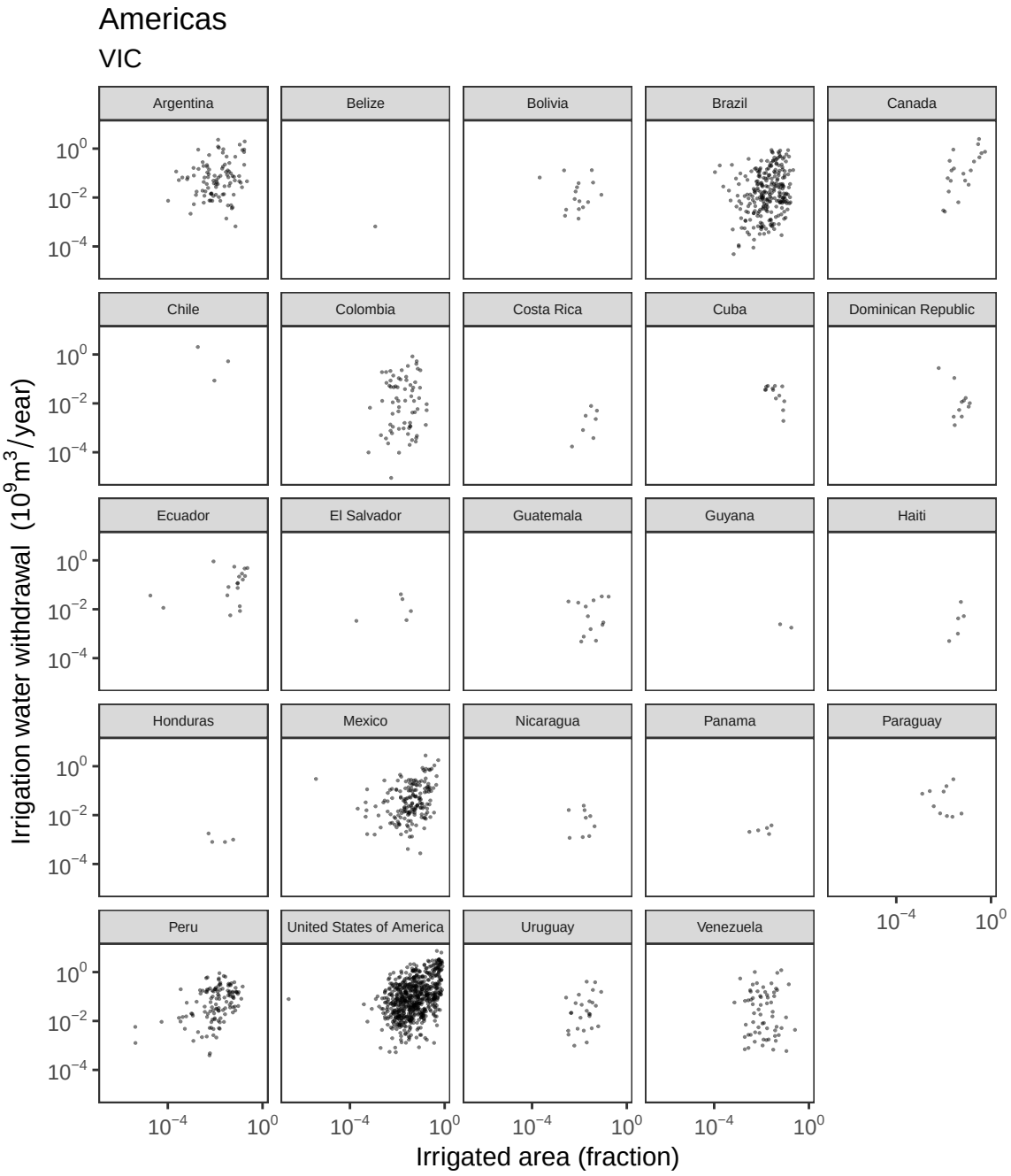

```
##
## $Americas$`MPI-HM`
```



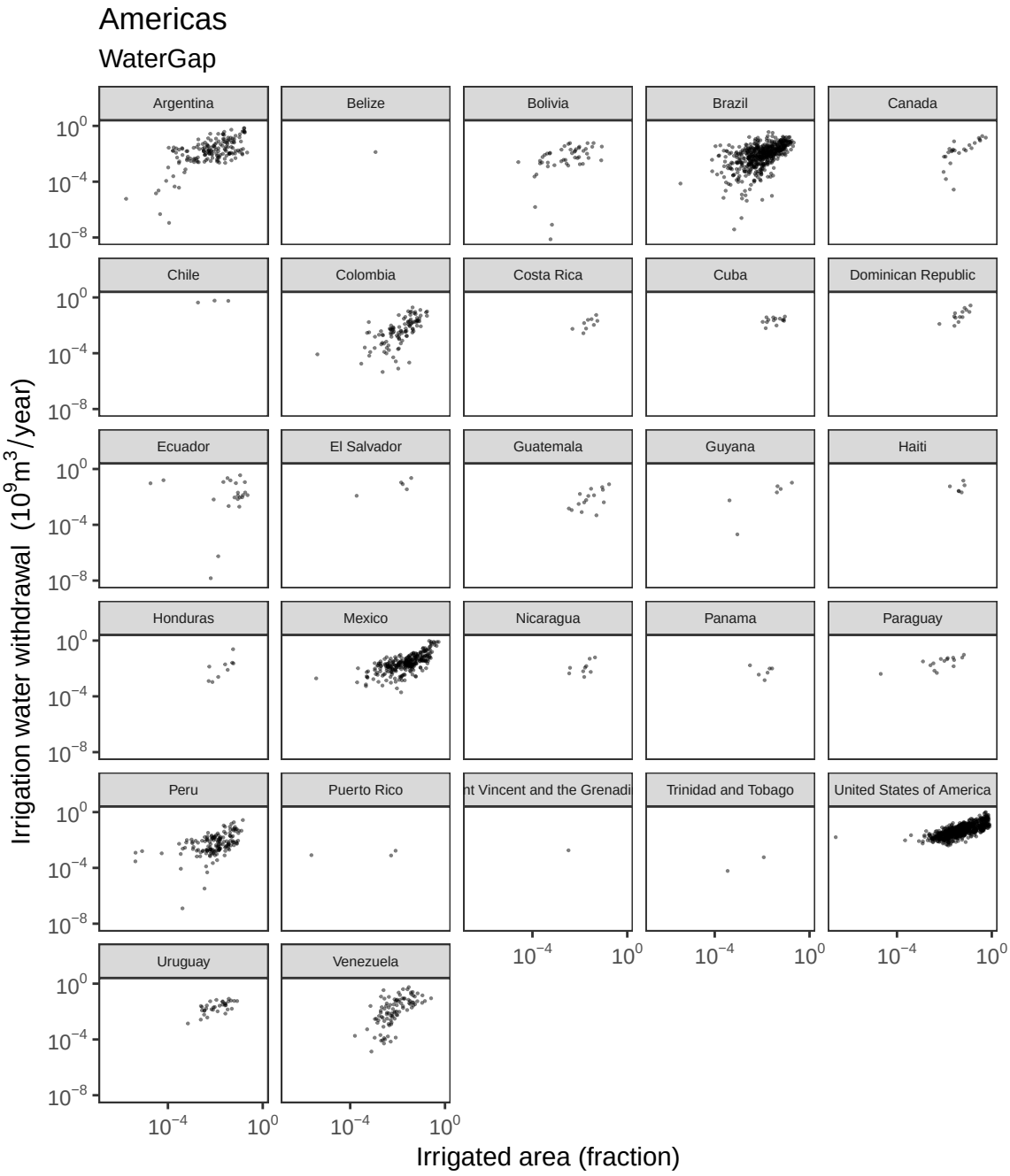

```
##
## $Americas$`PCR-GLOBWB`
```



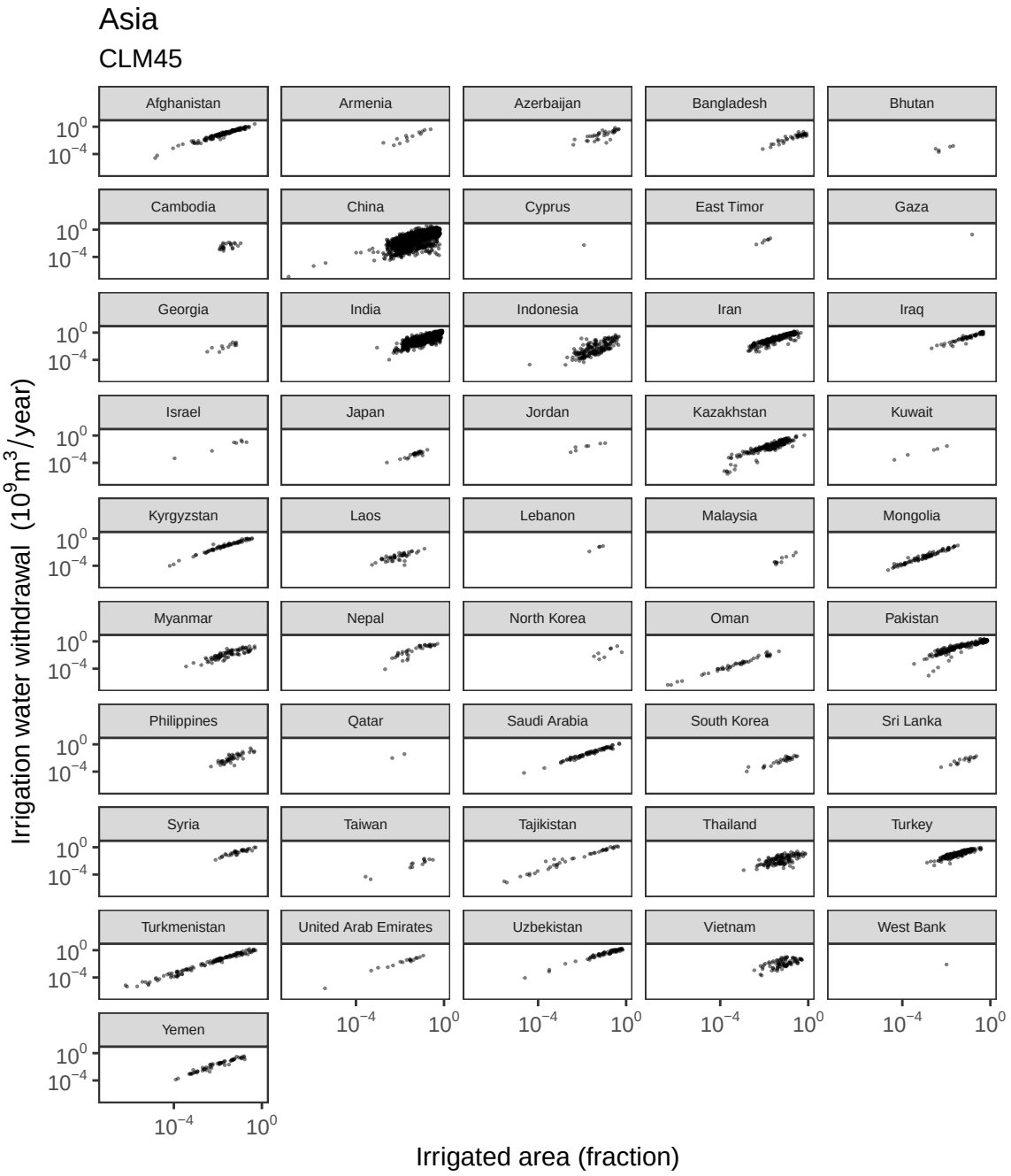

```
##
## $Americas$VIC
```




```
##
## $Americas$WaterGap
```

