

# Irrigated areas drive irrigation water withdrawals

R code

Arnald Puy

April 7 2020

## Contents

<b>1 Presentation</b>	<b>3</b>
1.1 Preliminary functions . . . . .	4
<b>2 Load the datasets</b>	<b>5</b>
2.1 Define loading functions . . . . .	5
2.2 Load Huang et al. datasets . . . . .	7
2.3 Load historic FAO-GMIA . . . . .	8
2.4 Long-term trends between irrigated areas and irrigation water withdrawals . . . . .	9
2.5 Plot trend between irrigated areas and irrigation water withdrawals . . . . .	18
<b>3 The influence of other parameters</b>	<b>22</b>
3.1 Potential evapotranspiration . . . . .	22
3.2 Potential evaporation . . . . .	24
3.3 Water efficiency . . . . .	25
<b>4 Creation of the final dataset</b>	<b>26</b>
<b>5 Missing values</b>	<b>32</b>
5.1 Imputation of missing values . . . . .	32
<b>6 Compute linear regressions</b>	<b>33</b>
<b>7 Predictions</b>	<b>36</b>
7.1 Compare our approach with estimates by GHM . . . . .	36
7.2 Can future irrigation water withdrawals be estimated as a function of irrigated areas? .	45
<b>8 The model</b>	<b>51</b>
8.1 Create lookup table . . . . .	51
8.2 Sample matrix . . . . .	51
8.3 Run the model . . . . .	53
<b>9 Uncertainty analysis</b>	<b>54</b>
<b>10 Sensitivity analysis</b>	<b>57</b>

<b>11 Irrigated areas vs irrigation water withdrawals at different scales</b>	<b>62</b>
11.1 Australia (scheme level), USA (State level) and Colorado (County level) . . . . .	62
11.2 Regression diagnostics . . . . .	66
<b>12 Irrigated area at the cell level</b>	<b>71</b>
<b>13 Session information</b>	<b>106</b>
<b>14 References</b>	<b>108</b>

# 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](mailto:apuy@princeton.edu); [arnald.puy@pm.me](mailto:arnald.puy@pm.me)).

## 1.1 Preliminary functions

```
# 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", "ggrepel"))

# SET CHECKPOINT ----

dir.create(".checkpoint")

library("checkpoint")

checkpoint("2021-04-07",
           R.version ="4.0.3",
           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 the datasets

### 2.1 Define loading functions

```
# 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")]
  setcolororder(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")
  out <- lapply(selected.years, function(x) {
    water <- rowSums(ww[, vec[[x]]])
    ww.df <- data.frame(cbind(lon, lat, water))
    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
})
names(out) <- selected.years
return(out)
}

# Function to load and extract data from .nc files from ISIMIP
open_nc_files <- function(file, dname, selected.years, vec) {
  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 <- lapply(selected.years, function(x) vec[[x]])

  out <- lapply(m, function(x) {
    tmp_slice <- lapply(x, function(y) tmp_array[, , y])
    # 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) <- x
    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.Withdrawn = sum(x)), Country]
    out[, Water.Withdrawn:= Water.Withdrawn * 10000]
    out[, Continent:= countrycode(out[, Country],
                                   origin = "country.name",
                                   destination = "continent")] %>%
      .[, Code:= countrycode(out[, Country],
                             origin = "country.name",
                             destination = "un")] %>%
      .[, Country:= countrycode(out[, Code],
                                origin = "un",

```

```

            destination = "country.name")] %>%
  .[!Continent == "Oceania"]
  setcolororder(out, c("Country", "Continent", "Code", "Water.Withdrawn"))
}
return(out)
}

```

## 2.2 Load Huang et al. datasets

```

# READ IN GHM DATASETS ----

# Read in Huang et al. datasets ----

# Define vector with the months and the years
vecs <- 1:(2010 - 1970) * 12
vec <- split(vecs, ceiling(seq_along(vecs) / 12))
names(vec) <- 1971:2010
selected.years <- c("1971", "1980", "1990", "2000", "2005")

# Vector with the files
names_nc_files <- c("withd_irr_lpjml.nc", "withd_irr_pcrglobwb.nc",
                     "withd_irr_h08.nc", "withd_irr_watergap.nc")

# Read in datasets
huang.dt <- mclapply(names_nc_files, function(x) get_nc_data(x), mc.cores = detectCores() * 0.7)
names.huang <- c("LPJmL", "PCR-GLOBWB", "H08", "WaterGap")
names(huang.dt) <- names.huang

# ISIMIP datasets ----

files <- list("dbh_wfdei_nobc_hist_varsoc_co2_airrrw_global_monthly_1971_2010.nc",
              "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrrw_global_monthly_1861_2005.nc",
              "vic_wfdei_nobc_hist_pressoc_co2_airrrw_global_monthly_1971_2010.nc",
              "clm45_gfdl-esm2m_ewembi_historical_2005soc_co2_pirrrw_global_monthly_1861_2005.nc")

isimip.dt <- mclapply(files, function(x) {
  if(x == "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrrw_global_monthly_1861_2005.nc") {
    vecs <- 1:(2005 - 1860) * 12
    vec <- split(vecs, ceiling(seq_along(vecs) / 12))
    names(vec) <- 1861:2005
    dname <- "airrrw"
  } else if (x == "clm45_gfdl-esm2m_ewembi_historical_2005soc_co2_pirrrw_global_monthly_1861_2005.nc") {
    vecs <- 1:(2005 - 1860) * 12
    vec <- split(vecs, ceiling(seq_along(vecs) / 12))
    names(vec) <- 1861:2005
    dname <- "pirrrw"
  } else {
    vecs <- 1:(2010 - 1970) * 12
    vec <- split(vecs, ceiling(seq_along(vecs) / 12))
    names(vec) <- 1971:2010
    dname <- "airrrw"
  }
  get_nc_data(x)
}, mc.cores = detectCores() * 0.7)

```

```

vecs <- 1:(2010 - 1970) * 12
vec <- split(vecs, ceiling(seq_along(vecs) / 12))
names(vec) <- 1971:2010
dname <- "airrww"
}
open_nc_files(file = x, dname = dname, selected.years = selected.years,
              vec = vec)
},
mc.cores = detectCores() * 0.75

names.isimip <- c("DBHM", "MPI-HM", "VIC", "CLM45")
names(isimip.dt) <- names.isimip

for(i in names(isimip.dt)) {
  names(isimip.dt[[i]]) <- selected.years
}

```

## 2.3 Load historic FAO-GMIA

```

# READ IN HISTORIC FAO-GMIA -----
gmia.hist <- fread("FAO_GMIA_historic.csv")
gmia.hist <- gmia.hist[, Continent:= countrycode(gmia.hist[, Country],
                                                 origin = "country.name",
                                                 destination = "continent")] %>%
  .[, Code:= countrycode(gmia.hist[, Country],
                         origin = "country.name",
                         destination = "un")] %>%
  .[, Country:= countrycode(gmia.hist[, Code],
                            origin = "un",
                            destination = "country.name")] %>%
  .[!Continent == "Oceania"]

## Warning in countrycode(gmia.hist[, Country], origin = "country.name", destination = "continent")
## Warning in countrycode(gmia.hist[, Country], origin = "country.name", destination = "un"):

# Create list with the columns selected

dt.tmp <- lapply(c("AEI_1970", paste("AEI_", c(seq(1980, 2000, 10), 2005), sep = "")), function(x)
  gmia.hist[, .SD, .SDcols = c("Country", "Code", "Continent", x)])
names(dt.tmp) <- selected.years

# MERGE GHM AND HISTORIC FAO-GMIA -----
all.ghm <- c(huang.dt, isimip.dt)
ghm.dt <- lapply(names(all.ghm), function(x)
  lapply(selected.years, function(y)

```

```

all.ghm[[x]][[y]][, merge(.SD, dt.tmp[[y]], by = c("Country", "Code", "Continent"),
                           all.y = TRUE)]) %>%
lapply(., function(x) lapply(x, function(y) setnames(y, 5, "AEI")) %>%
na.omit())))

```

names(ghm.dt) <- c(names.huang, names.isimip)

```

for(i in names(ghm.dt)) {
  names(ghm.dt[[i]]) <- selected.years
}

```

## 2.4 Long-term trends between irrigated areas and irrigation water withdrawals

```

# PLOT GHM WATER WITHDRAWALS THROUGH TIME (1970-2005) ----

tmp <- lapply(ghm.dt, function(x) rbindlist(x, idcol = "Year"))

# Plot
gg <- list()
for(i in 1:length(tmp)) {
  gg[[i]] <- ggplot(tmp[[i]], aes(AEI, Water.Withdrawn,
                                    color = Continent)) +
    geom_point(size = 0.4) +
    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))) +
    labs(x = "Irrigated area (ha)",
         y = expression(paste("Water withdrawal ", " ", "(", 10^9, m^3/year, "", ")")))
    facet_wrap(~Year) +
    theme_AP() +
    theme(legend.position = "top",
          strip.background = element_rect(fill = "white")) +
    ggtitle(names(tmp[i]))
}

gg

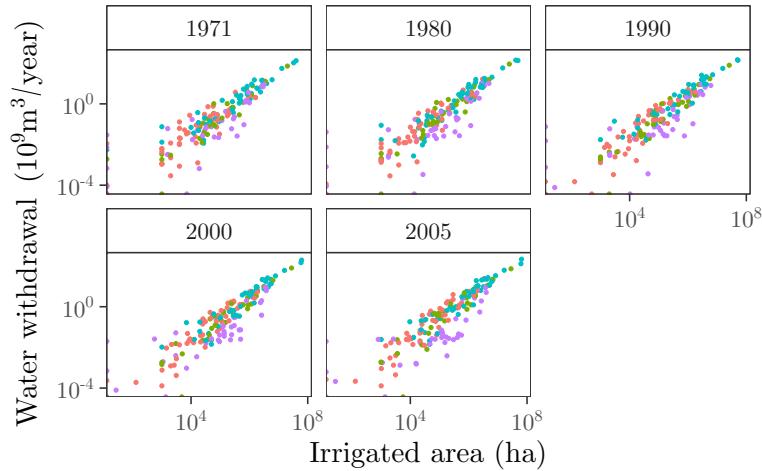
## [[1]]

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

```

## LPJmL

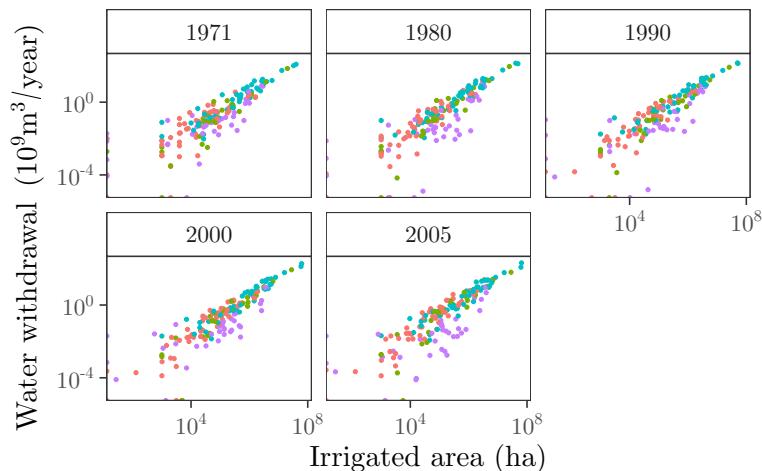
Continent • Africa • Americas • Asia • Europe



```
##  
## [[2]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

## PCR-GLOBWB

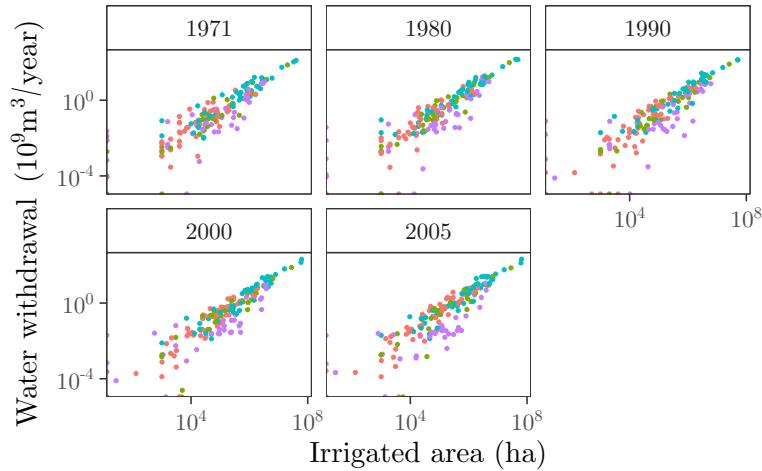
Continent • Africa • Americas • Asia • Europe



```
##  
## [[3]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

H08

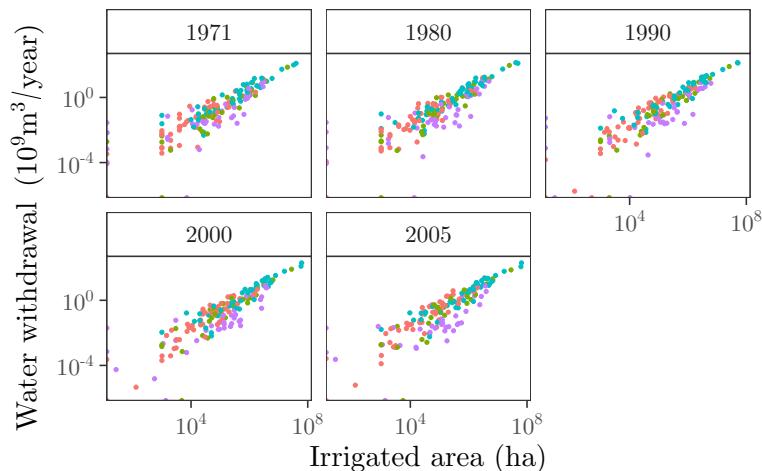
Continent • Africa • Americas • Asia • Europe



```
##  
## [[4]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

WaterGap

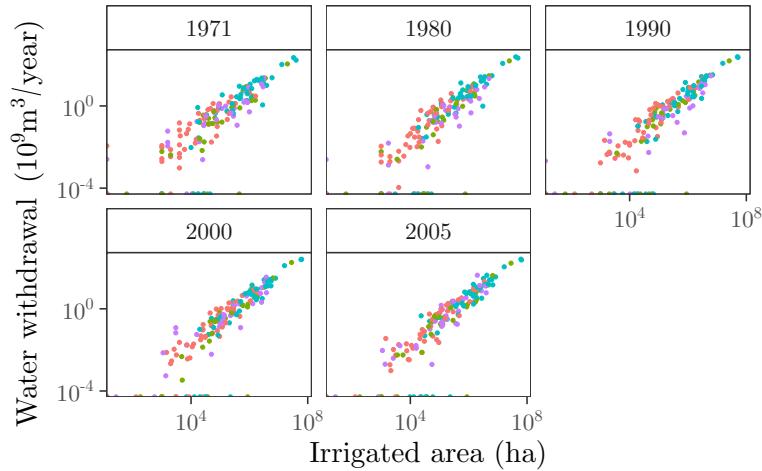
Continent • Africa • Americas • Asia • Europe



```
##  
## [[5]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

## DBHM

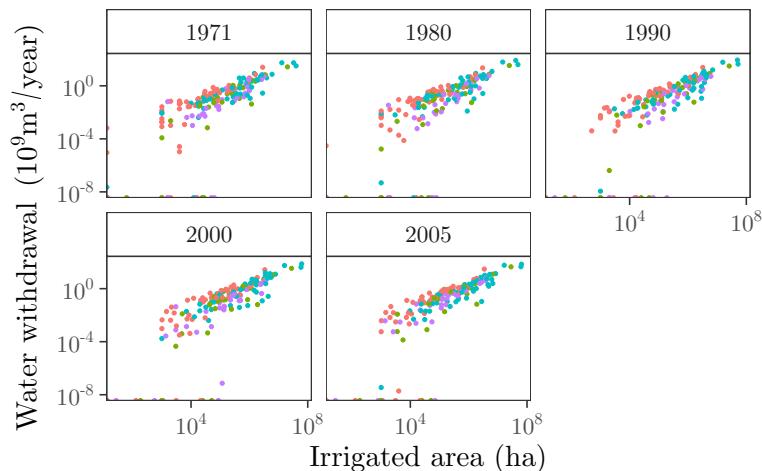
Continent • Africa • Americas • Asia • Europe