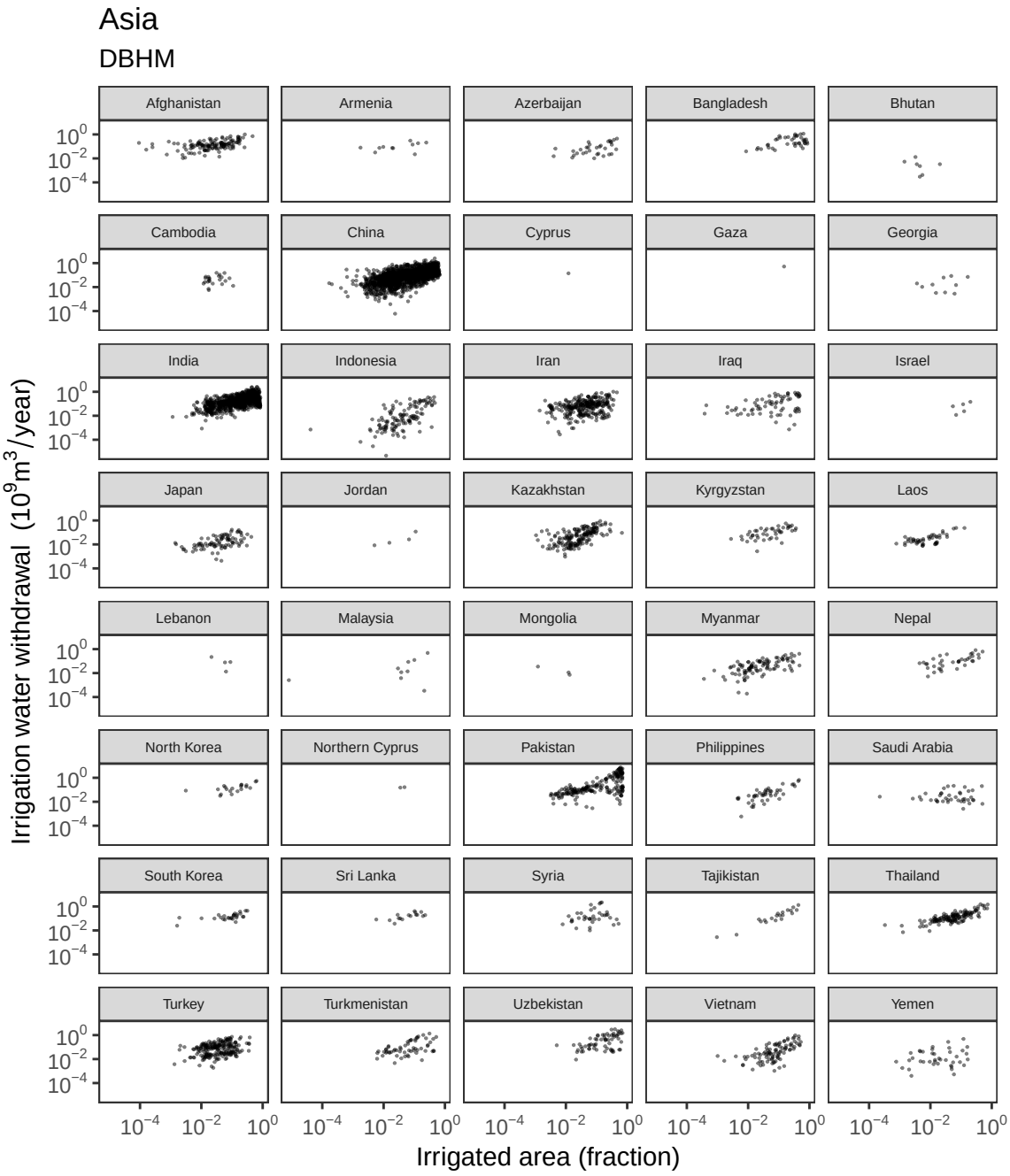



```
##
##
## $Asia
## $Asia$CLM45
```

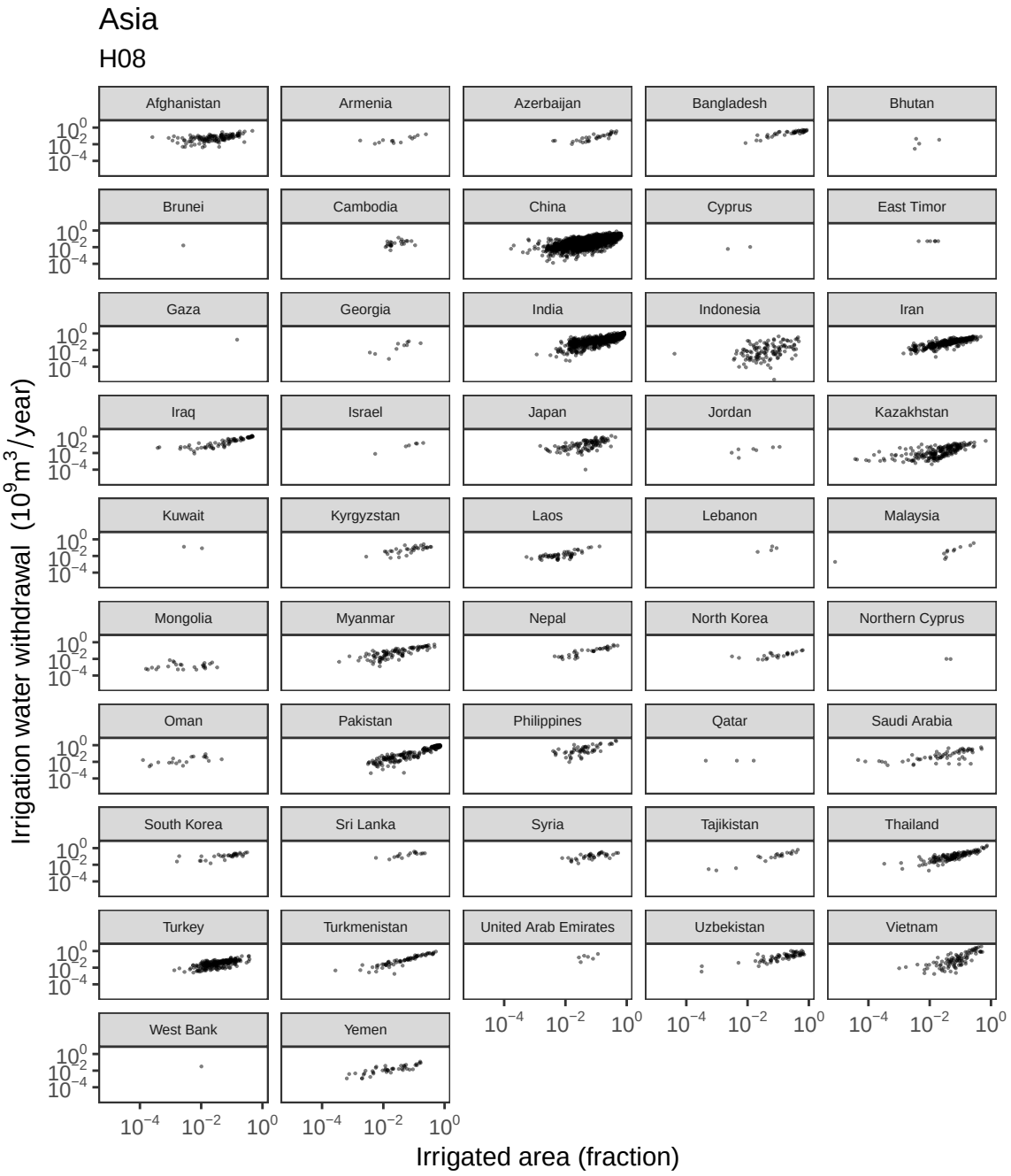

Asia
CLM45
Afghanistan
Armenia
Azerbaijan
Bangladesh
Bhutan
Cambodia
China
Cyprus
East Timor
Gaza
Georgia
India
Indonesia
Iran
Iraq
Israel
Japan
Jordan
Kazakhstan
Kuwait
Kyrgyzstan
Laos
Lebanon
Malaysia
Mongolia
Myanmar
Nepal
North Korea
Oman
Pakistan
Philippines
Qatar
Saudi Arabia
South Korea
Sri Lanka
Syria
Taiwan
Tajikistan
Thailand
Turkey
Turkmenistan
United Arab Emirates
Uzbekistan
Vietnam
West Bank
Yemen
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