```
##  
## [[6]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

## MPI-HM

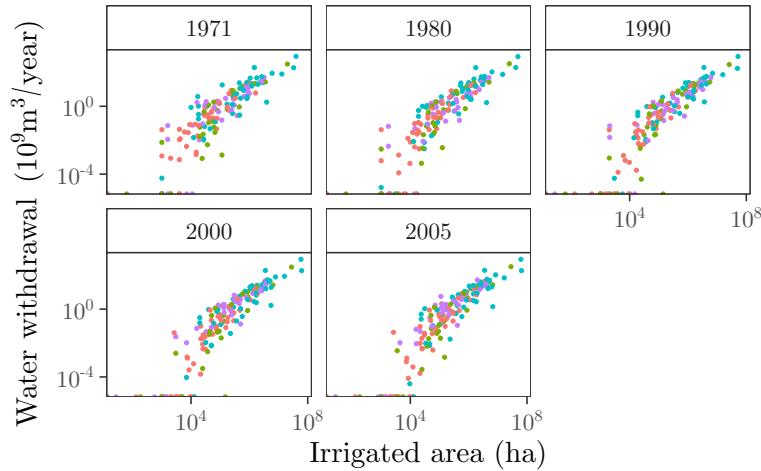
Continent • Africa • Americas • Asia • Europe



```
##  
## [[7]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

## VIC

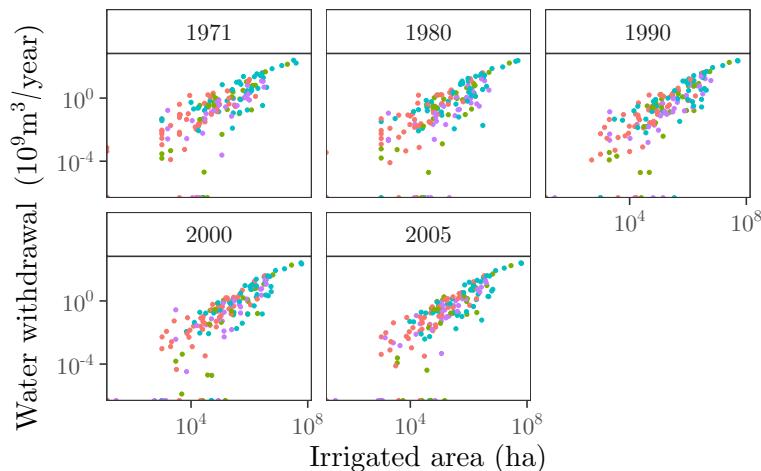
Continent • Africa • Americas • Asia • Europe



```
##  
## [[8]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```

## CLM45

Continent • Africa • Americas • Asia • Europe



```
# ANALYZE LONG-TERM HISTORICAL TRENDS -----
```

```
files <- list("pcr-globwb_gfdl-esm2m_ewembi_historical_histsoc_co2_airrrw_global_monthly_1861_2005.nc",  
               "mpi-hm_gfdl-esm2m_ewembi_historical_histsoc_co2_airrrw_global_monthly_1861_2005.nc",  
               "lpjml_miroc5_ewembi_historical_histsoc_co2_airrrw_global_monthly_1861_2005.nc",  
               "h08_miroc5_ewembi_historical_histsoc_co2_airrrw_global_monthly_1861_2005.nc")
```

```

vecs <- 1:(2005 - 1860) * 12)
vec <- split(vecs, ceiling(seq_along(vecs) / 12))
names(vec) <- 1861:2005
dname <- "airrww"
selected.years <- as.character(seq(1900, 2000, 10))

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

names.isimip.hist <- c("PCR-GLOBWB", "MPI-HM", "LPJmL", "H08")
names(isimip.hist) <- names.isimip.hist

for(i in names(isimip.hist)) {
  names(isimip.hist[[i]]) <- selected.years
}

# Create list with the columns selected

dt.tmp <- lapply(paste("AEI_", seq(1900, 2000, 10), sep = ""), function(x)
  gmia.hist[, .SD, .SDcols = c("Country", "Code", "Continent", x)])
names(dt.tmp) <- selected.years

out <- lapply(names(isimip.hist), function(x)
  lapply(selected.years, function(y)
    isimip.hist[[x]][[y]][, merge(.SD, dt.tmp[[y]], by = c("Country", "Code", "Continent"),
      all.y = TRUE))]) %>%
  lapply(., function(x) lapply(x, function(y) setnames(y, 5, "AEI")) %>%
    na.omit())))
  }

names(out) <- c("PCR-GLOBWB", "MPI-HM", "LPJmL", "H08")

for(i in names(out)) {
  names(out[[i]]) <- selected.years
}

# PLOT GHM WATER WITHDRAWALS THROUGH TIME (1900-1960) ----

tmp <- lapply(out, function(x) rbindlist(x, idcol = "Year")) %>%
  na.omit()

# Plot
gg <- list()
for(i in 1:length(tmp)) {
  gg[[i]] <- ggplot(tmp[[i]], aes(AEI, Water.Withdrawn,
    color = Continent)) +

```

```

geom_point(size = 0.4) +
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))) +
labs(x = "Irrigated area (ha)",
      y = expression(paste("Water withdrawal ", " ", "(\"", 10^9, m^3/year, "\"", " "))) +
facet_wrap(~Year, ncol = 5) +
theme_AP() +
theme(legend.position = "top",
      strip.background = element_rect(fill = "white")) +
ggtitle(names(tmp[i]))
}

gg

```

```

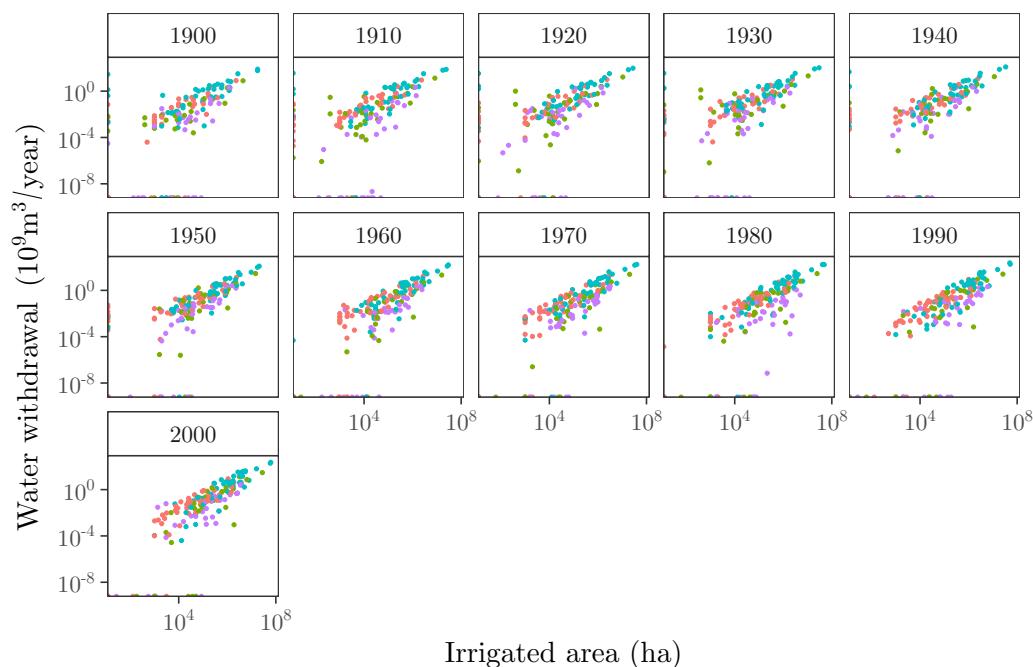
## [[1]]

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

```

### PCR-GLOBWB

Continent • Africa • Americas • Asia • Europe



```

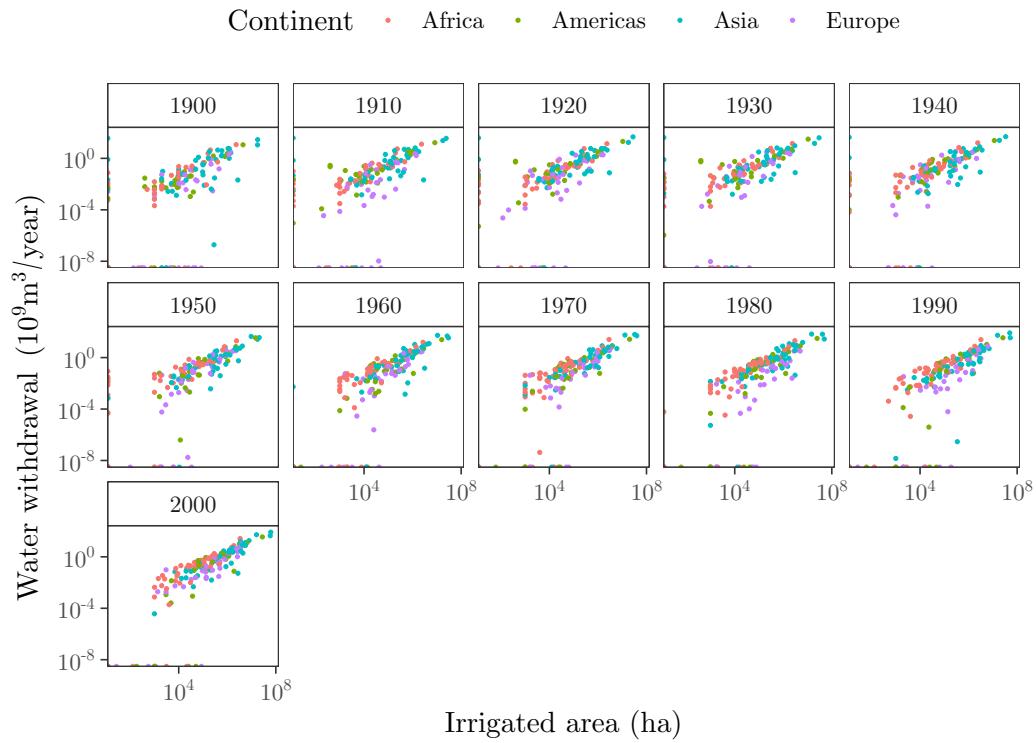
##
## [[2]]

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

```

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

## MPI-HM

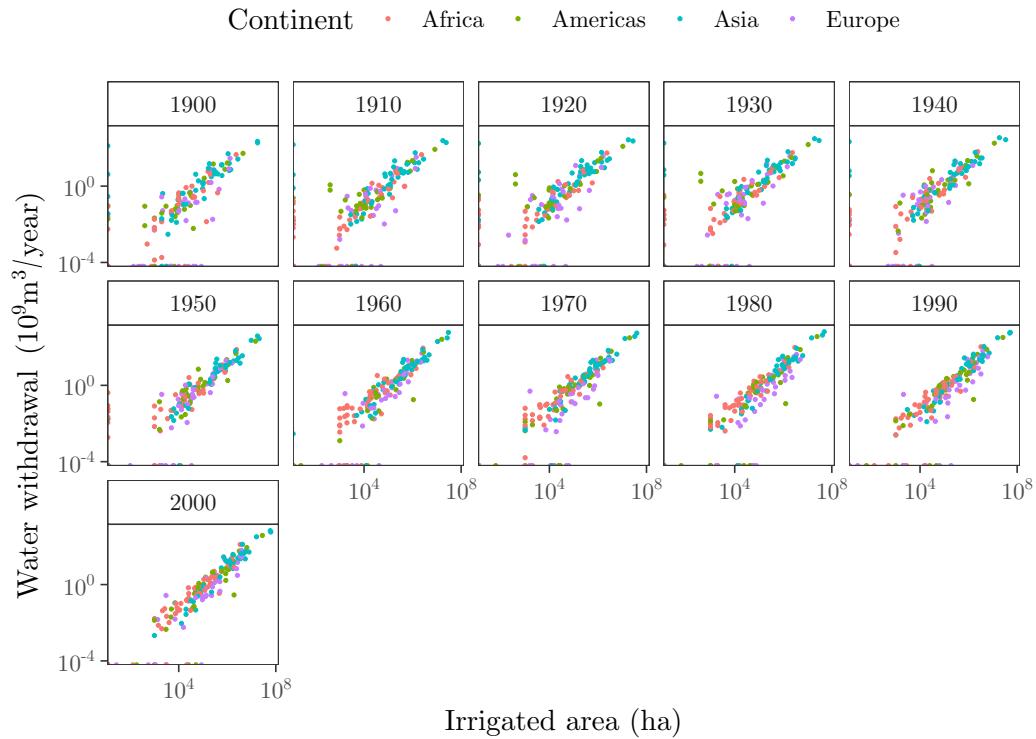


```
##
```

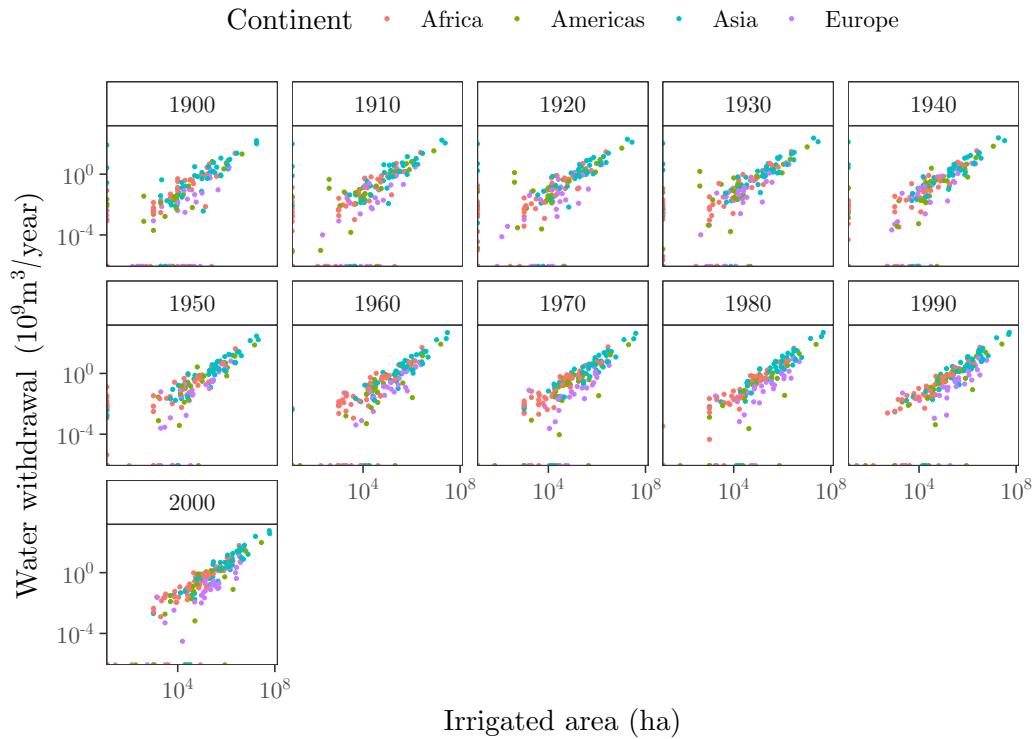
```
## [[3]]
```

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

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



```
##  
## [[4]]  
  
## Warning: Transformation introduced infinite values in continuous x-axis  
  
## Warning: Transformation introduced infinite values in continuous y-axis
```



## 2.5 Plot trend between irrigated areas and irrigation water withdrawals

```
# READ IN DATASETS ON IRRIGATION WATER WITHDRAWAL AND MEIER DATASET -----
# 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] %>%
  country_code(.) %>%
  setcolororder(., 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"] %>%
  country_code(.) %>%
  setcolororder(., cols_order)

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

# MERGE WITH MEIER ET AL. FAO-GMIA -----
water.dt <- lapply(all.ghm, function(x) rbindlist(x, idcol = "Year")) %>%
  rbindlist(., idcol = "Water.Dataset") %>%
  .[Year == 2005] %>%
  .[, Year:= NULL] %>%
  setcolororder(., cols_order) %>%
  rbind(., table4.dt, liu.dt)

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

## [1] "Cyprus"                               "Palestinian Territories"
## [3] NA

# 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]

## [1] NA

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:        LPJmL       Grenada
## 2:        LPJmL        Malta

```

## 3:	LPJmL	Saint Lucia
## 4:	LPJmL	Seychelles
## 5:	LPJmL	Swaziland
## 6:	PCR-GLOBWB	Argentina
## 7:	PCR-GLOBWB	Canada
## 8:	PCR-GLOBWB	Grenada
## 9:	PCR-GLOBWB	Malta
## 10:	PCR-GLOBWB	Portugal
## 11:	PCR-GLOBWB	Russia
## 12:	PCR-GLOBWB	Saint Lucia
## 13:	PCR-GLOBWB	Seychelles
## 14:	PCR-GLOBWB	Swaziland
## 15:	H08	Grenada
## 16:	H08	Malta
## 17:	H08	Saint Lucia
## 18:	H08	Seychelles
## 19:	H08	Swaziland
## 20:	WaterGap	Grenada
## 21:	WaterGap	Malta
## 22:	WaterGap	Saint Lucia
## 23:	WaterGap	Seychelles
## 24:	WaterGap	Swaziland
## 25:	DBHM	Grenada
## 26:	DBHM	Malta
## 27:	DBHM	Mauritius
## 28:	DBHM	Saint Lucia
## 29:	DBHM	Seychelles
## 30:	DBHM	Swaziland
## 31:	MPI-HM	Grenada
## 32:	MPI-HM	Malta
## 33:	MPI-HM	Saint Lucia
## 34:	MPI-HM	Seychelles
## 35:	MPI-HM	Swaziland
## 36:	VIC	Grenada
## 37:	VIC	Malta
## 38:	VIC	Saint Lucia
## 39:	VIC	Seychelles
## 40:	VIC	Swaziland
## 41:	CLM45	Cape Verde
## 42:	CLM45	Grenada
## 43:	CLM45	Malta
## 44:	CLM45	Saint Lucia
## 45:	CLM45	Seychelles
## 46:	CLM45	Swaziland
## 47:	Aquastat	Belize
## 48:	Aquastat	Bosnia & Herzegovina
## 49:	Aquastat	Cape Verde
## 50:	Aquastat	Croatia

```

## 51:      Aquastat          Cuba
## 52:      Aquastat        El Salvador
## 53:      Aquastat         Grenada
## 54:      Aquastat        Guyana
## 55:      Aquastat        Haiti
## 56:      Aquastat       Ireland
## 57:      Aquastat       Lebanon
## 58:      Aquastat Luxembourg
## 59:      Aquastat      Malaysia
## 60:      Aquastat       Panama
## 61:      Aquastat        Peru
## 62:      Aquastat Saint Lucia
## 63:      Aquastat       Sudan
## 64:      Aquastat Suriname
## 65:      Aquastat   Swaziland
## 66:      Aquastat Trinidad & Tobago
## 67:      Aquastat       Uruguay
## 68: Liu et al. 2016      Saint Lucia
## 69: Liu et al. 2016      Swaziland
##      Water.Dataset       V1

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

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

# 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

```

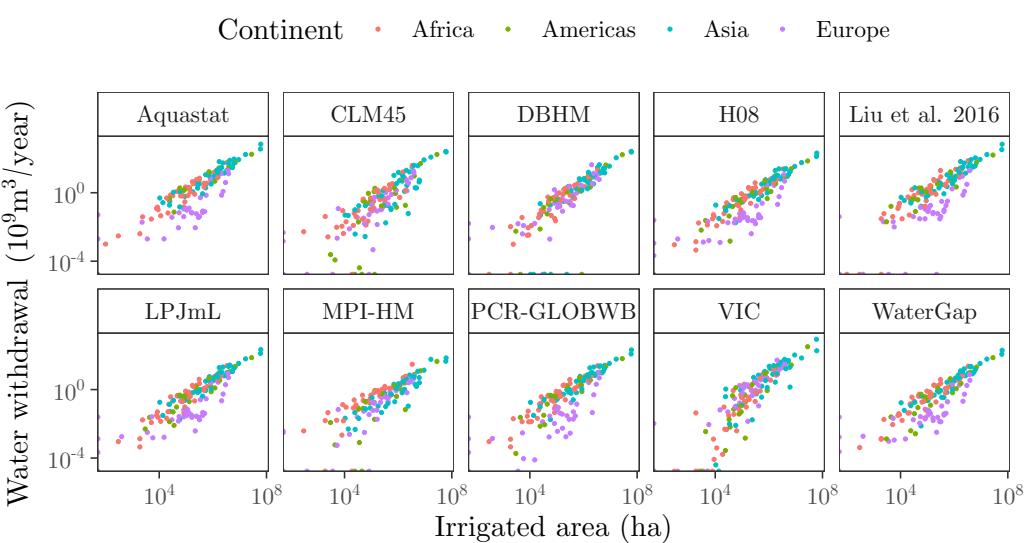


Figure 1: Scatterplots of irrigated areas reported by FAO-GMIA against irrigation water withdrawals. Each dot is a country.

### 3 The influence of other parameters

#### 3.1 Potential evapotranspiration

```

# 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() %>%

```

```

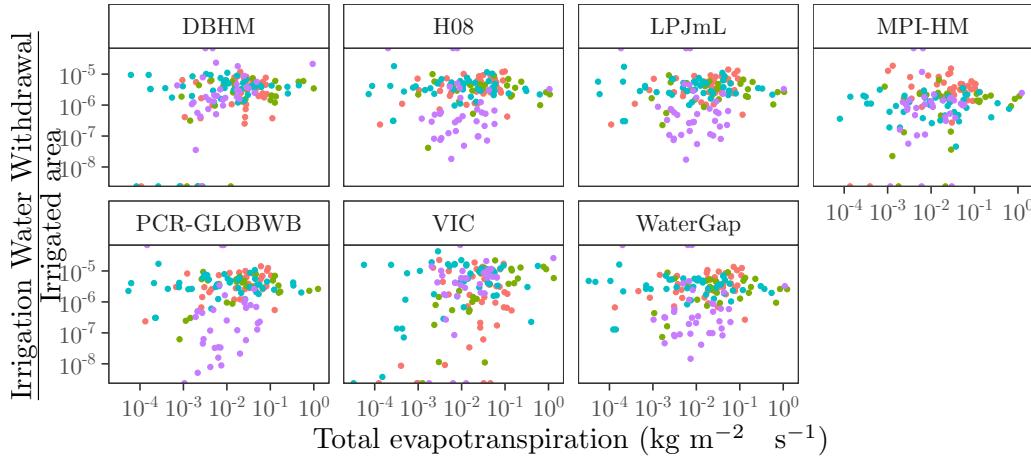
merge(full.dt[Water.Dataset %in% gm_list], ., by = c("Continent", "Country",
                                                    "Code", "Water.Dataset")) %>%
  ggplot(., aes(totlevap, div,
                color = Continent)) +
  geom_point(size = 0.6) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  labs(x = "Total evapotranspiration (kg $\\text{m}^{-2} \\text{quad} \\text{m}^{-1})",
       y = "$\\frac{\\text{Irrigation Water Withdrawal}}{\\text{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).

Continent • Africa • Americas • Asia • Europe



### 3.2 Potential evaporation

```

# 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(totlevap, 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 \text{ m}^{-2} \quad \text{s}^{-1})",
       y = "$\\frac{\\text{Irrigation Water Withdrawal}}{\\text{Irrigated area}}$") +
  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).

Continent • Africa • Americas • Asia • Europe


```

### 3.3 Water efficiency

```

# 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") %>%

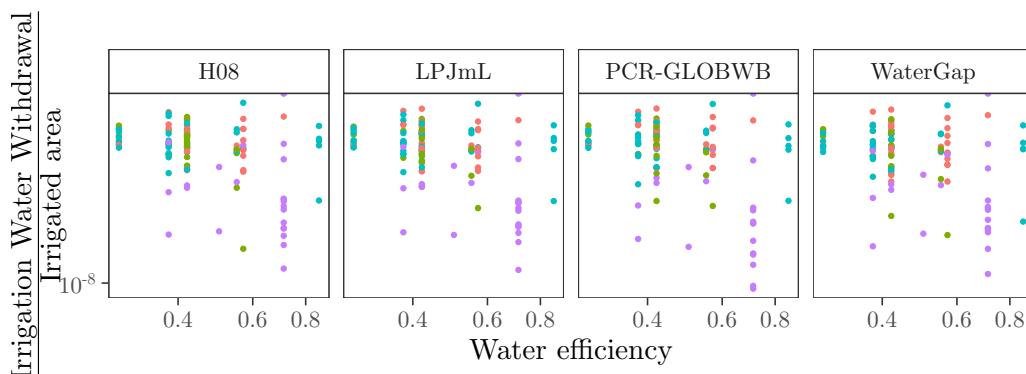
```

```

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 = "$\\frac{\\text{Irrigation Water Withdrawal}}{\\text{Irrigated area}}$") +
  facet_wrap(~Water.Dataset, ncol = 4) +
  theme_AP() +
  theme(legend.position = "top",
        strip.background = element_rect(fill = "white"))

```

Continent • Africa • Americas • Asia • Europe



## 4 Creation of the final dataset

```

# 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))

```

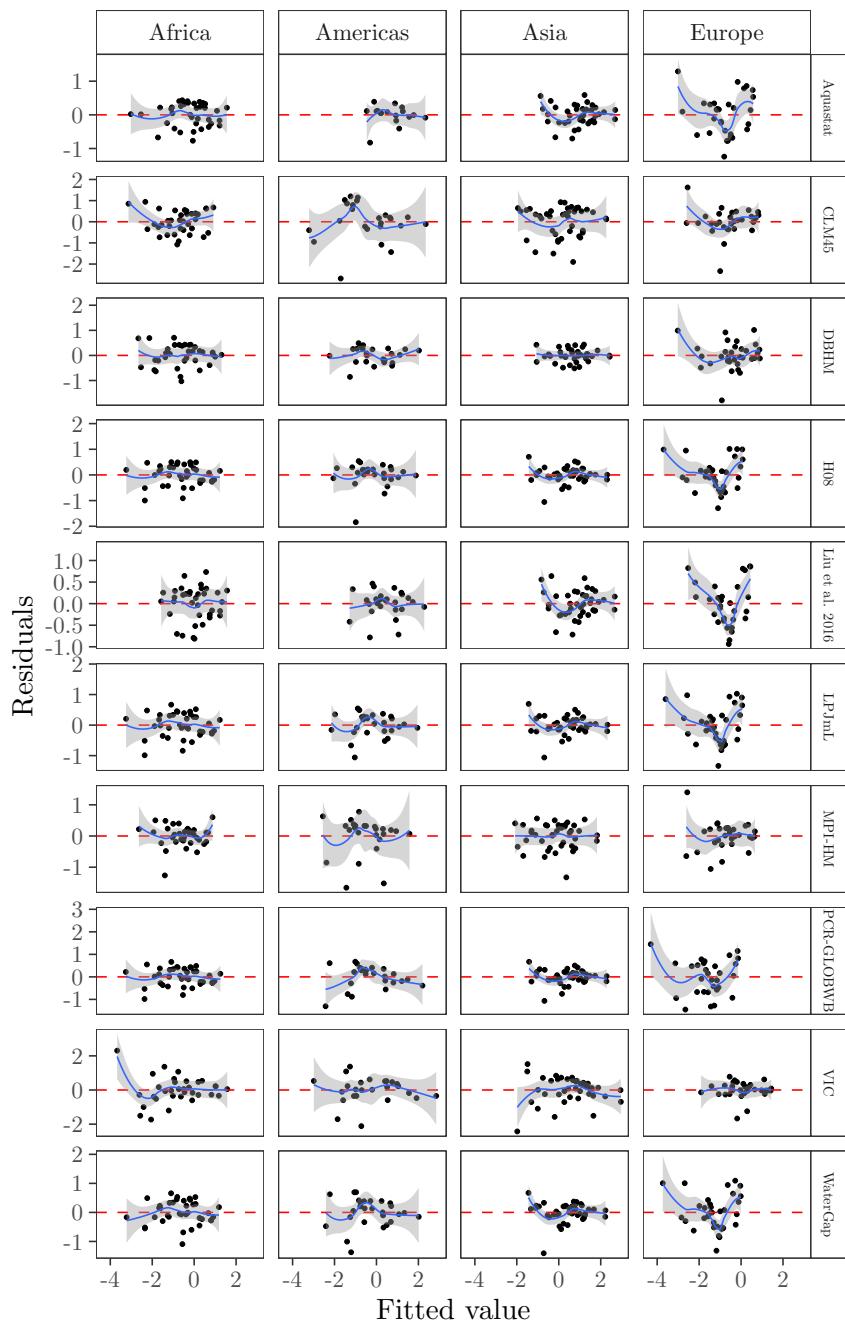
```

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

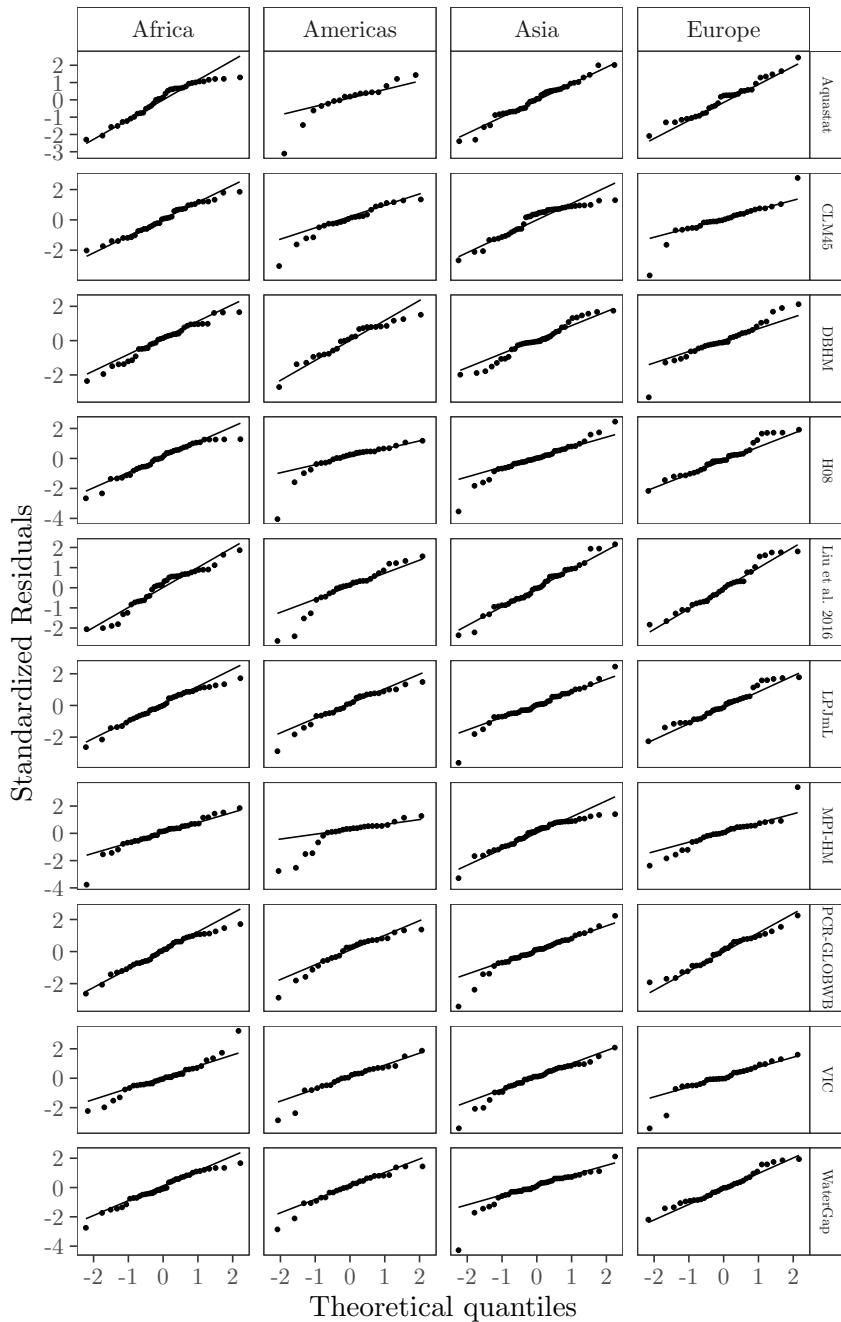
# Residuals versus fitted
ggplot(diagnostics.dt, aes(.fitted, .resid)) +
  geom_point(size = 0.5) +
  geom_hline(yintercept = 0, lty = 2, color = "red") +
  geom_smooth(size = 0.5) +
  labs(x = "Fitted value", y = "Residuals") +
  facet_grid(Water.Dataset ~ Continent,
             scales = "free_y")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