##

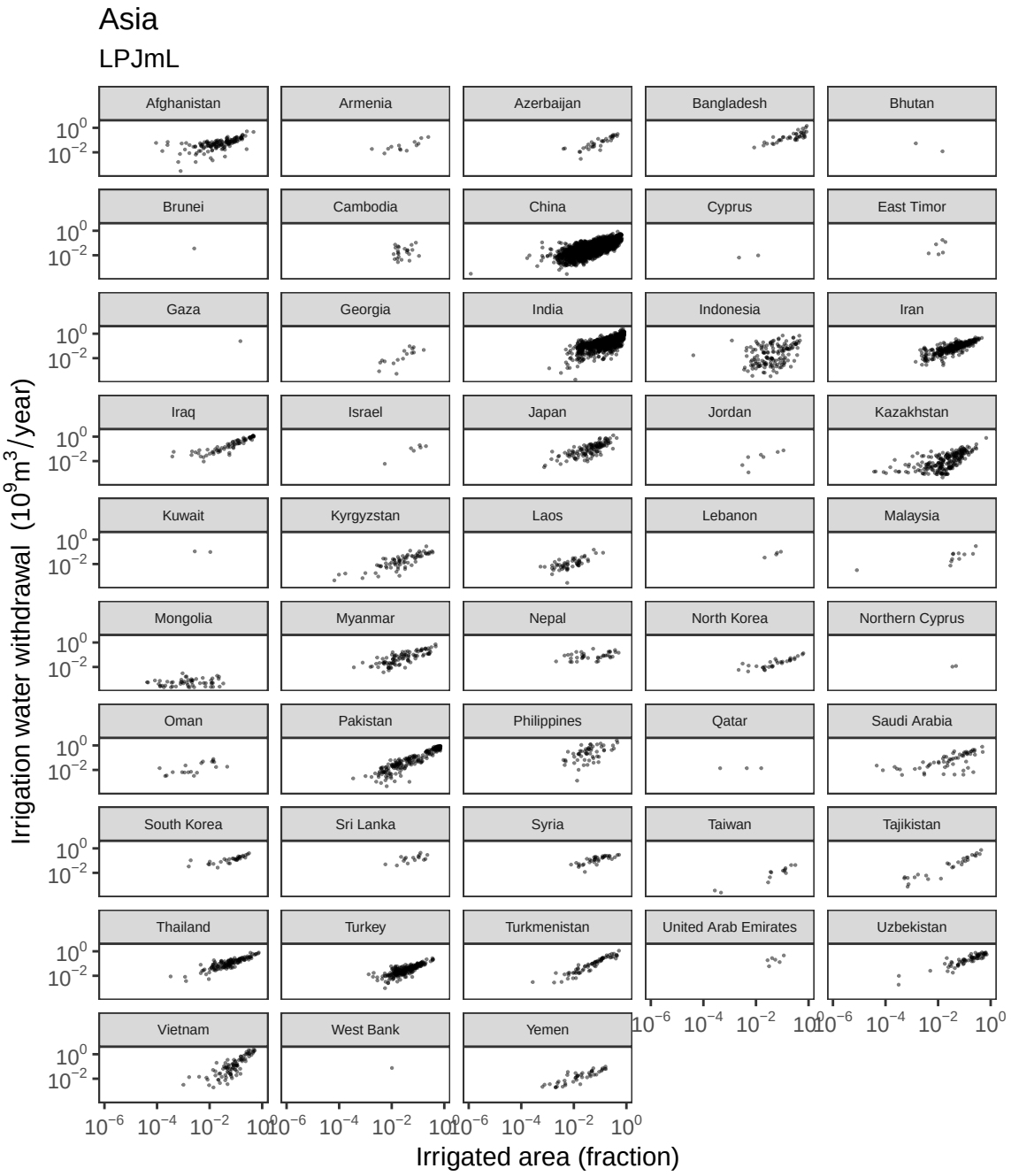
$Asia$DBHM


Asia
DBHM
Afghanistan
Armenia
Azerbaijan
Bangladesh
Bhutan
Cambodia
China
Cyprus
Gaza
Georgia
India
Indonesia
Iran
Iraq
Israel
Japan
Jordan
Kazakhstan
Kyrgyzstan
Laos
Lebanon
Malaysia
Mongolia
Myanmar
Nepal
North Korea
Northern Cyprus
Pakistan
Philippines
Saudi Arabia
South Korea
Sri Lanka
Syria
Tajikistan
Thailand
Turkey
Turkmenistan
Uzbekistan
Vietnam
Yemen
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

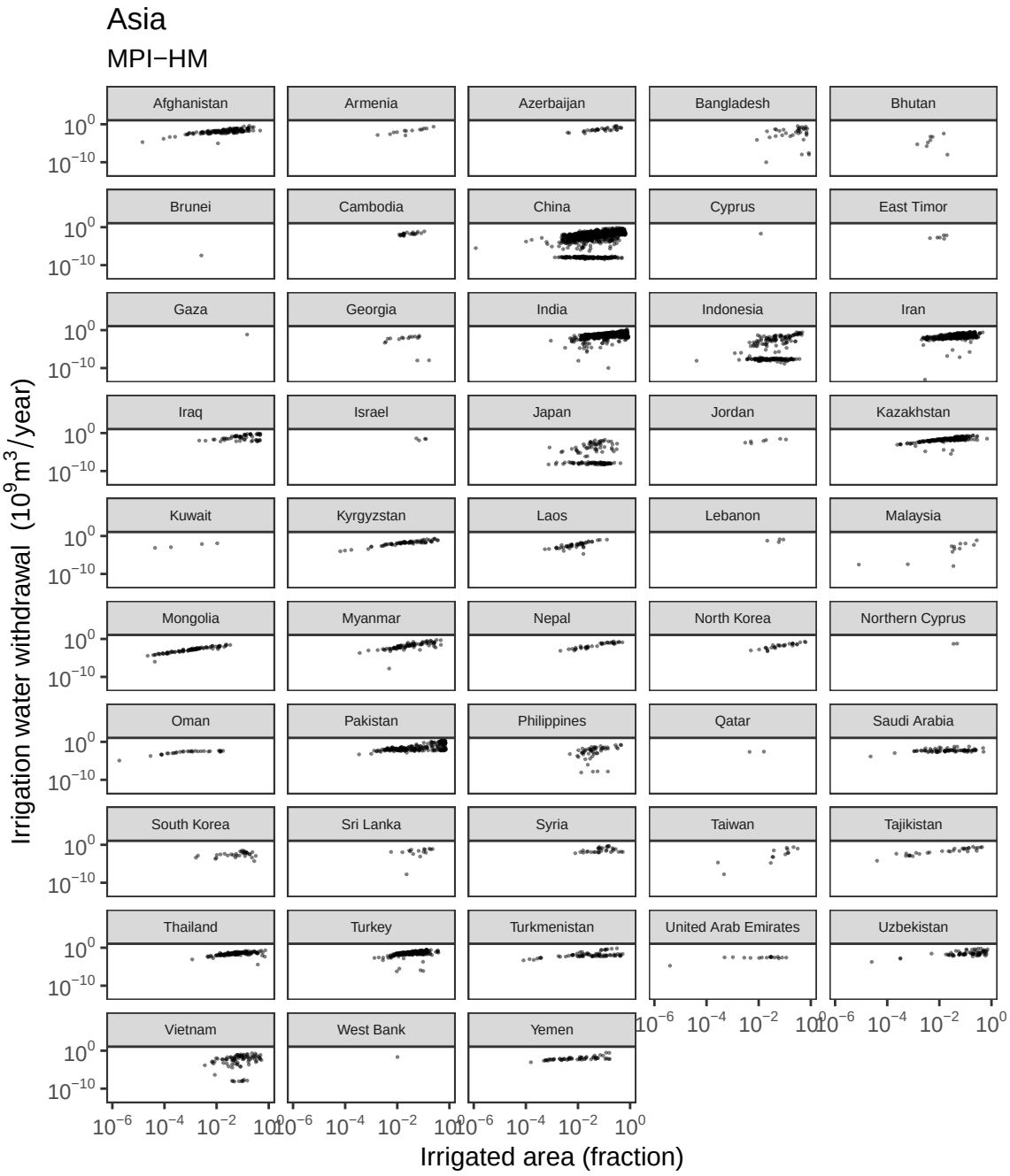
```
##
## $Asia$H08
```


Asia
H08
Afghanistan
Armenia
Azerbaijan
Bangladesh
Bhutan
Brunei
Cambodia
China
Cyprus
East Timor
Gaza
Georgia
India
Indonesia
Iran
Iraq
Israel
Japan
Jordan
Kazakhstan
Kuwait
Kyrgyzstan
Laos
Lebanon
Malaysia
Mongolia
Myanmar
Nepal
North Korea
Northern Cyprus
Oman
Pakistan
Philippines
Qatar
Saudi Arabia
South Korea
Sri Lanka
Syria
Tajikistan
Thailand
Turkey
Turkmenistan
United Arab Emirates
Uzbekistan
Vietnam
West Bank
Yemen
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