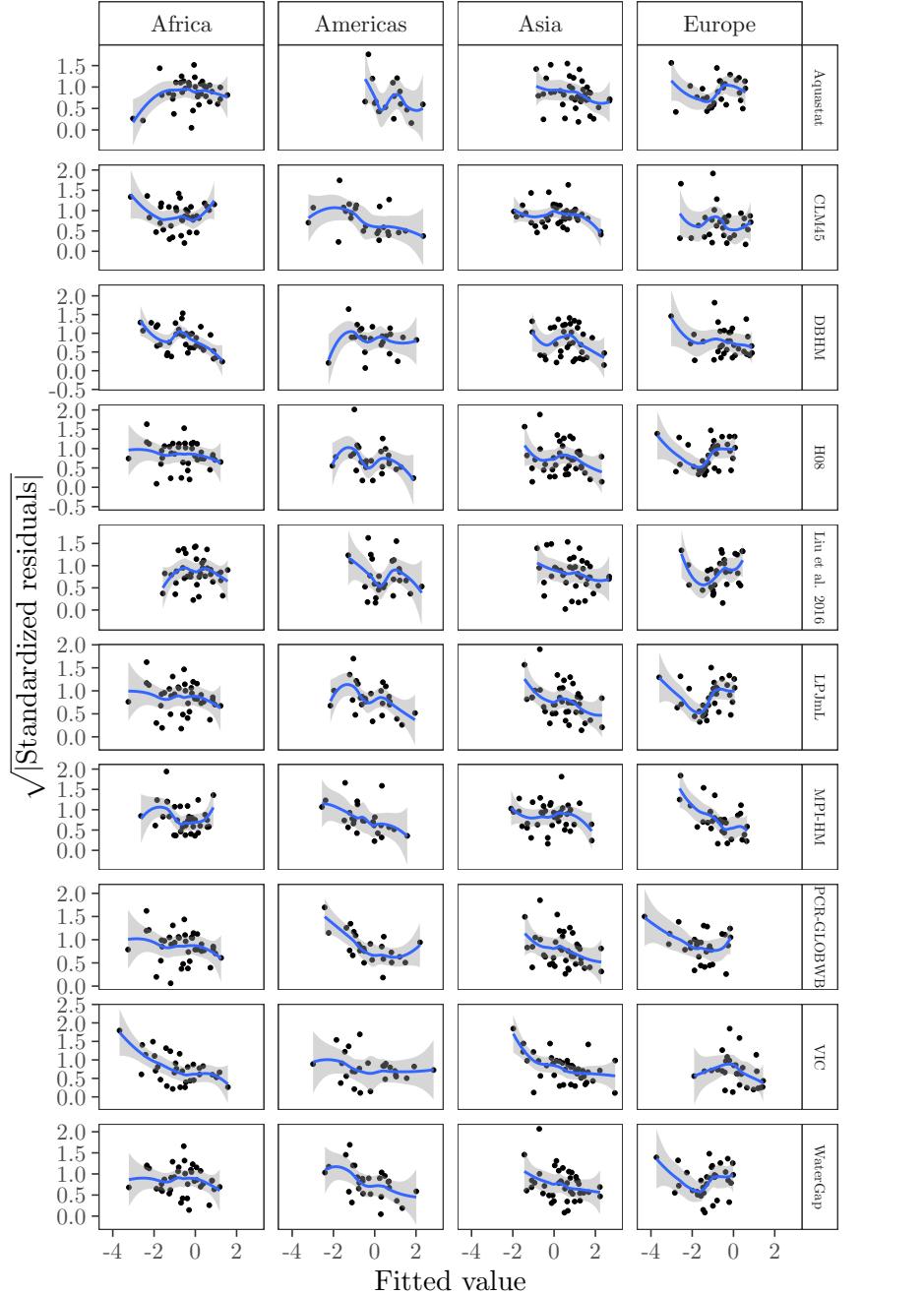
```
# Scale location plot
ggplot(diagnostics.dt, aes(.fitted, sqrt(abs(.std.resid)))) +
  geom_point(size = 0.5) +
  geom_smooth() +
  facet_grid(Water.Dataset ~ Continent,
             scales = "free_y") +
  labs(x = "Fitted value",
       y = "$\\sqrt{|\\text{Standardized residuals}|}$") +
  theme_AP()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

```

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

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

```



```

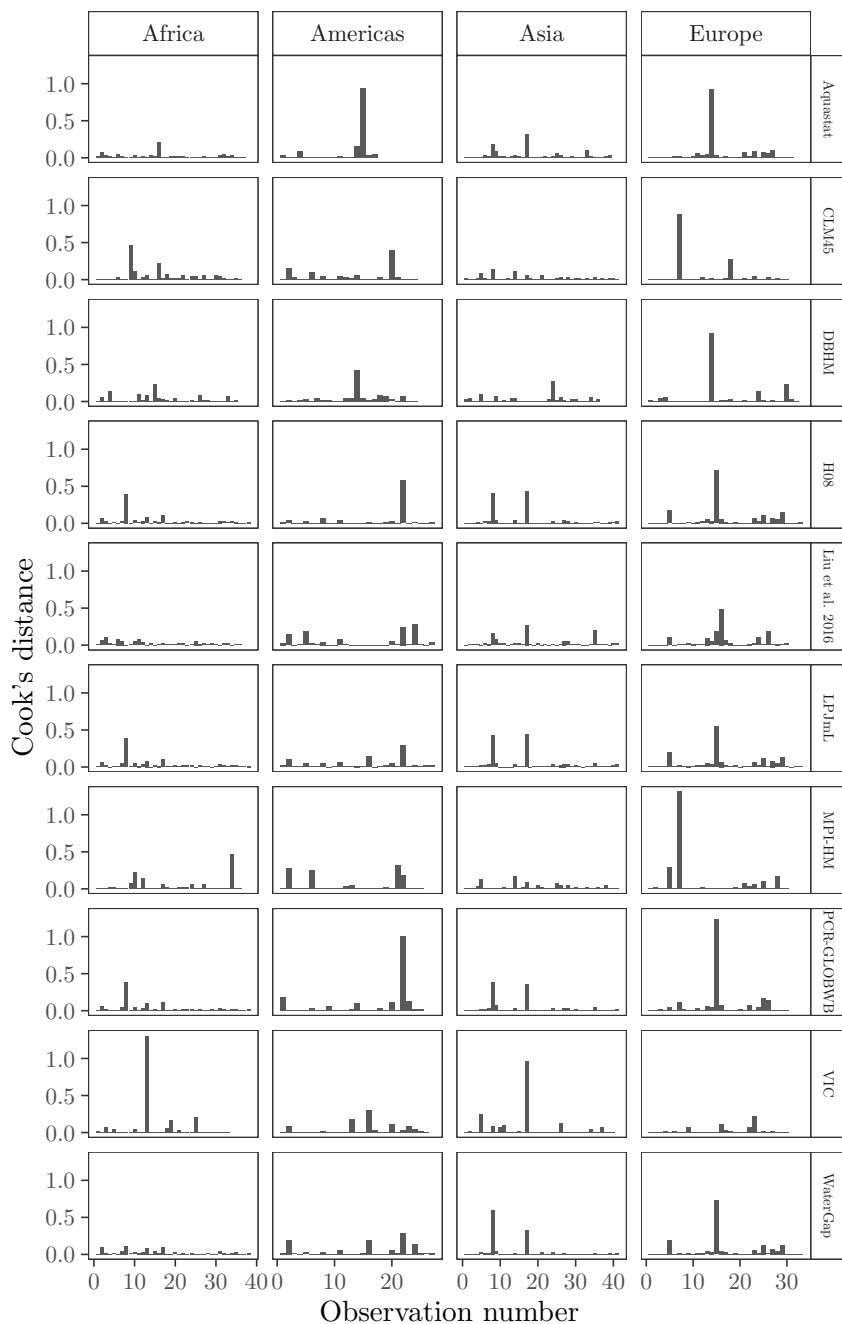
# Cook's distance
ID <- diagnostics.dt[, .N, .(Continent, Water.Dataset)]
diagnostics.dt <- diagnostics.dt[, ID:= unlist(sapply(ID[, N], function(x) 1:x))]
ggplot(diagnostics.dt, aes(ID, .cooksdf)) +
  geom_col() +
  facet_grid(Water.Dataset ~ Continent,

```

```

    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")

```

## 5 Missing values

```
# 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")
```

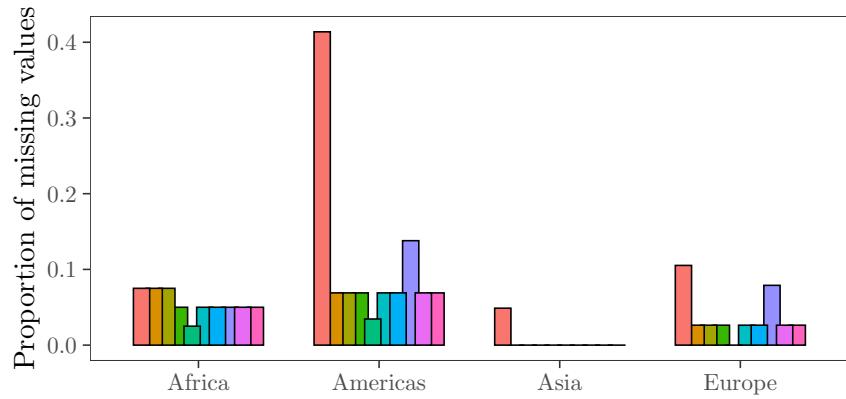
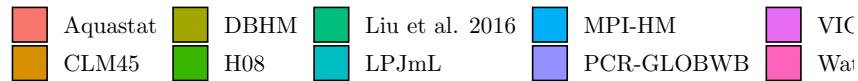


Figure 2: Proportion of missing values in irrigation water withdrawal per dataset.

### 5.1 Imputation of missing values

```
# 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")

# Run
full.dt <- full.dt[, , div:= NULL]

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)]

```

## 6 Compute linear regressions

```

# 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))

# 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"]

```

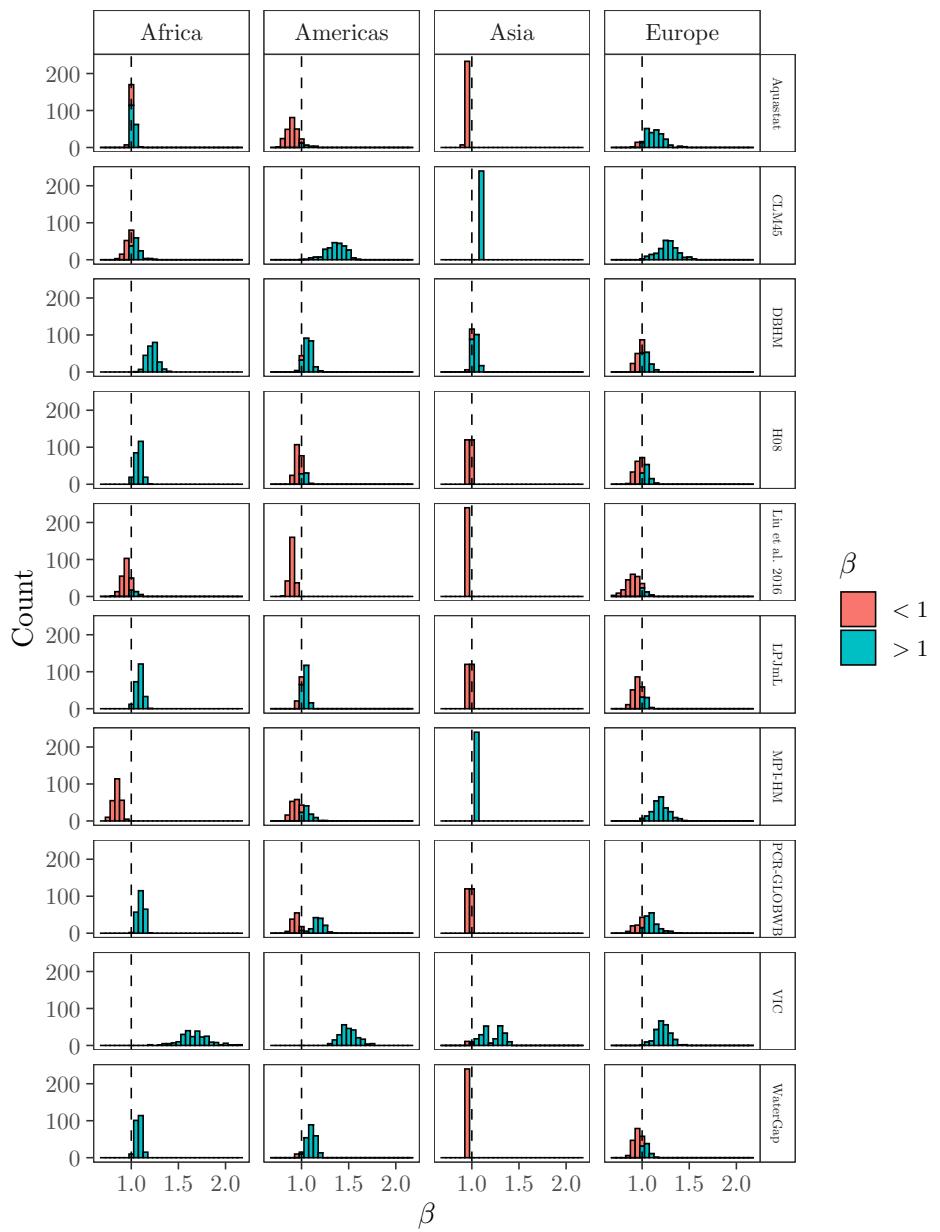
```

# 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`.

```



## 7 Predictions

### 7.1 Compare our approach with estimates by GHM

```
# 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")]

# 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:	LPJmL	Africa	3	0.07894737
## 2:	PCR-GLOBWB	Africa	2	0.05263158
## 3:	H08	Africa	2	0.05263158
## 4:	WaterGap	Africa	4	0.10526316
## 5:	DBHM	Africa	5	0.13513514
## 6:	MPI-HM	Africa	5	0.13157895
## 7:	VIC	Africa	17	0.44736842
## 8:	CLM45	Africa	8	0.21621622
## 9:	Aquastat	Africa	13	0.35135135
## 10:	Liu et al. 2016	Africa	18	0.46153846
## 11:	LPJmL	Americas	2	0.07692308

Water dataset    Aquastat    DBHM    Liu et al. 2016    MPI-HM    PCR-GLOBWB    VIC  
 CLM45    H08    LPJmL    WaterGap