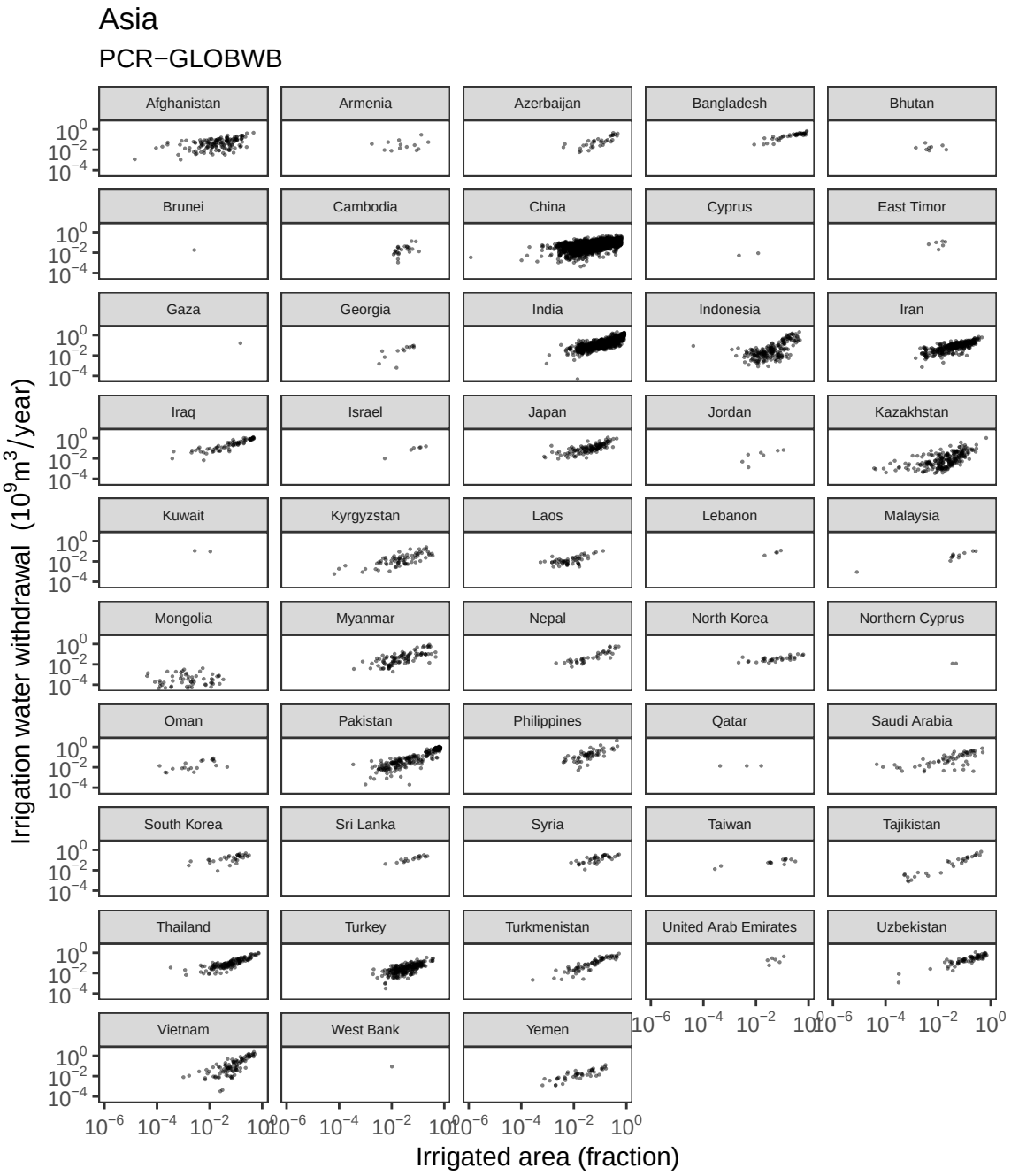
```
##
## $Asia$LPJmL
```


Asia
LPJmL
Afghanistan
Armenia
Azerbaijan
Bangladesh
Bhutan
Brunei
Cambodia
China
Cyprus
East Timor
Gaza
Georgia
India
Indonesia
Iran
Iraq
Israel
Japan
Jordan
Kazakhstan
Kuwait
Kyrgyzstan
Laos
Lebanon
Malaysia
Mongolia
Myanmar
Nepal
North Korea
Northern Cyprus
Oman
Pakistan
Philippines
Qatar
Saudi Arabia
South Korea
Sri Lanka
Syria
Taiwan
Tajikistan
Thailand
Turkey
Turkmenistan
United Arab Emirates
Uzbekistan
Vietnam
West Bank
Yemen
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

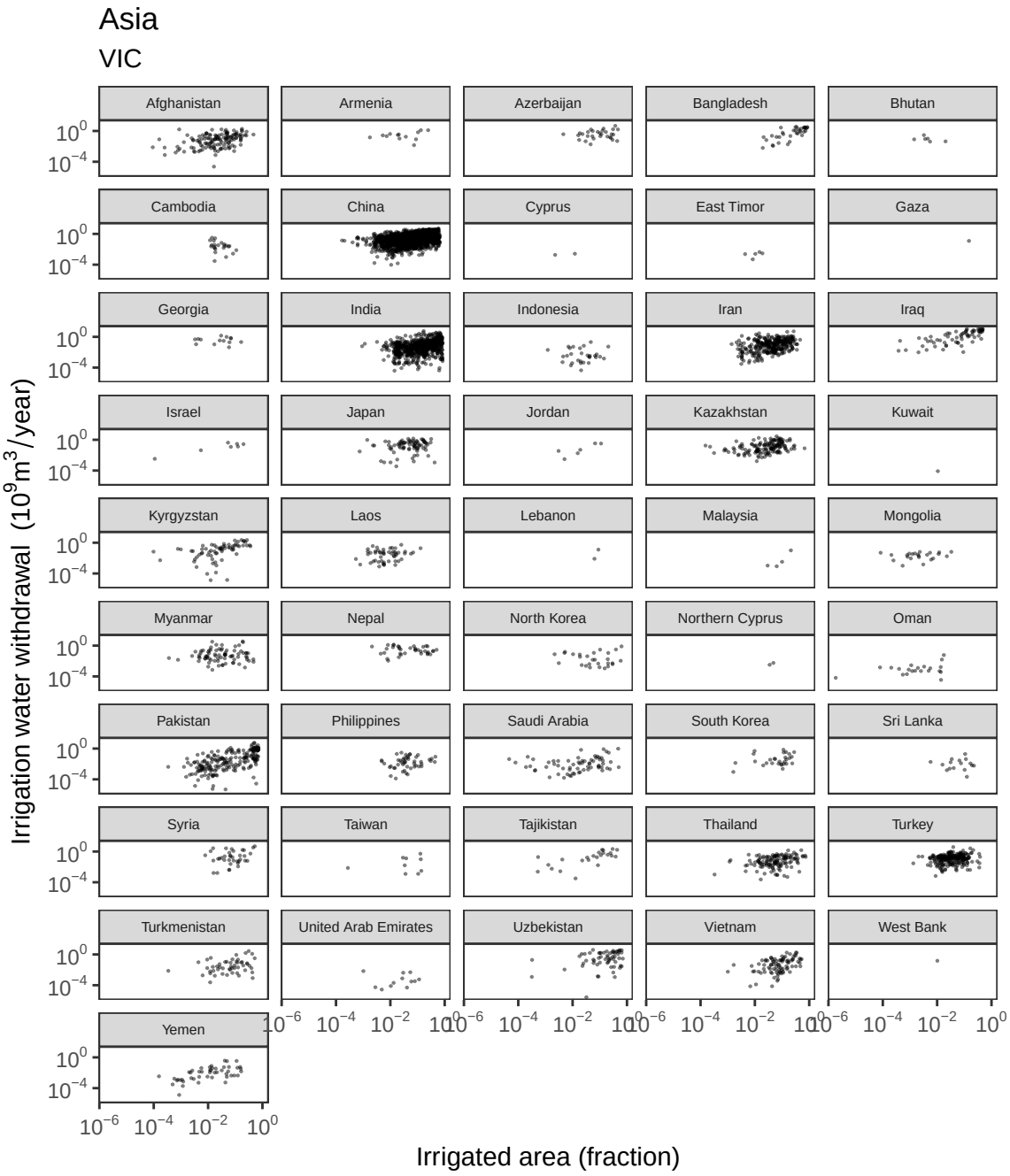
```
##
## $Asia$`MPI-HM`
```



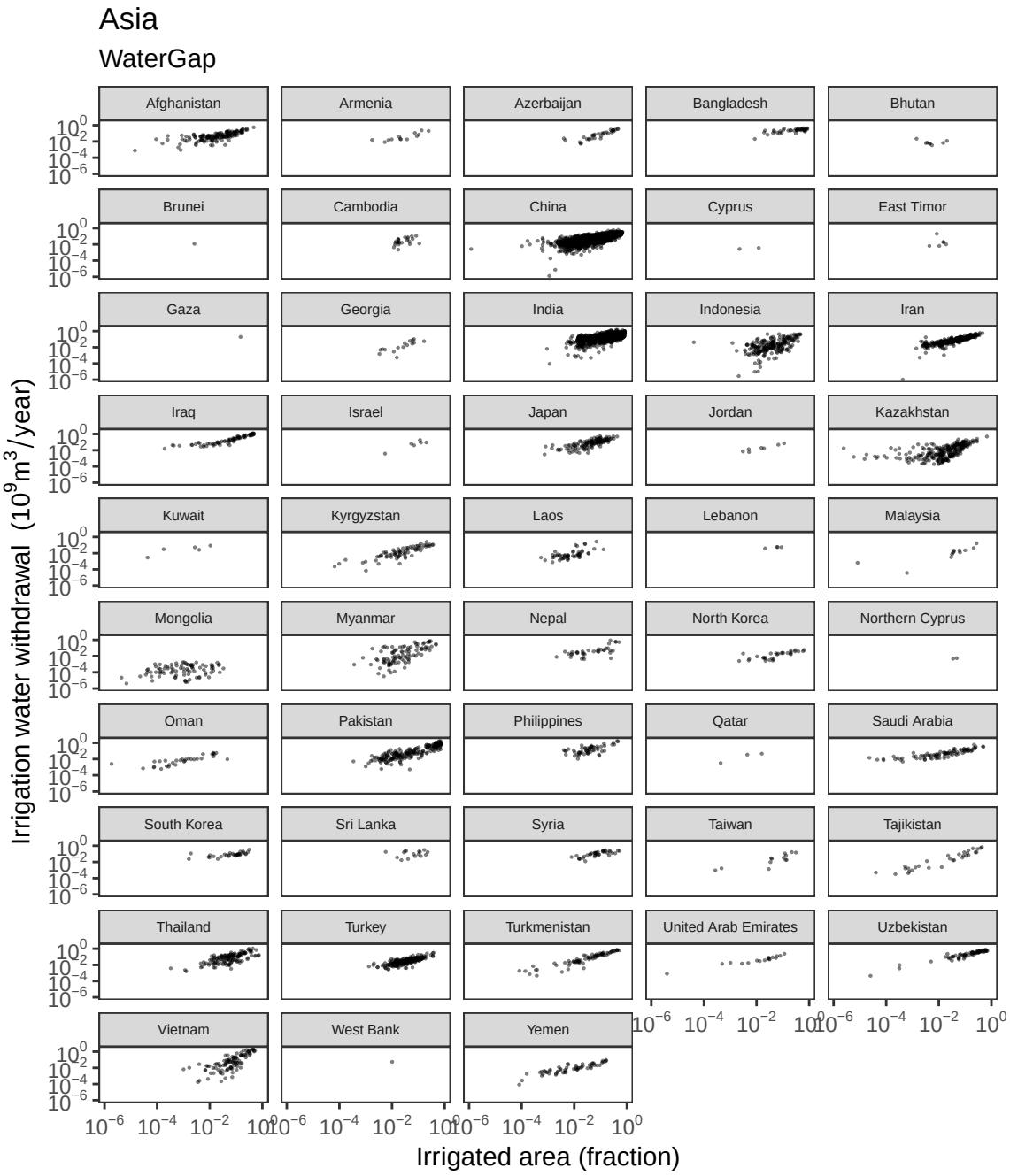

```
##
## $Asia$`PCR-GLOBWB`
```


Asia
PCR−GLOBWB
Afghanistan
Armenia
Azerbaijan
Bangladesh
Bhutan
Brunei
Cambodia
China
Cyprus
East Timor
Gaza
Georgia
India
Indonesia
Iran
Iraq
Israel
Japan
Jordan
Kazakhstan
Kuwait
Kyrgyzstan
Laos
Lebanon
Malaysia
Mongolia
Myanmar
Nepal
North Korea
Northern Cyprus
Oman
Pakistan
Philippines
Qatar
Saudi Arabia
South Korea
Sri Lanka
Syria
Taiwan
Tajikistan
Thailand
Turkey
Turkmenistan
United Arab Emirates
Uzbekistan
Vietnam
West Bank
Yemen
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

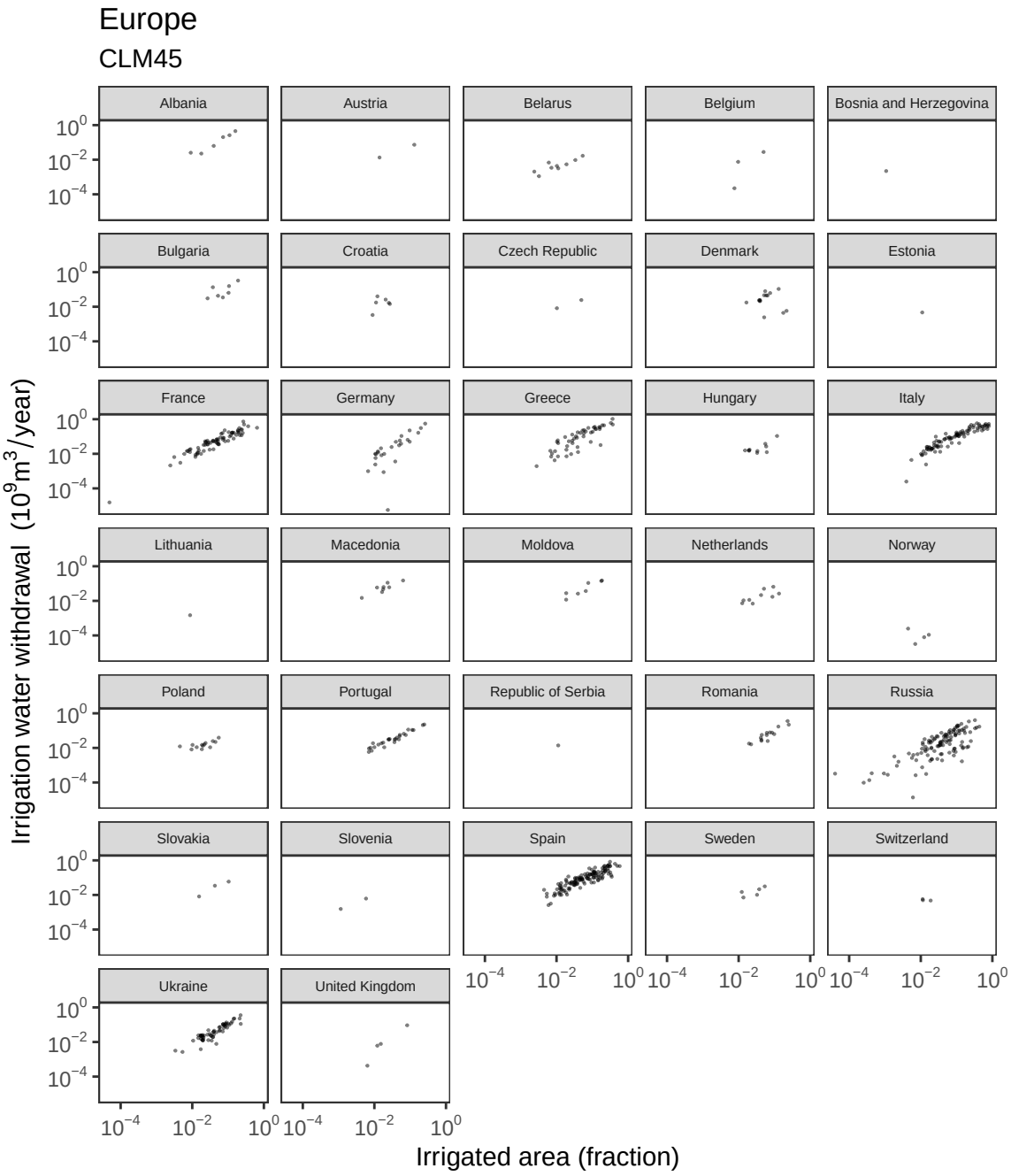
```
##
## $Asia$VIC
```



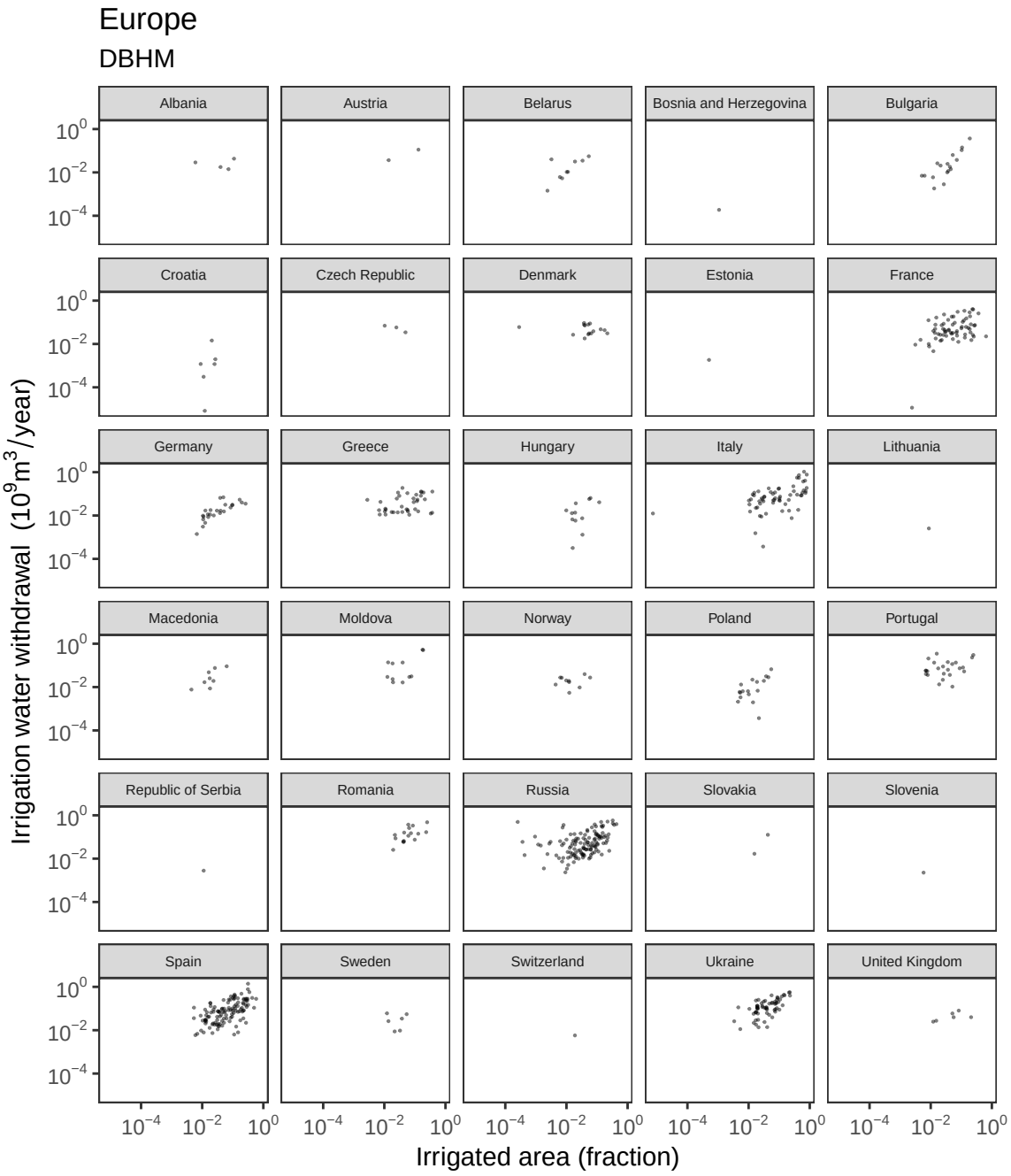

```
##
## $Asia$WaterGap
```




```
##
##
## $Europe
## $Europe$CLM45
```