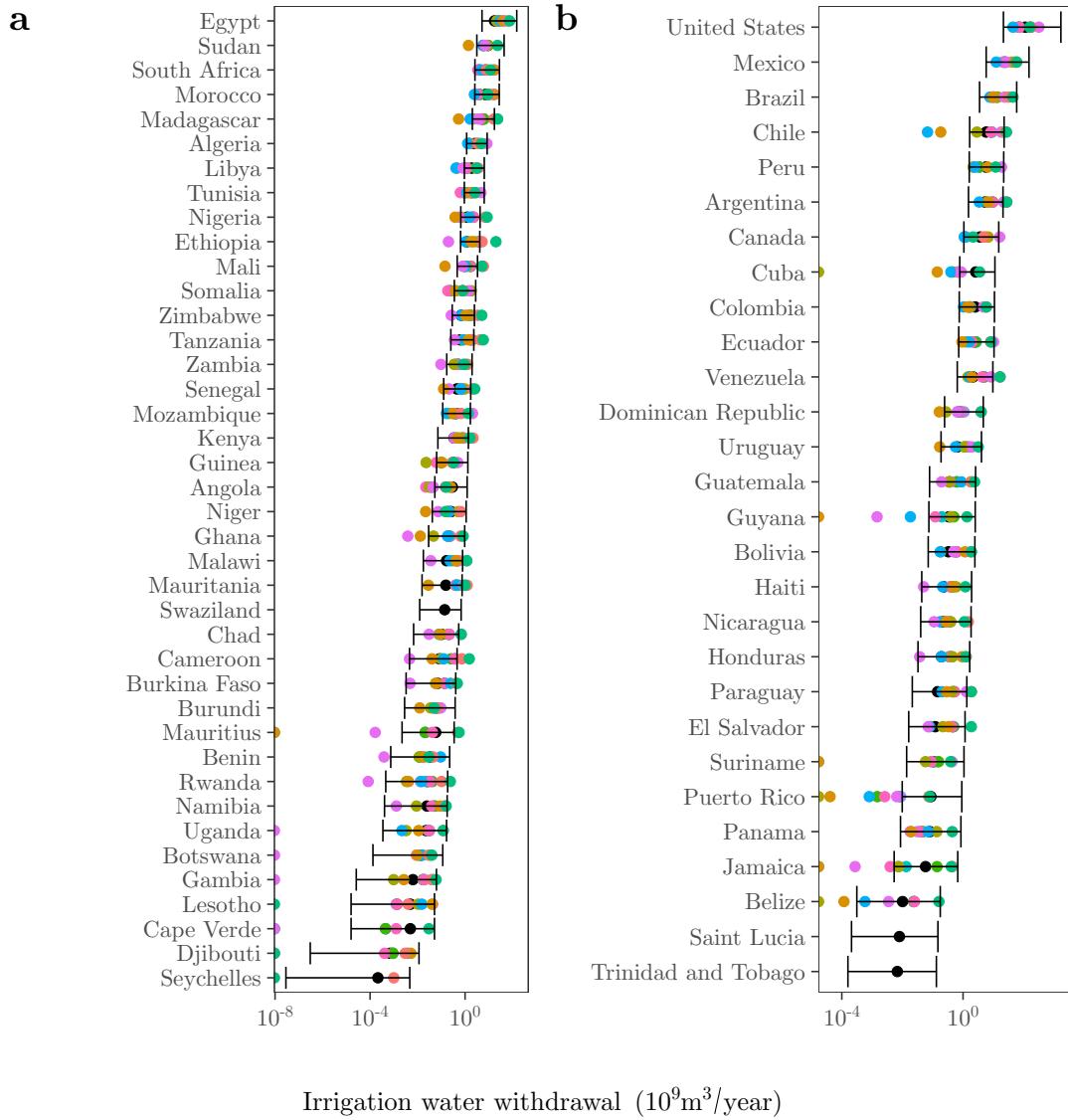


Figure 3: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models (DBHM, Ho8, LPJmL, MPI-HM, PCR-GLOBWB,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

Water dataset Aquastat DBHM Liu et al. 2016 MPI-HM PCR-GLOBWB VIC  
 CLM45 H08 LPJmL

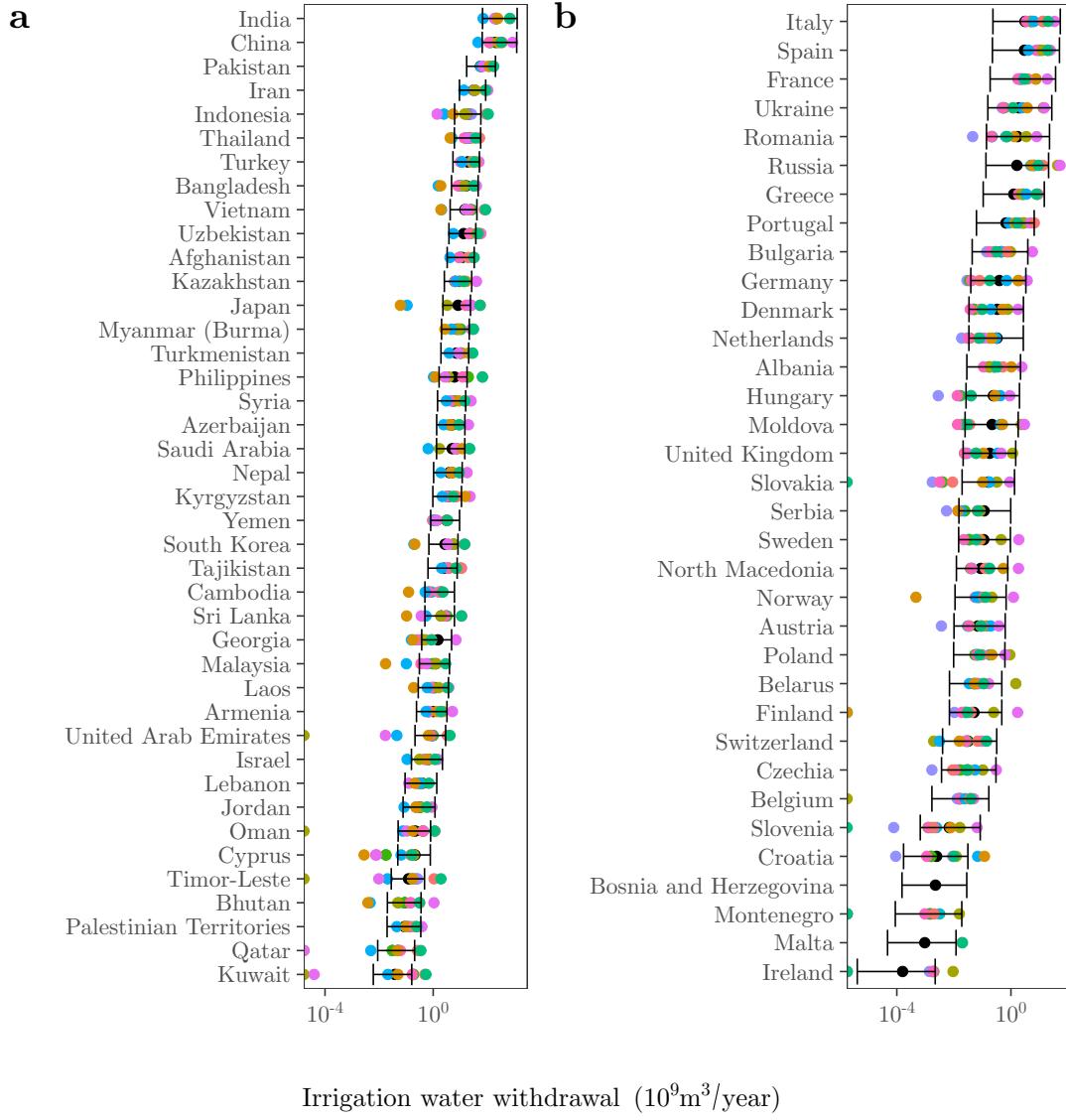


Figure 4: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models (DBHM, Ho8, LPJmL, MPI-HM, PCR-GLOBWB,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

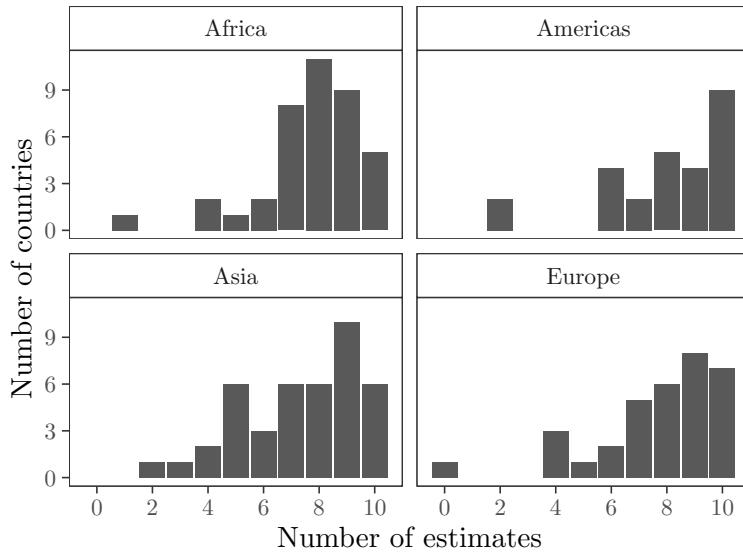
```

## 12:          H08 Americas 2 0.07692308
## 13: WaterGap Americas 3 0.11538462
## 14:        DBHM Americas 3 0.11538462
## 15:      MPI-HM Americas 6 0.23076923
## 16:        VIC Americas 5 0.19230769
## 17:      CLM45 Americas 9 0.34615385
## 18:    Aquastat Americas 4 0.23529412
## 19: Liu et al. 2016 Americas 5 0.19230769
## 20: PCR-GLOBWB Americas 2 0.08333333
## 21:      LPJmL Asia 3 0.07317073
## 22: PCR-GLOBWB Asia 4 0.09756098
## 23:          H08 Asia 4 0.09756098
## 24: WaterGap Asia 3 0.07317073
## 25:        DBHM Asia 6 0.14634146
## 26:      MPI-HM Asia 16 0.39024390
## 27:        VIC Asia 22 0.53658537
## 28:      CLM45 Asia 17 0.41463415
## 29:    Aquastat Asia 16 0.41025641
## 30: Liu et al. 2016 Asia 16 0.39024390
## 31:      LPJmL Europe 4 0.12500000
## 32: PCR-GLOBWB Europe 12 0.40000000
## 33:          H08 Europe 4 0.12500000
## 34: WaterGap Europe 4 0.12500000
## 35:        DBHM Europe 7 0.21875000
## 36:      MPI-HM Europe 5 0.15625000
## 37:        VIC Europe 14 0.43750000
## 38:      CLM45 Europe 6 0.18750000
## 39:    Aquastat Europe 3 0.09677419
## 40: Liu et al. 2016 Europe 5 0.15151515
##       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      107  0.7697842
##
## [[2]]
##   Total.countries Bounded Proportion
## 1:           139       27  0.1942446
# 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: Puerto Rico    Americas  2
## 3:    Kuwait     Asia  2
##
## [[4]]

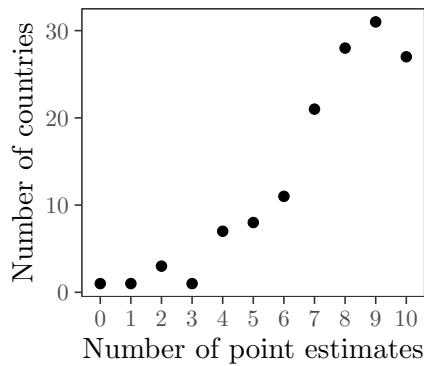
```

```

##          Country Continent V1
## 1: Philippines      Asia  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()

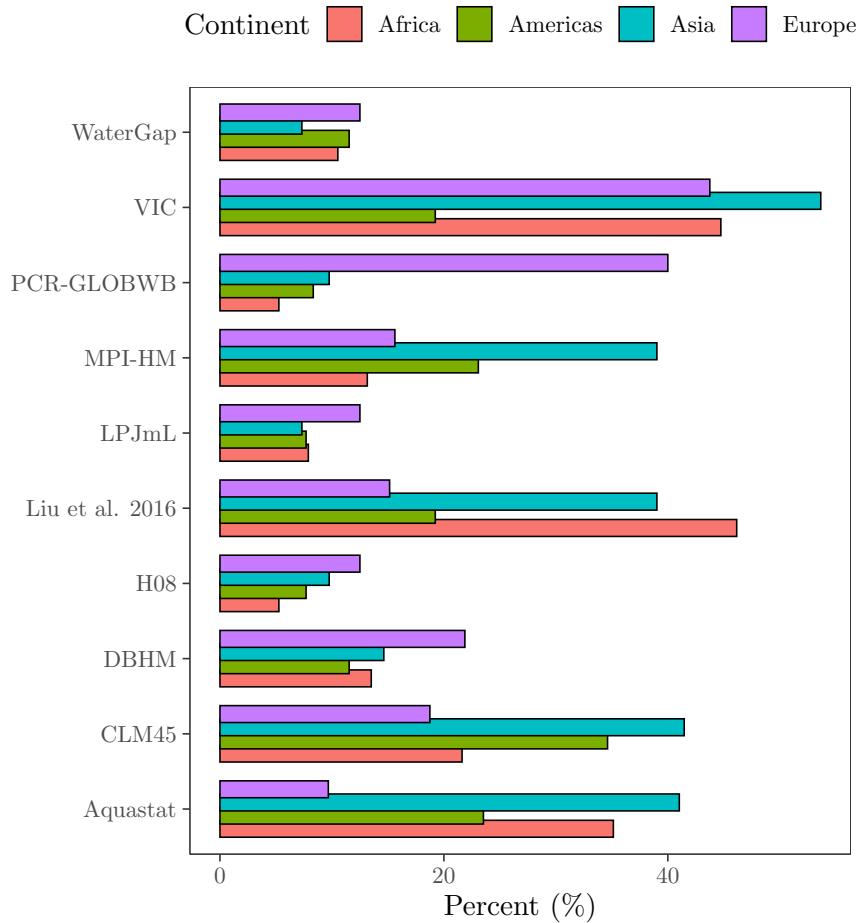