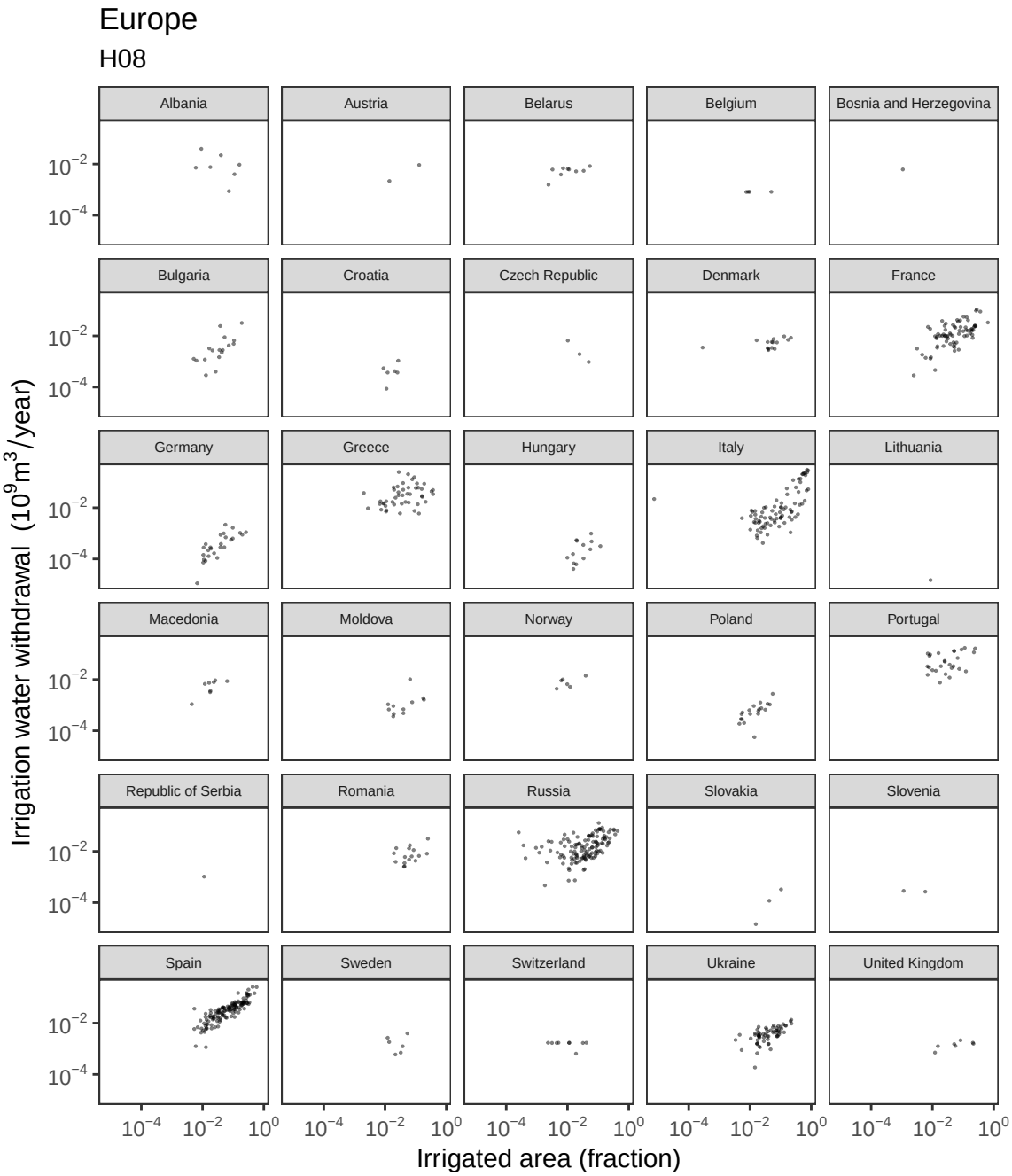

Europe
CLM45
Albania
Austria
Belarus
Belgium
Bosnia and Herzegovina
Bulgaria
Croatia
Czech Republic
Denmark
Estonia
France
Germany
Greece
Hungary
Italy
Lithuania
Macedonia
Moldova
Netherlands
Norway
Poland
Portugal
Republic of Serbia
Romania
Russia
Slovakia
Slovenia
Spain
Sweden
Switzerland
Ukraine
United Kingdom
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

```
##
## $Europe$DBHM
```


Europe
DBHM
Albania
Austria
Belarus
Bosnia and Herzegovina
Bulgaria
Croatia
Czech Republic
Denmark
Estonia
France
Germany
Greece
Hungary
Italy
Lithuania
Macedonia
Moldova
Norway
Poland
Portugal
Republic of Serbia
Romania
Russia
Slovakia
Slovenia
Spain
Sweden
Switzerland
Ukraine
United Kingdom
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