```

# 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")



## 7.2 Can future irrigation water withdrawals be estimated as a function of irrigated areas?

```
# FUTURE IRRIGATION WATER WITHDRAWALS -----
files <- list(
  "pcr-globwb_miroc5_ewembi_rcp60_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "pcr-globwb_miroc5_ewembi_rcp26_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp60_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp26_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp60_rcp60soc_co2_airrrw_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp26_rcp26soc_co2_airrrw_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp85_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp60_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp26_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp60_rcp60soc_co2_airrrw_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp85_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp26_rcp26soc_co2_airrrw_global_monthly_2006_2099.nc",
  "mpi-hm_miroc5_ewembi_rcp60_2005soc_co2_airrrw_global_monthly_2006_2099.nc",
  "mpi-hm_miroc5_ewembi_rcp26_2005soc_co2_airrrw_global_monthly_2006_2099.nc"
)
```

```

vecs <- 1:(2099 - 2005) * 12)
vec <- split(vecs, ceiling(seq_along(vecs) / 12))
names(vec) <- 2006:2099
dname <- "airrww"
selected.years <- as.character(seq(2030, 2050, 10))

# Read in datasets
isimip.future <- mclapply(
  files, function(x)
    open_nc_files(file = x, dname = dname, selected.years = selected.years,
                  vec = vec), mc.cores = detectCores() * 0.75
)

# PLOT -----
# Create vector to name the slots
ghms <- toupper(unlist(lapply(files, function(x) sub("\\_.*", "", x))))
pattern <- "ewembi_(.*?)soc"
climate <- unlist(lapply(files, function(x) str_match(x, pattern)[, 2]))
names.isimip.future <- gsub("_", ".", paste(ghms, climate, sep = "/"))

# Name the slots
names(isimip.future) <- names.isimip.future

for(i in names(isimip.future)) {
  names(isimip.future[[i]]) <- selected.years
}

# Arrange data
isimip.future.dt <- lapply(isimip.future, function(x) rbindlist(x, idcol = "Year")) %>%
  rbindlist(., idcol = "Water.Dataset") %>%
  .[!Continent == "Oceania"] %>%
  separate(., "Water.Dataset", c("Water.Dataset", "Scenario"), "/") %>%
  na.omit()

Cont <- unique(isimip.future.dt$Continent)
water.tmp.future <- isimip.future.dt[Country %in% data.table(water.quantiles)[, Country]]

gg <- list()
for(j in Cont) {
  gg[[j]] <- water.quantiles[Continent == j] %>%
    ggplot(., aes(reorder(Country, median), median)) +
    geom_point() +
    geom_point(data = water.tmp.future[Year == "2050" & Continent == j],
               aes(reorder(Country, Water.Withdrawn), Water.Withdrawn,
                   shape = Scenario,
                   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 = "Global Model") +
labs(y = "", x = "") +
coord_flip() +
theme_AP() +
theme(legend.position = "none")
}

legend <- get_legend(gg[[1]] +
                      theme(legend.position = "top",
                            legend.box = "vertical"))

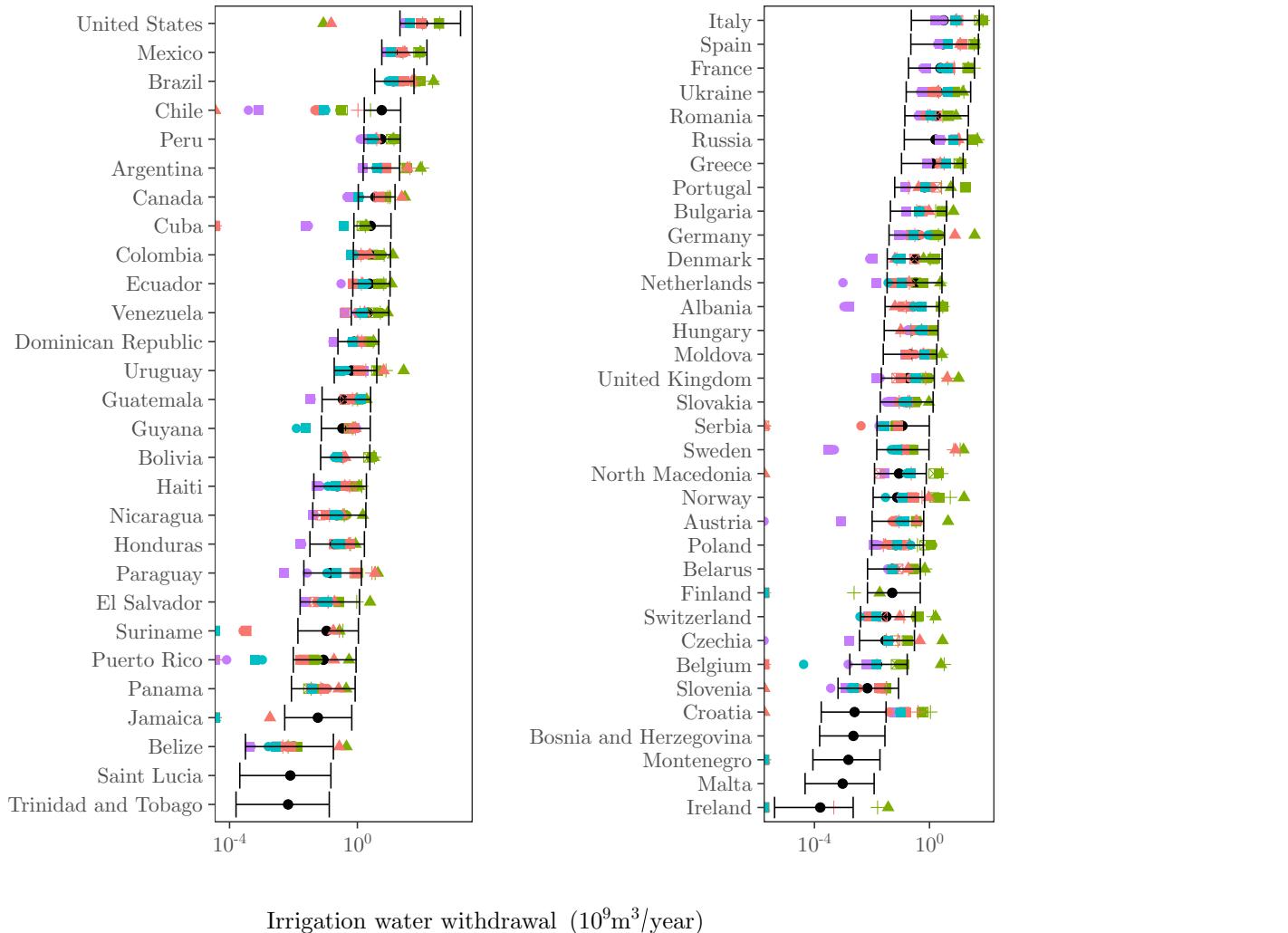
## Warning: Transformation introduced infinite values in continuous y-axis
bottom1 <- plot_grid(gg[[1]] + theme(legend.position = "none"),
                      gg[[2]] + theme(legend.position = "none"), ncol = 2)

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

## Warning: Transformation introduced infinite values in continuous y-axis
all1 <- plot_grid(legend, bottom1, rel_heights = c(0.2, 1), ncol = 1)
grid.arrange(arrangeGrob(all1, bottom = grid::textGrob(
  label = expression(paste("Irrigation water withdrawal ", " ", "(",
    10^9, "m^3/year, "" , "))))
```

Global Model ● H08 ● LPJML ● MPI-HM ● PCR-GLOBWB

Scenario ● rcp26.2005 ▲ rcp26.rcp26 ■ rcp60.2005 + rcp60.rcp60 □ rcp85.2005



```
bottom2 <- plot_grid(gg[[3]] + theme(legend.position = "none"),
                      gg[[4]] + theme(legend.position = "none"), ncol = 2)

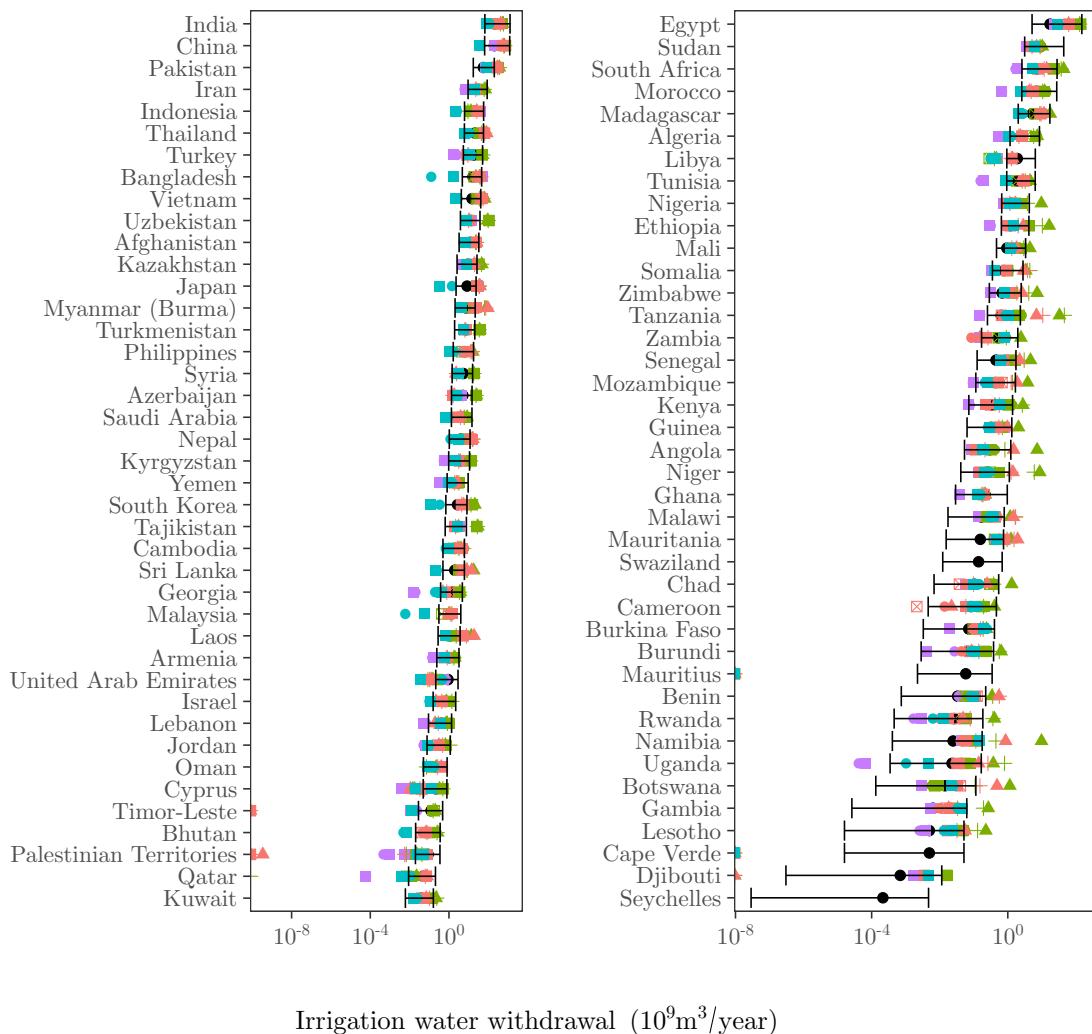
## Warning: Transformation introduced infinite values in continuous y-axis

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

all2 <- plot_grid(legend, bottom2, rel_heights = c(0.2, 1), ncol = 1)
grid.arrange(arrangeGrob(all2, bottom = grid::textGrob(
  label = expression(paste("Irrigation water withdrawal ", " ", "(", "10^9, m^3/year, "", ")")))
```

Global Model ● H08 ● LPJML ● MPI-HM ● PCR-GLOBWB

Scenario ● rcp26.2005 ▲ rcp26.rcp26 ■ rcp60.2005 + rcp60.rcp60 □ rcp85.2005



# COMPARE CURRENT WITH FUTURE PREDICTIONS -----

```
da2 <- merge(water.tmp.future, 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
fit.future <- da2[, .(fit.future = sum(fit) / .N), .(Continent, Country)]

fit.current <- da[!Water.Dataset %in% c("Aquastat", "Liu et al. 2016")] %>%
  .[, .(fit.current = sum(fit) / .N), .(Continent, Country)]
```

```

fit.all <- merge(fit.future, fit.current, by = c("Country", "Continent"))

fit.all[, .(all.good = sum(fit.future >= 0.5 & fit.current >= 0.5) / .N,
            good.current = sum(fit.future <= 0.5 & fit.current >= 0.5) / .N,
            good.future = sum(fit.future > 0.5 & fit.current < 0.5) / .N,
            all.bad = sum(fit.future < 0.5, fit.current < 0.5) / .N), Continent]

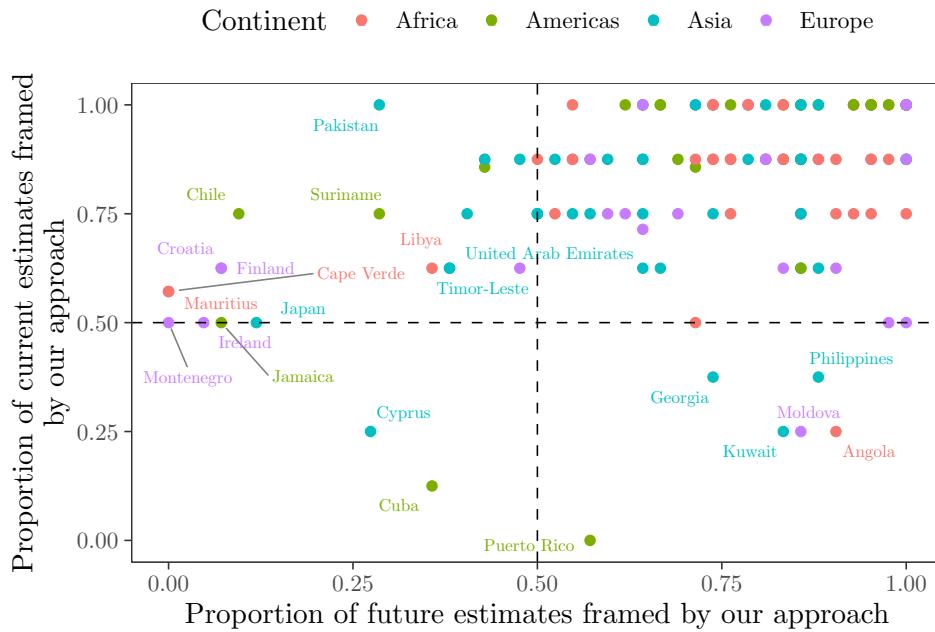
##      Continent all.good good.current good.future   all.bad
## 1:      Asia 0.7317073  0.2439024  0.07317073 0.2926829
## 2:    Europe 0.7812500  0.1875000  0.03125000 0.2187500
## 3:   Africa 0.8947368  0.1052632  0.02631579 0.1052632
## 4: Americas 0.7692308  0.1538462  0.03846154 0.2692308

fit.all[Country == "India"]

##      Country Continent fit.future fit.current
## 1:   India       Asia           1           1

ggplot(fit.all, aes(fit.future, fit.current, color = Continent)) +
  geom_text_repel(data = fit.all[fit.future < 0.4 & fit.current < 0.3 |
                                    fit.future < 0.4 | fit.current < 0.4],
                  aes(label = Country),
                  box.padding = 0.35,
                  point.padding = 0.5,
                  size = 2.3,
                  segment.color = 'grey50') +
  geom_point() +
  theme_AP() +
  geom_vline(xintercept = 0.5, lty = 2) +
  geom_hline(yintercept = 0.5, lty = 2) +
  labs(x = "Proportion of future estimates framed by our approach",
       y = "Proportion of current estimates framed \n by our approach") +
  theme(legend.position = "top")

```



## 8 The model

### 8.1 Create lookup table

```
# CREATE LOOKUP TABLE -----
lookup <- setkey(all.results, index)

# EXPORT DATASETS -----
fwrite(full.imput, "full.imput.csv")
fwrite(results, "results.csv")
fwrite(lookup, "lookup.csv")
fwrite(water.quantiles, "water.quantiles.csv")
```

### 8.2 Sample matrix

```
# 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"
```

```

# 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())

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

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

```

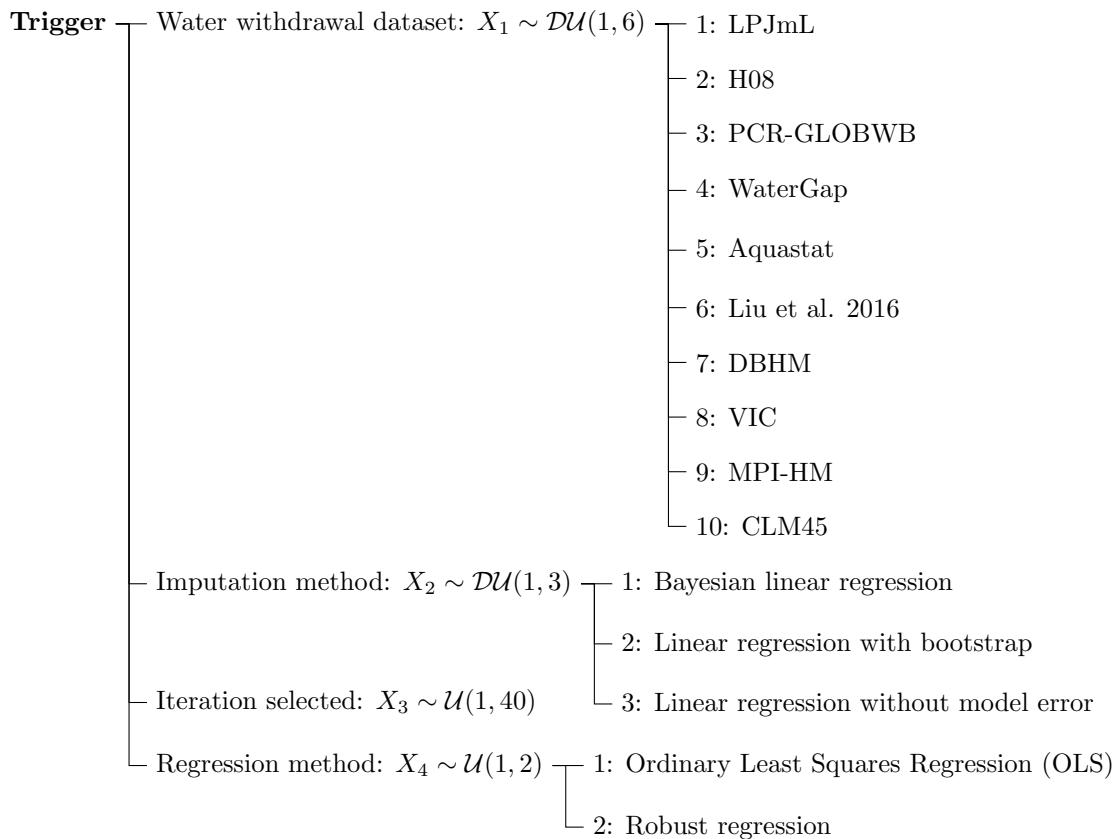


Figure 5: Tree diagram coding the discrete probability distributions of each trigger into each uncertainty level.

```

# 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")

# PRINT SAMPLE MATRIX ----

print(sample.matrix.dt)

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

```

### 8.3 Run the model

```

# THE MODEL ----

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

# RUN THE MODEL ----

# 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)

# 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")

```

## 9 Uncertainty analysis

```

# 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 0.7547564 0.8005998 0.8620280 0.8852212 0.8942089 0.9087420
## 2: Americas 0.6834208 0.7433140 0.8283374 0.8846235 0.9129680 0.9473124
## 3:     Asia 0.6774377 0.7280933 0.8642012 0.8912723 0.9185866 0.9340638
## 4:   Europe 0.5787127 0.6548570 0.7136559 0.7569199 0.8331306 0.9341776
##          q0.99     q1     mean     median
## 1: 0.9117541 0.9336117 0.8691079 0.8852212

```

```

## 2: 0.9576745 0.9810536 0.8602771 0.8846235
## 3: 0.9367093 0.9427824 0.8643384 0.8912723
## 4: 0.9414859 0.9569898 0.7679499 0.7569199

# 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 = "Count") +
  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
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")

```

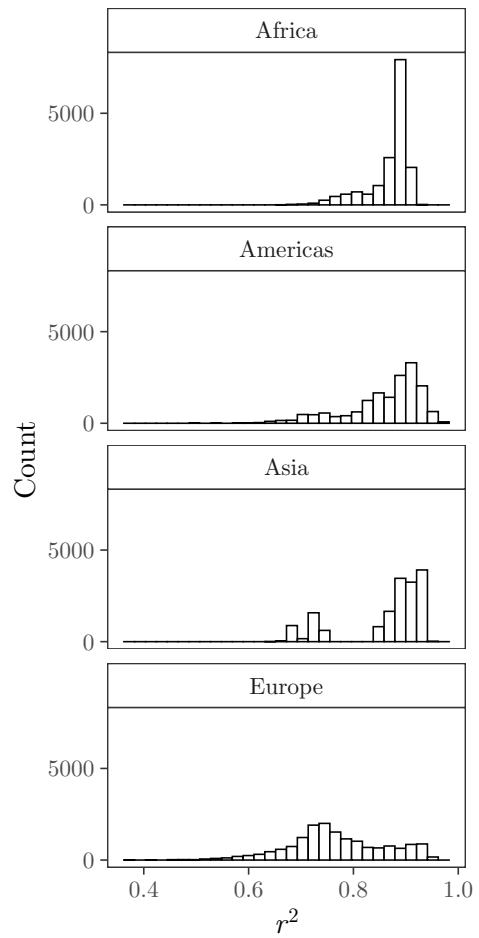


Figure 6: Uncertainty in the empirical distribution of  $r^2$ .

```

## `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()

```

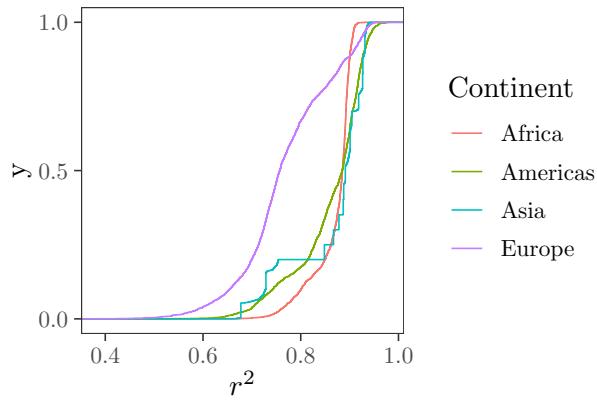


Figure 7: Cumulative empirical distribution for  $r^2$ .

## 10 Sensitivity analysis

```

# 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",
    strip.background = element_rect(fill = "white"))

# SENSITIVITY ANALYSIS ----

# Number of bootstrap replicas
R <- 10^3

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)$results,
                                             Continent]

# EXPORT SENSITIVITY INDICES ----

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"))

```

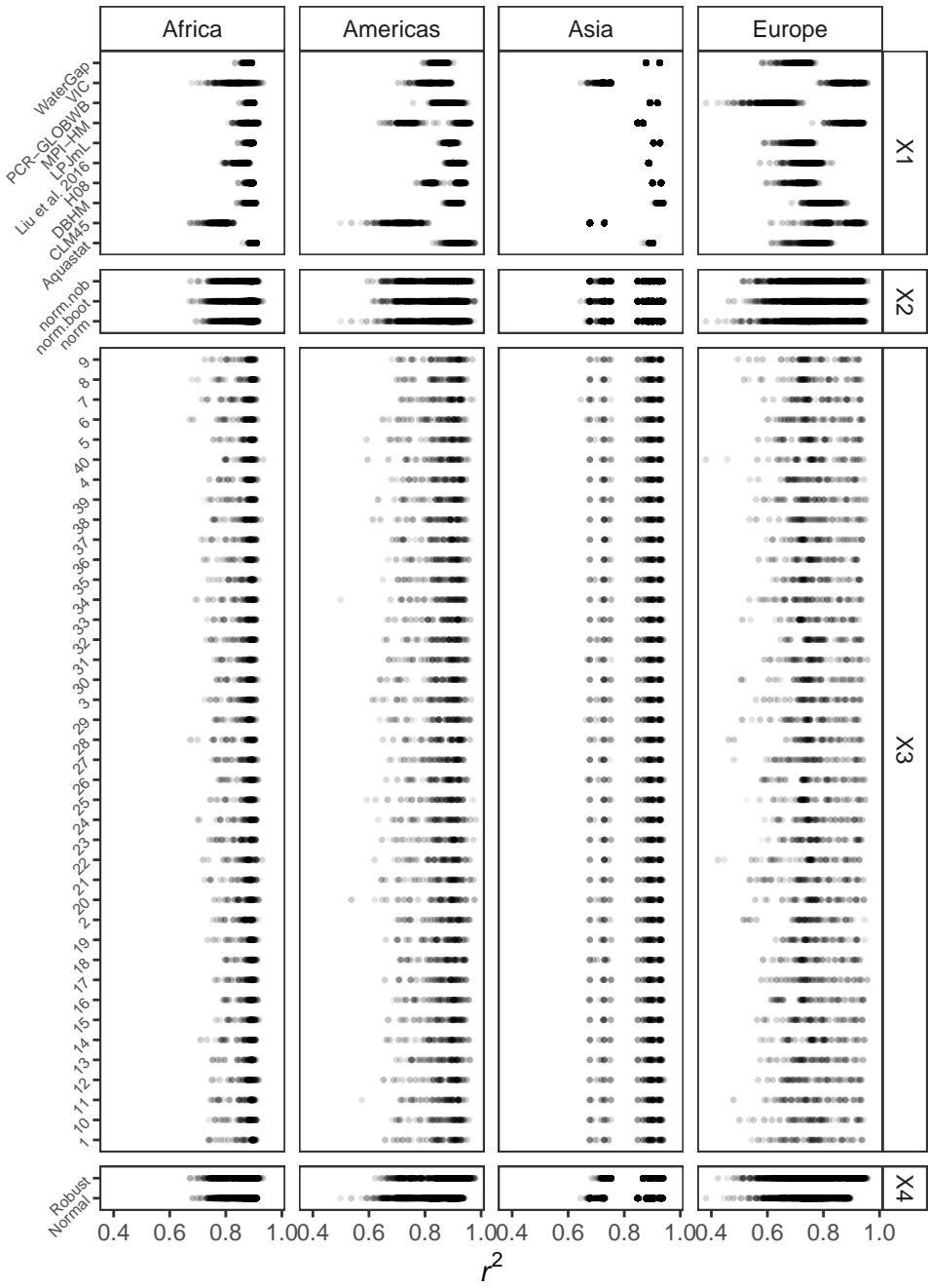


Figure 8: Scatterplots of  $r^2$  against the triggers' levels.

bottom

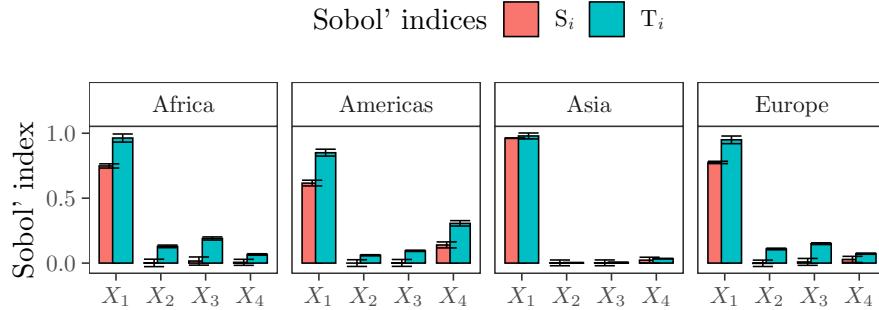


Figure 9: Sobol' indices.  $S_i$  and  $T_i$  refer respectively to Sobol' first and total order indices.  $S_i$  measures the influence of a parameter in the model output, while  $T_i$  measures the influence of a parameter jointly with its interactions.

```
# 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.

# CHECK SUM OF FIRST-ORDER INDICES -----
indices[sensitivity == "Si", sum(original), Continent]

##      Continent      V1
## 1:    Africa 0.7720726
## 2:  Americas 0.7566790
## 3:     Asia 0.9877018
## 4:   Europe 0.8099094

# 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",
```

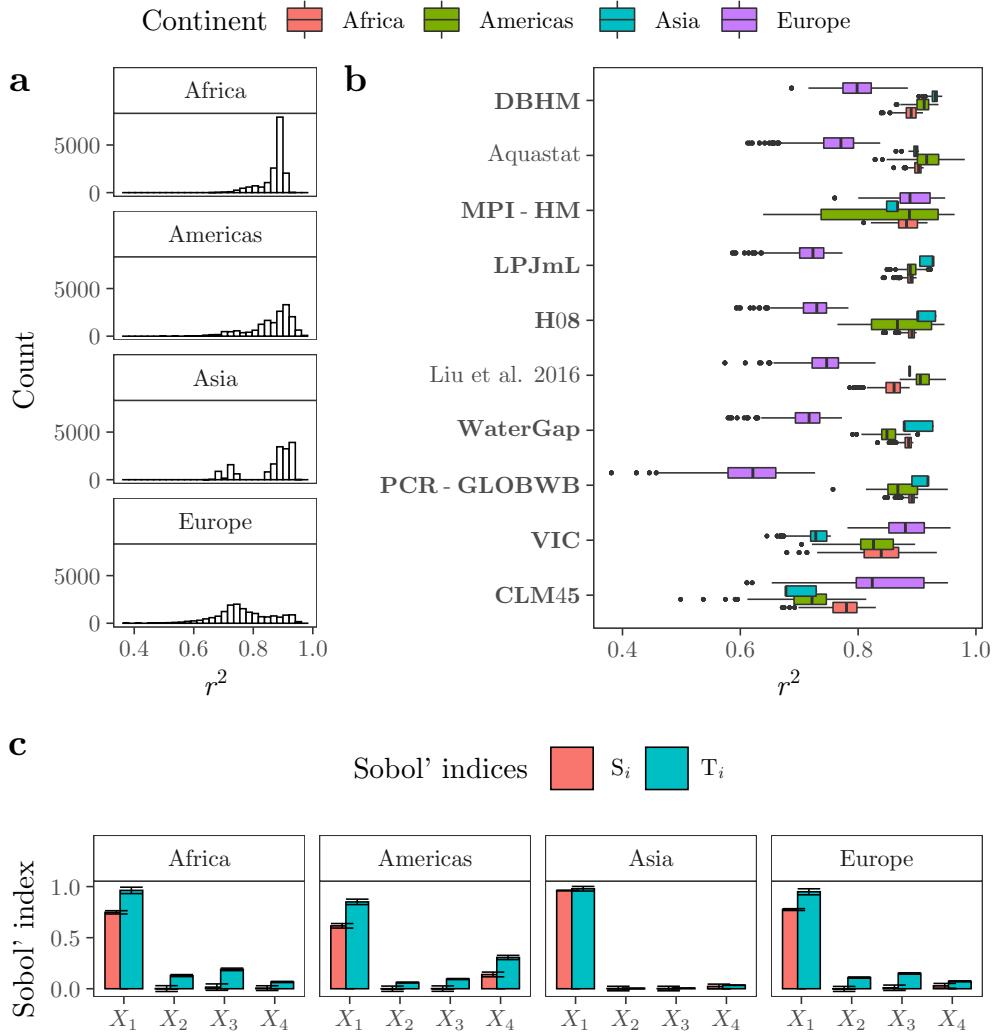


Figure 10: Uncertainty and sensitivity analysis. a) Empirical distribution for  $r^2$  at the continental level. b) Boxplots of  $r^2$  values obtained when regressions were run with GHM (in bold) and FAO-based datasets. c) Sobol' indices.  $S_i$  and  $T_i$  refer respectively to Sobol' first and total 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.

```

    space = "free_x") +
scale_color_discrete(name = "Continent") +
scale_y_continuous(breaks = pretty_breaks(n = 3)) +
labs(x = "",
y = "Sobol' index") +
theme_AP() +
theme(strip.background = element_rect(fill = "white"))

```

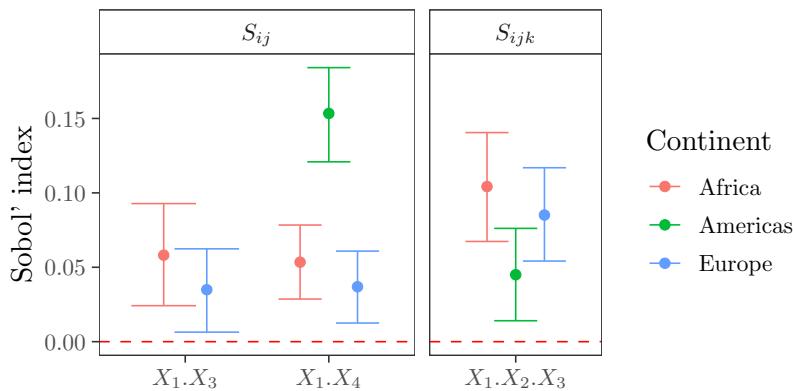


Figure 11: High-order interactions between the triggers. The dots and the errorbars show the mean and the 95% confidence intervals after bootstrapping ( $R=1000$ ).