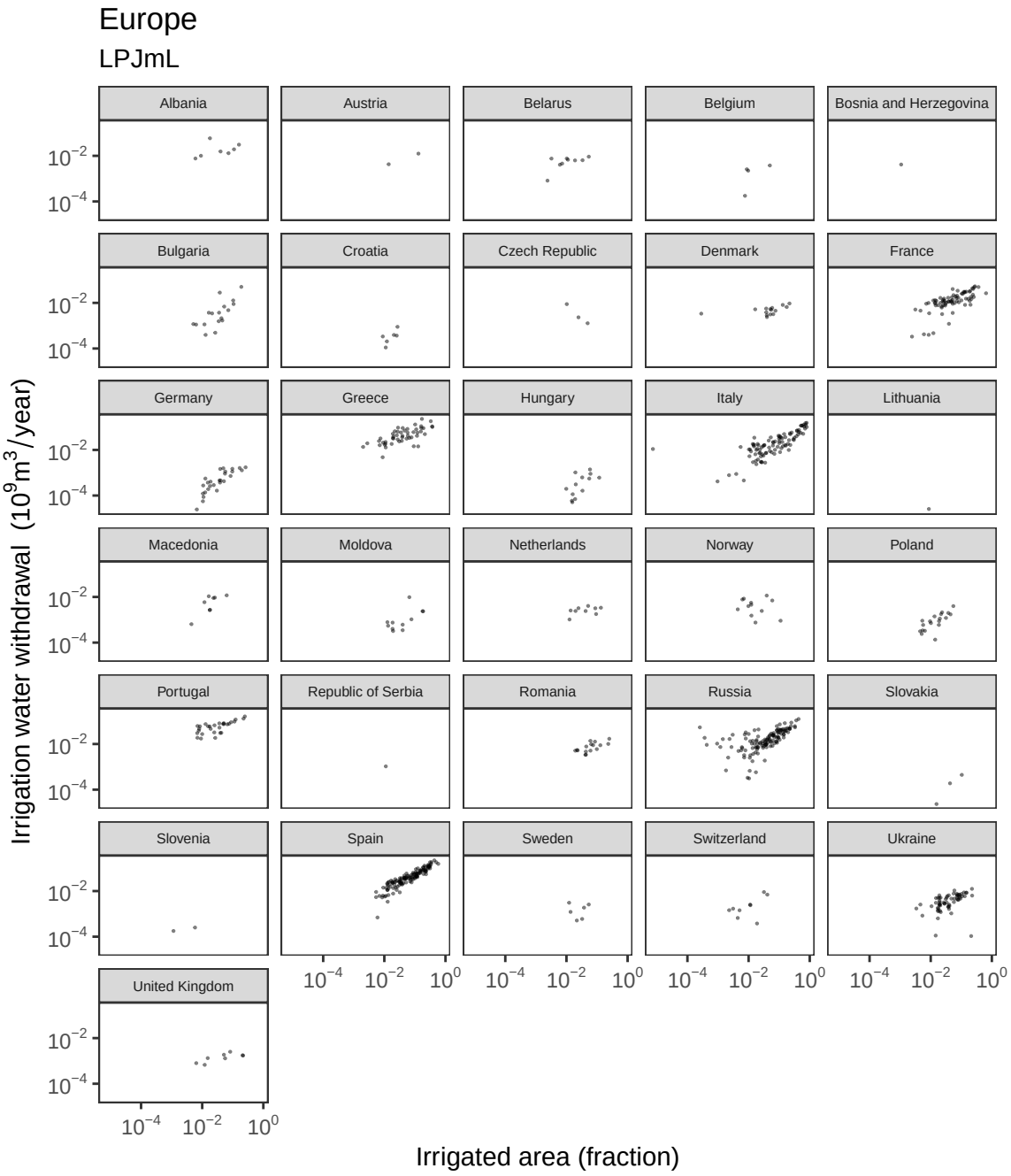
##

$Europe$H08

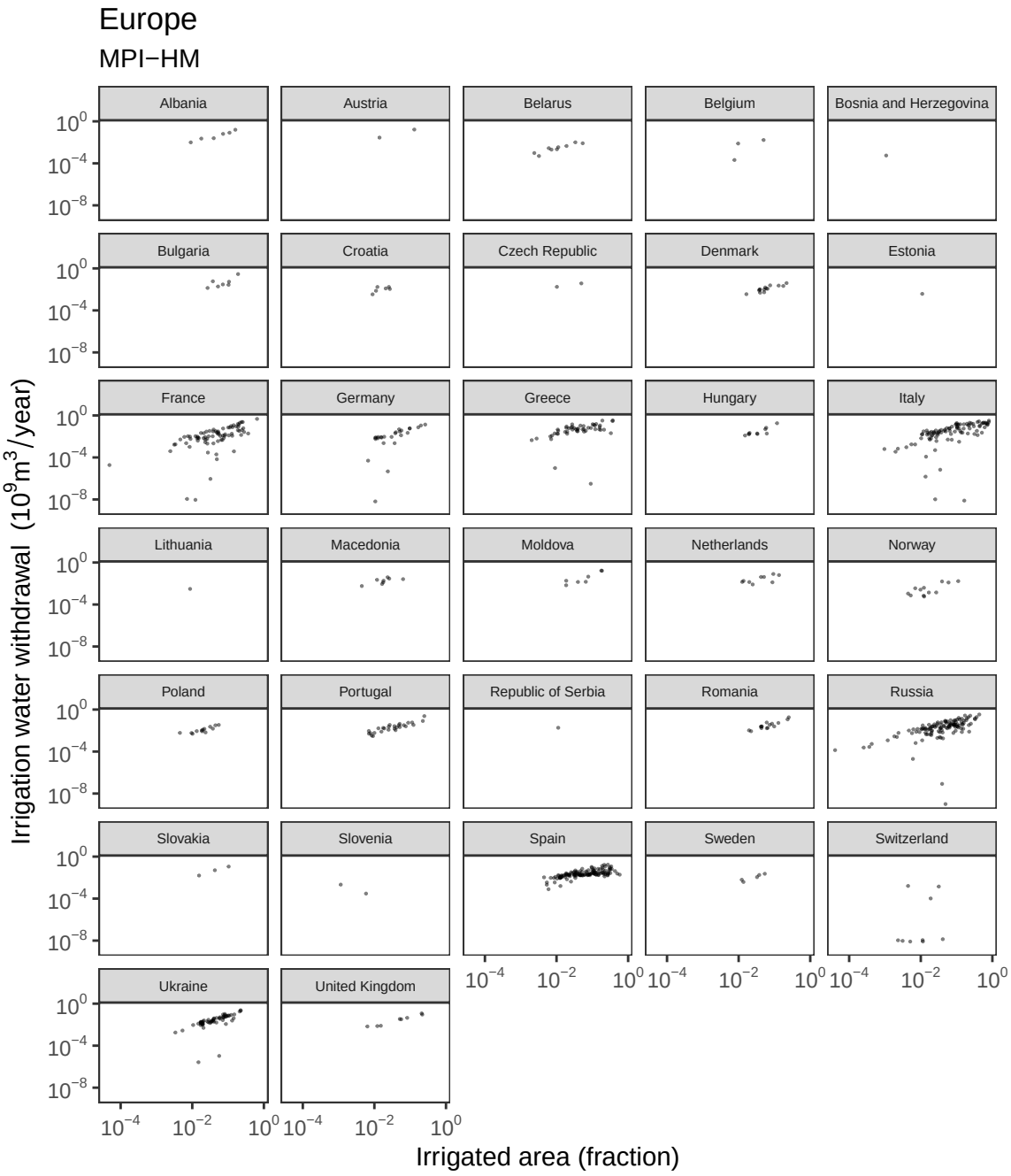


##

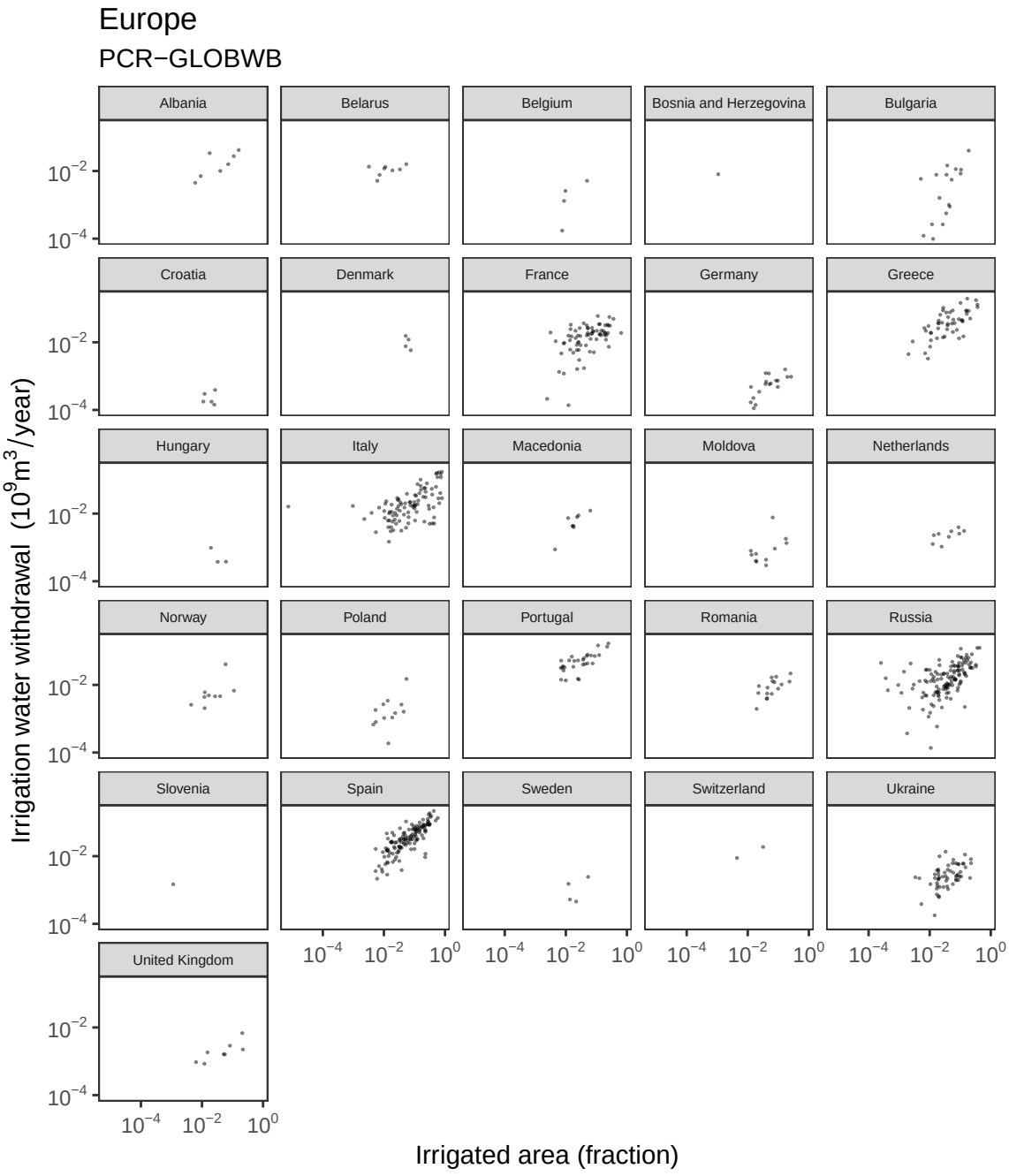
$Europe$LPJmL



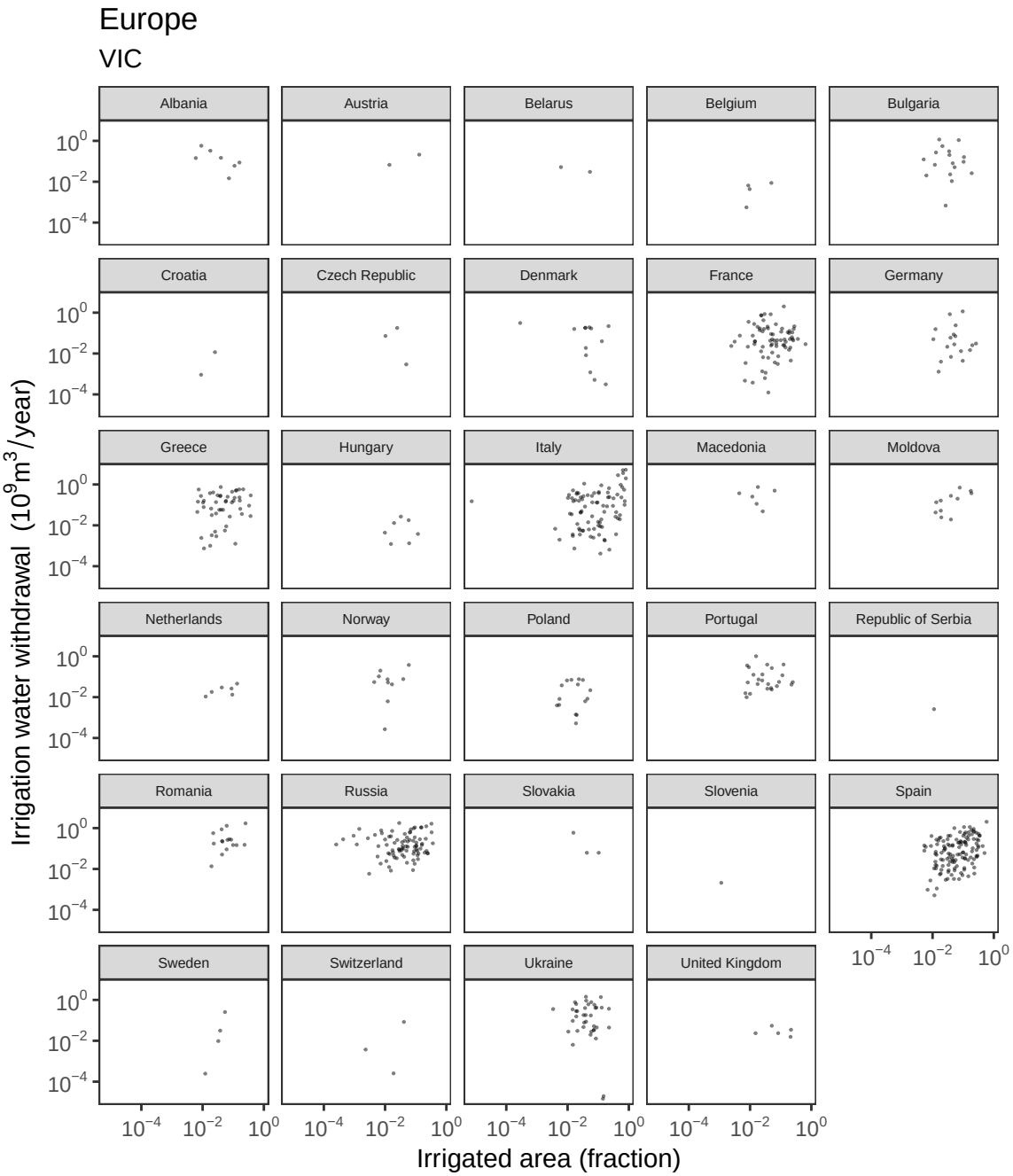

```
##
## $Europe$`MPI-HM`
```



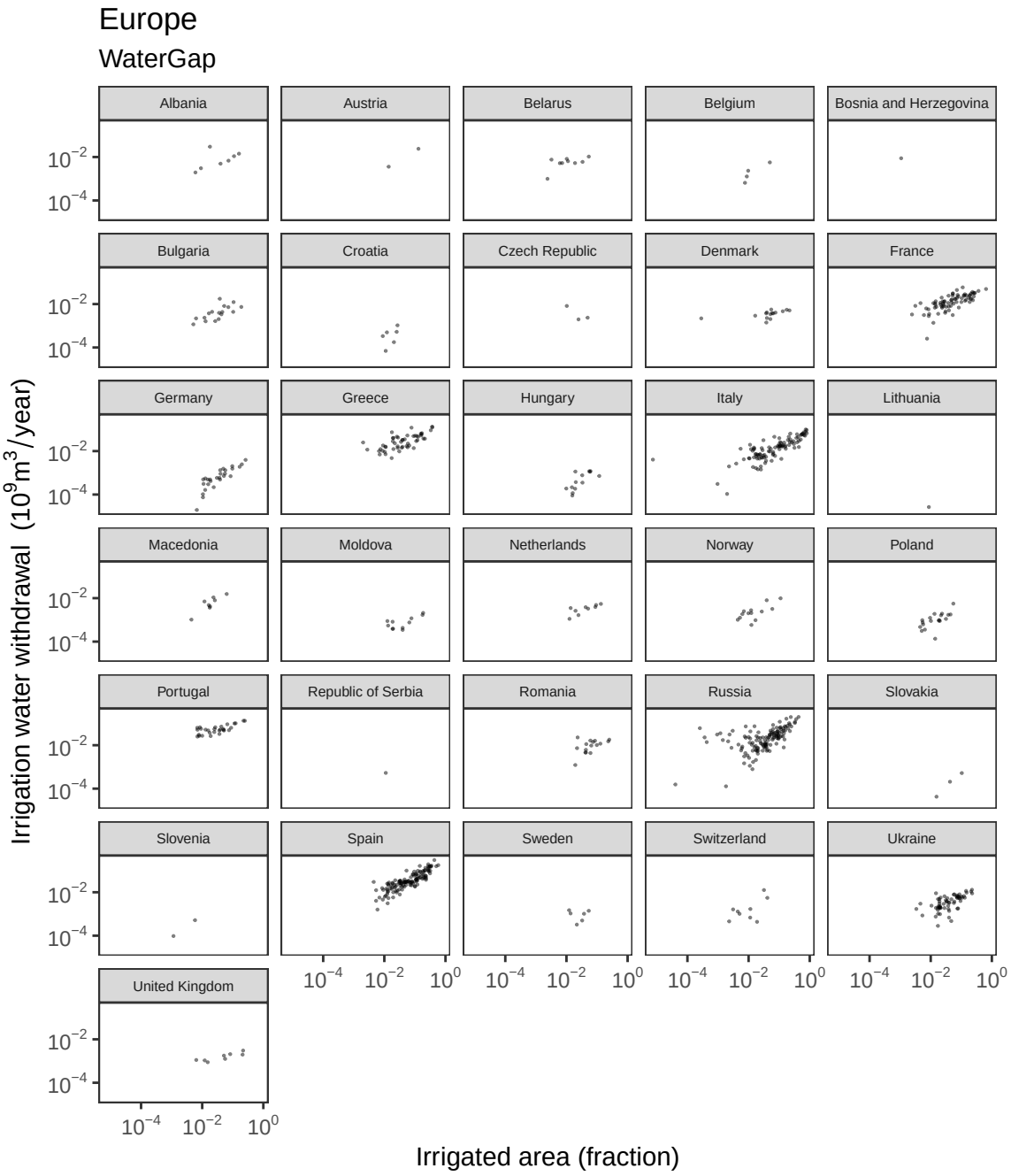

```
##
## $Europe$`PCR-GLOBWB`
```


Europe
PCR−GLOBWB
Albania
Belarus
Belgium
Bosnia and Herzegovina
Bulgaria
Croatia
Denmark
France
Germany
Greece
Hungary
Italy
Macedonia
Moldova
Netherlands
Norway
Poland
Portugal
Romania
Russia
Slovenia
Spain
Sweden
Switzerland
Ukraine
United Kingdom
Irrigation water withdrawal (10^9 m^3/year)
Irrigated area (fraction)

```
##
## $Europe$VIC
```




```
##
## $Europe$WaterGap
```



12 Session information

```
# SESSION INFORMATION ------------------------------------------------------------

sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
```

```
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] grid      parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] checkpoint_0.4.9  rsample_0.0.7     robustbase_0.93-6 gridExtra_2.3
##  [5] forcats_0.5.0     stringr_1.4.0     dplyr_1.0.2       purrr_0.3.4
##  [9] readr_1.3.1       tidyr_1.1.2       tibble_3.0.3      ggplot2_3.3.2
## [13] tidyverse_1.3.0   benchmarkme_1.0.4 cowplot_1.1.0     naniar_0.6.0
## [17] broom_0.7.0       ggridges_0.5.2    mice_3.11.0       lmtest_0.9-37
## [21] zoo_1.8-8         sandwich_2.5-1    mvoutlier_2.0.9   sgeostat_1.0-27
## [25] complmrob_0.7.0   doParallel_1.0.15 iterators_1.0.12  foreach_1.5.0
## [29] MASS_7.3-52       scales_1.1.1      boot_1.3-25       IDPmisc_1.1.20
## [33] countrycode_1.2.0 rworldmap_1.3-6   sp_1.4-2          ncdf4_1.17
## [37] sensobol_0.3      data.table_1.12.8
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1          backports_1.1.9       spam_2.5-1
##   [4] plyr_1.8.6            splines_3.6.3         listenv_0.8.0
##   [7] digest_0.6.25         htmltools_0.5.0       fansi_0.4.1
##  [10] magrittr_1.5          cluster_2.1.0         openxlsx_4.1.5
##  [13] globals_0.13.0        tikzDevice_0.12.3.1   modelr_0.1.8
##  [16] colorspace_1.4-1      blob_1.2.1            rvest_0.3.6
##  [19] rrcov_1.5-5           haven_2.3.1           xfun_0.16
##  [22] crayon_1.3.4          jsonlite_1.7.0        survival_3.2-3
##  [25] glue_1.4.1.9000       gtable_0.3.0          car_3.0-9
##  [28] kernlab_0.9-29        RcppZiggurat_0.1.5    prabclus_2.3-2
##  [31] DEoptimR_1.0-8        maps_3.3.0            abind_1.4-5
##  [34] VIM_5.1.1             mvtnorm_1.1-1         DBI_1.1.0
##  [37] GGally_2.0.0          bibtex_0.4.2.2        Rcpp_1.0.5
##  [40] sROC_0.1-2            laeken_0.5.1          foreign_0.8-75
##  [43] mclust_5.4.6          dotCall64_1.0-0       stats4_3.6.3
##  [46] vcd_1.4-7             truncnorm_1.0-8       httr_1.4.2
##  [49] RColorBrewer_1.1-2    fpc_2.2-7             modeltools_0.2-23
##  [52] ellipsis_0.3.1        farver_2.0.3          pkgconfig_2.0.3
##  [55] reshape_0.8.8         NADA_1.6-1.1          flexmix_2.3-15
##  [58] nnet_7.3-14           dbplyr_1.4.4          labeling_0.3
##  [61] tidyselect_1.1.0      rlang_0.4.7           munsell_0.5.0
##  [64] cellranger_1.1.0      tools_3.6.3           cli_2.0.2
##  [67] generics_0.0.2        ranger_0.12.1         pls_2.7-3
```

```
##  [70] evaluate_0.14         cvTools_0.3.2         yaml_2.2.1
##  [73] knitr_1.29            fs_1.5.0              filehash_2.4-2
##  [76] zip_2.1.1             visdat_0.5.3          nlme_3.1-149
##  [79] future_1.18.0         xml2_1.3.2            compiler_3.6.3
##  [82] rstudioapi_0.11       curl_4.3              e1071_1.7-3
##  [85] zCompositions_1.3.4   reprex_0.3.0          robCompositions_2.2.1
##  [88] pcaPP_1.9-73          stringi_1.4.6         highr_0.8
##  [91] fields_11.4           lattice_0.20-41       Matrix_1.2-18
##  [94] vctrs_0.3.4           furrr_0.1.0           pillar_1.4.6
##  [97] lifecycle_0.2.0       Rdpack_1.0.0          maptools_1.0-2
## [100] gbRd_0.4-11           R6_2.4.1              rio_0.5.16
## [103] codetools_0.2-16      benchmarkmeData_1.0.4 assertthat_0.2.1
## [106] withr_2.2.0           mgcv_1.8-33           diptest_0.75-7
## [109] hms_0.5.3             class_7.3-17          Rfast_2.0.0
## [112] rmarkdown_2.3         carData_3.0-4         lubridate_1.7.9
## [115] tinytex_0.25
```

```
## Return the machine CPU
cat("Machine: "); print(get_cpu()$model_name)
```

```
## Machine:
```

```
## [1] "Intel(R) Core(TM) i7-3520M CPU @ 2.90GHz"
```

```
## Return number of true cores
cat("Num cores: "); print(detectCores(logical = FALSE))
```

```
## Num cores:
```

```
## [1] 2
```

```
## Return number of threads
cat("Num threads: "); print(detectCores(logical = TRUE))
```

```
## Num threads:
```

```
## [1] 4
```

```
## Return the machine RAM
cat("RAM: "); print (get_ram()); cat("\n")
```

```
## RAM:
```

```
## 17.2 GB
```

References

FAO. 2016. "AQUASTAT website." Food; Agriculture Organization of the United Nations. http://www.fao.org/nr/water/aquastat/didyouknow/index3.stm.

———. 2017. "FAOSTAT database." Rome. http://www.fao.org/faostat/en/.

Frenken, Karen, and Virginie Gillet. 2012. "Irrigation water requirement and water withdrawal by country." Food; Agriculture Organization of the United Nations. http://www.fao.org/3/a-

bc824e.pdf.

Huang, Zhongwei, Mohamad Hejazi, Xinya Li, Qiuhong Tang, Chris Vernon, Guoyong Leng, Yaling Liu, et al. 2018. "Global gridded monthly sectoral water use dataset for 1971-2010: v2." https://doi.org/10.5281/ZENODO.1209296.

Jansen, M. 1999. "Analysis of variance designs for model output." *Computer Physics Communications* 117 (1-2): 35–43. https://doi.org/10.1016/S0010-4655(98)00154-4.

Liu, Yaling, Mohamad Hejazi, Page Kyle, Son H. Kim, Evan Davies, Diego G. Miralles, Adriaan J. Teuling, Yujie He, and Dev Niyogi. 2016. "Global and regional evaluation of energy for water." *Environmental Science and Technology* 50 (17): 9736–45. https://doi.org/10.1021/acs.est.6b01065.

Meier, J., F. Zabel, and W. Mauser. 2018. "A global approach to estimate irrigated areas . A comparison between different data and statistics." *Hydrology and Earth System Sciences* 22 (2): 1119–33. https://doi.org/10.5194/hess-22-1119-2018.

Puy, Arnald, Samuele Lo Piano, Andrea Saltelli, and Simon Levin. n.d. "sensobol: an R package to compute high-order sensitivity indices."

Rohwer, Janine, Dieter Gerten, and Wolfgang Lucht. 2007. "Development of functional irrigation types for improved global crop modelling." *PIK Report*, no. 104: 1–61.

Salmon, J.M., M. A. Friedl, S. Frolking, D. Wisser, and E. M. Douglas. 2015. "Global rain-fed, irrigated, and paddy croplands: A new high resolution map derived from remote sensing, crop inventories and climate data." *International Journal of Applied Earth Observation and Geoinformation* 38. Elsevier B.V.: 321–34. https://doi.org/10.1016/j.jag.2015.01.014.

Siebert, S., V. Henrich, K. Frenken, and J. Burke. 2013. "Update of the digital global map of irrigation areas to version 5." Rome: Rheinische Friedrich-Wilhelms-University; Food; Agriculture Organization of the United Nations.

Sobol', Ilya. M. 1967. "On the distribution of points in a cube and the approximate evaluation of integrals." *USSR Computational Mathematics and Mathematical Physics* 7 (4): 86–112. https://doi.org/10.1016/0041-5553(67)90144-9.

———. 1976. "Uniformly distributed sequences with an additional uniform property." *USSR Computational Mathematics and Mathematical Physics* 16 (5): 236–42. https://doi.org/10.1016/0041-5553(76)90154-3.

Thenkabail, P. S., C. M. Biradar, P. Noojipady, V. Dheeravath, Y. Li, M. Velpuri, M. Gumma, et al. 2009. "Global irrigated area map (GIAM), derived from remote sensing, for the end of the last millennium." *International Journal of Remote Sensing* 30 (14): 3679–3733. https://doi.org/10.1080/01431160802698919.