## 11 Irrigated areas vs irrigation water withdrawals at different scales

### 11.1 Australia (scheme level), USA (State level) and Colorado (County level)

```

# RETRIEVE DATA FROM DIFFERENT SCALES -----
# Australia 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)] %>%
  .[, Scale:= "Australian \n irrigation systems"]

# USA level -----
# Irrigated area is in thousand acres
# Water withdrawals in thousand acres feet

```

```

# To convert from acre feet to cubic meters, multiply IWW by 1233.48.
# To convert from acre to ha, divide the area value by 2.471.

col_transf <- c("Irrigated.Area", "Water.Withdrawal")

solley.dt <- fread("solley.dt.csv") %>%
  .[, Scale:= "USA"] # state level

colorado.dt <- fread("colorado_data.csv") # colorado
colorado.dt <- colorado.dt[, `:=` (Irrigated.Area = Flood + Sprinkler + Micro,
                                Water.Withdrawal = Groundwater + Surface.water)] %>%
  .[, .SD, .SDcols = (col_transf)] %>%
  .[, Scale:= "Colorado counties"]

# Merge both data sets
usa.dt <- rbind(solley.dt, colorado.dt) %>%
  .[, Irrigated.Area:= (Irrigated.Area * 1000) / 2.471] %>%
  .[, Water.Withdrawal:= Water.Withdrawal * 1000 * 1233.48]

# PLOT SCATTERPLOT -----
scatter_usa <- ggplot(usa.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))) +
  labs(x = "Irrigated area (ha)",
       y = "Irrigation water \n Withdrawals (m$^3$)") +
  facet_wrap(~Scale) +
  theme_AP() +
  theme(strip.background = element_rect(fill = "white"))

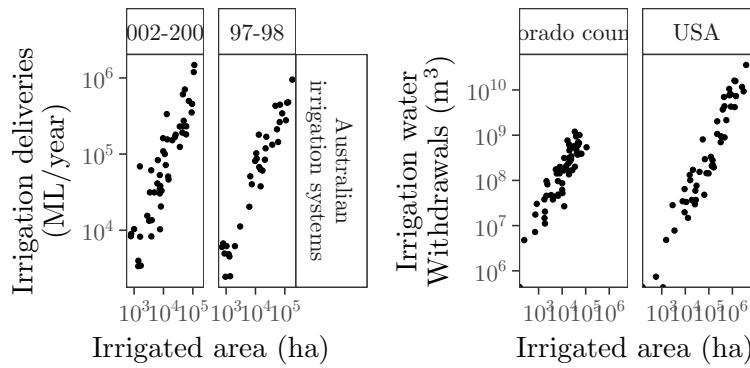
scatter_australia <- ggplot(australia.dt, aes(Irrigated.Area, Water.Withdrawal)) +
  geom_point(size = 0.6) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  facet_grid(Scale~Year) +
  labs(x = "Irrigated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

all_scatters <- plot_grid(scatter_australia, scatter_usa, ncol = 2)

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

```

```
all_scatters
```



```
# COMPUTE REGRESSIONS AND BOOTSTRAP -----
```

```
# AUSTRALIA IRRIGATION SYSTEMS -----
```

```
australia.dt <- 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))

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

# Confidence intervals
ci_plot1 <- boot.ci(foo, type = "all")
```

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

```
# USA LEVEL (COLORADO COUNTY AND COUNTY LEVEL) -----
```

```
usa.dt.lm <- usa.dt[, (col_transf) := lapply(.SD, log10), .SDcols = col_transf] %>%
  NaRV.omit()

usa.regressions <- usa.dt.lm %>%
  group_by(Scale) %>%
  nest() %>%
  mutate(fit = map(.x = data, .f = ~lm(Water.Withdrawal ~ Irrigated.Area,
                                         data = .)),
         results = map(fit, glance)),
```

```

    residuals = map(fit, augment))

# Bootstrap
usa.loop <- c("USA", "Colorado counties")
foo.usa <- lapply(usa.loop, function(x)
  boot(usa.dt.lm[Scale == x], function(data, indices)
    summary(lm(Water.Withdrawal ~ Irrigated.Area,
               data[indices, ] ))$r.squared, R = 5000))

names(foo.usa) <- usa.loop

# Confidence intervals
ci_plot2 <- lapply(foo.usa, function(x) boot.ci(x, type = "all"))

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

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

ci.dt <- data.table(Scale = c("USA", "Colorado counties", "Australian irrigation systems"),
                      low.ci = c(ci_plot2$`USA`$bca[[4]],
                                 ci_plot2$`Colorado counties`$bca[[4]],
                                 ci_plot1$bca[[4]]),
                      high.ci = c(ci_plot2$`USA`$bca[[5]],
                                 ci_plot2$`Colorado counties`$bca[[5]],
                                 ci_plot1$bca[[5]])) %>%
  melt(., measure.vars = c("low.ci", "high.ci"))

# PLOT ALL -----
tmp <- lapply(usa.loop, function(x) foo.usa[[x]]$t) %>%
  lapply(., data.table)

names(tmp) <- usa.loop

all.histo <- rbindlist(tmp, idcol = "Scale") %>%
  rbind(., data.table(foo$t) %>%
    .[, Scale:= "Australian irrigation systems"] %>%
    setcolororder(., c("Scale", "V1"))) %>%
  .[, Scale:= factor(Scale, levels = c("Colorado counties", "USA",
                                         "Australian irrigation systems"))] %>%
  ggplot(., aes(V1, group = Scale)) +
  geom_histogram(color = "black", fill = "white") +
  geom_vline(data = ci.dt, aes(xintercept = value), color = "blue") +
  labs(x = "$r^2$",
       y = "Count") +
  facet_wrap(~Scale) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +

```

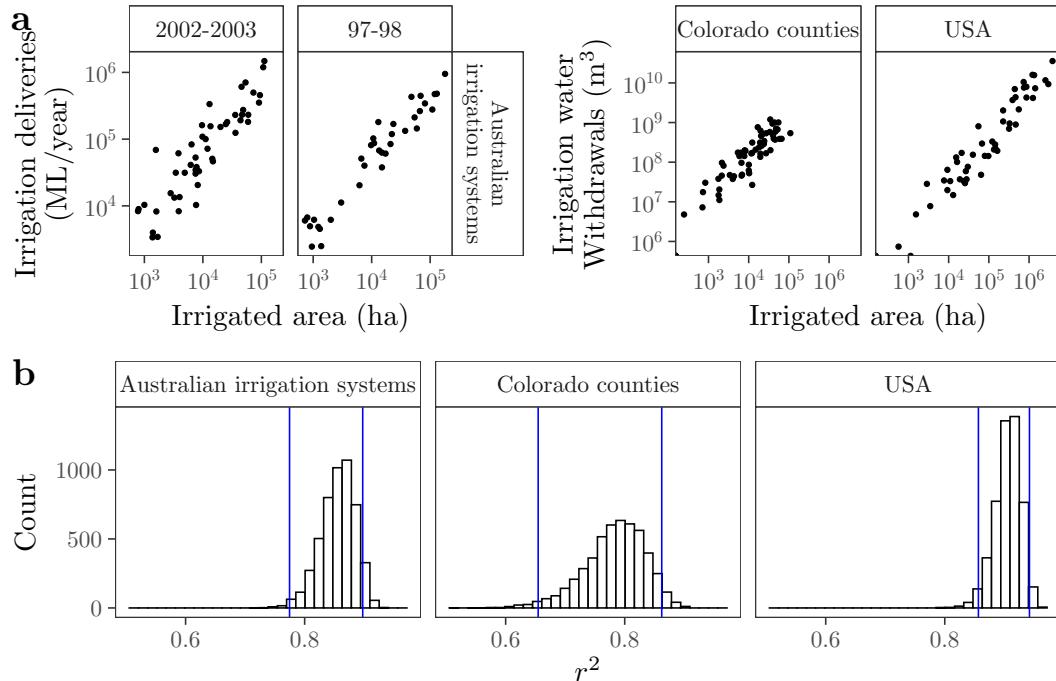
```

theme_AP() +
theme(strip.background = element_rect(fill = "white"))

plot_grid(all_scatters, all.histo, ncol = 1, labels = "auto")

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

```



## 11.2 Regression diagnostics

```

# REGRESSION DIAGNOSTICS ----

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

# USA -----
usa.residuals <- usa.regressions %>%
  dplyr::select(Scale, residuals) %>%
  unnest(residuals) %>%
  data.table()

# functions to plot:

# residuals versus fitted
plot_res <- function(dt) {
  gg <- ggplot(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") +
  theme_AP() +
  theme(strip.text.y = element_text(size = 6.4),
        strip.background = element_rect(fill = "white"))
  return(gg)
}

# qq plot
plot_qq <- function(dt) {
  gg <- ggplot(dt, aes(sample = .std.resid)) +
    stat_qq(size = 0.5) +
    stat_qq_line() +
    labs(x = "Theoretical quantiles",
         y = "Standardized Residuals") +
    theme_AP() +
    theme(strip.text.y = element_text(size = 6.4),
          strip.background = element_rect(fill = "white"))
  return(gg)
}

# PLOT FOR AUSTRALIA ----

a <- plot_res(australia.residuals) +
  facet_grid(~Year, scales = "free_y")

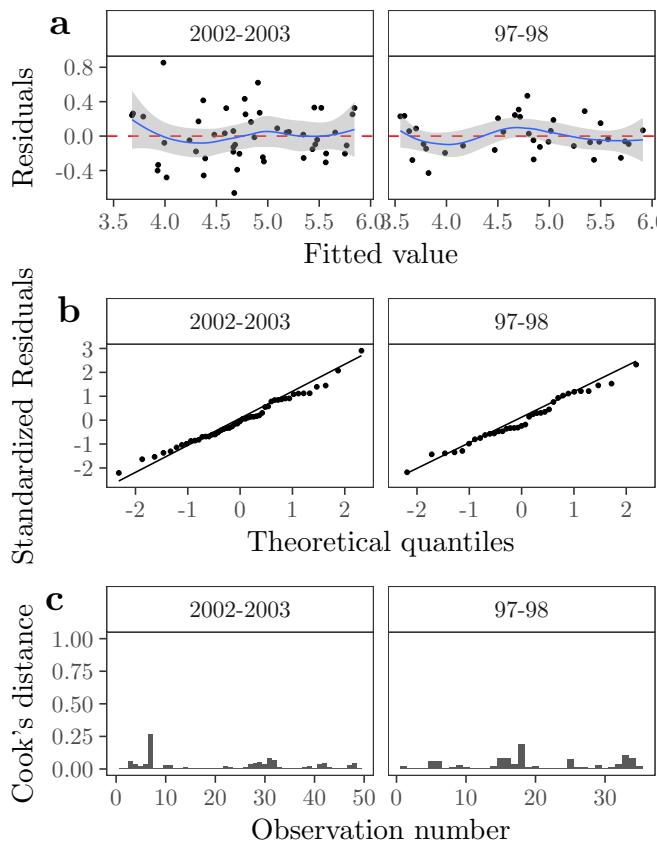
b <- plot_qq(australia.residuals) +
  facet_grid(~Year, scales = "free_y")

ID <- australia.residuals[, .N, Year]
numb <- unlist(sapply(ID[, N], function(x) 1:x))
c <- cbind(australia.residuals, numb) %>%
  ggplot(., aes(numb, .cooksdi)) +
  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'

```



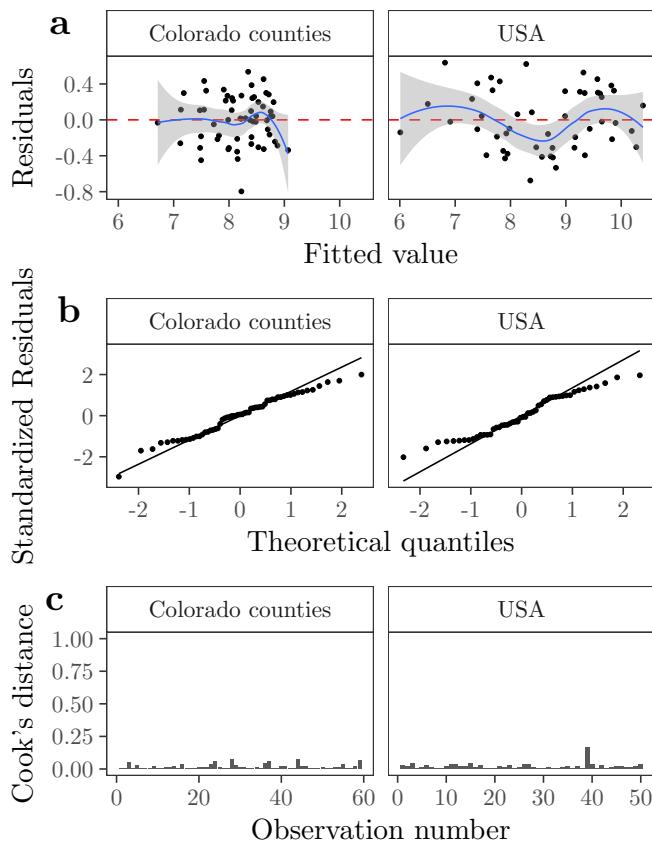
```
# PLOT FOR USA -----
a <- plot_res(usa.residuals) +
  facet_grid(~Scale, scales = "free_y")

b <- plot_qq(usa.residuals) +
  facet_grid(~Scale, scales = "free_y")

ID <- usa.residuals[, .N, Scale]
numb <- unlist(sapply(ID[, N], function(x) 1:x))
c <- cbind(usa.residuals, numb) %>%
  ggplot(., aes(numb, .cooksdi)) +
  geom_col() +
  scale_y_continuous(limits = c(0, 1)) +
  facet_grid(~Scale, 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'
```



```

# BOOTSTRAP BETA -----  

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

# USA
foo.states <- boot(usa.dt.lm[Scale == "USA"], function(data,indices)
  coef(lm(Water.Withdrawal ~ Irrigated.Area,
    data[indices, ])), R = 5000)  

foo.counties <- boot(usa.dt.lm[Scale == "Colorado counties"], function(data,indices)
  coef(lm(Water.Withdrawal ~ Irrigated.Area,
    data[indices, ])), R = 5000)  

# Confidence intervals
ci_plot2 <- lapply(list(foo.states,
  foo.counties,
  foo.beta), function(x) boot.ci(x, type = "all"))

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

```

```

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

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

names(ci_plot2) <- c("USA", "Colorado counties", "Australian irrigation systems")

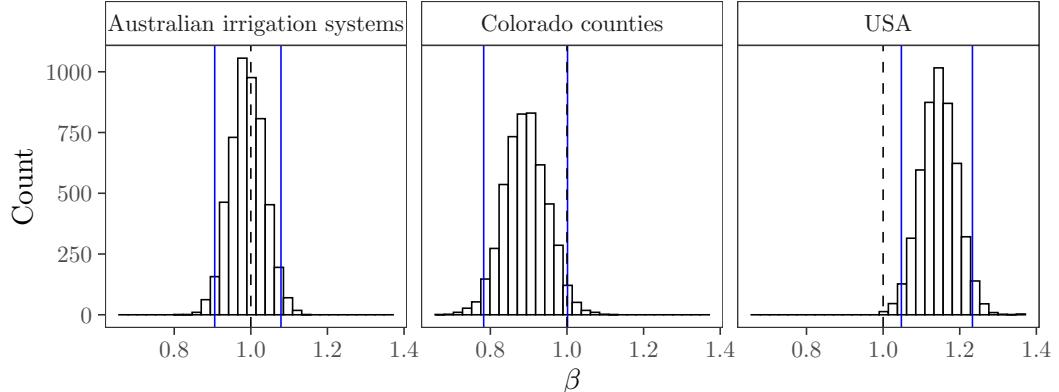
ci.dt <- data.table(Scale = c("USA", "Colorado counties", "Australian irrigation systems"),
                     low.ci = c(ci_plot2$`USA`$bca[[4]],
                                ci_plot2$`Colorado counties`$bca[[4]],
                                ci_plot2$`Australian irrigation systems`$bca[[4]]),
                     high.ci = c(ci_plot2$`USA`$bca[[5]],
                                ci_plot2$`Colorado counties`$bca[[5]],
                                ci_plot2$`Australian irrigation systems`$bca[[5]])) %>%
melt(., measure.vars = c("low.ci", "high.ci"))

tmp <- rbind(data.table(foo.states$t)[, Scale:= "USA"],
              data.table(foo.counties$t)[, Scale:= "Colorado counties"],
              data.table(foo.beta$t)[, Scale:= "Australian irrigation systems"])

# PLOT BETA -----
# Plot
ggplot(tmp, aes(V1)) +
  geom_histogram(color = "black", fill = "white") +
  geom_vline(data = ci.dt, aes(xintercept = value), color = "blue") +
  geom_vline(xintercept = 1, lty = 2) +
  labs(x = "$\\beta$", y = "Count") +
  facet_wrap(~Scale) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP() +
  theme(strip.background = element_rect(fill = "white"))

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

```



## 12 Irrigated area at the cell level

```
# PIXEL LEVEL ----

# Read HYDE 3.2
mirca <- fread("hyde_3.2_irrigated_2005.csv")

# 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_airrrww_global_monthly_1971_2010.nc",
  "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrrww_global_monthly_1861_2005.nc",
  "vic_wfdei_nobc_hist_pressoc_co2_airrrww_global_monthly_1971_2010.nc")
dname <- "airrrww"

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_pirr",
  dname = "pirrrww")

CLM45.mirca <- CLM45.mirca[, Water.Dataset:= "CLM45"] %>%
  setcolororder(., 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 <- 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 na
## Warning in countrycode(full.isimip[, Country], origin = "country.name", : Some values were na
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(Irrigated.Area.ha, 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 (ha)",
            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])))
  }
}

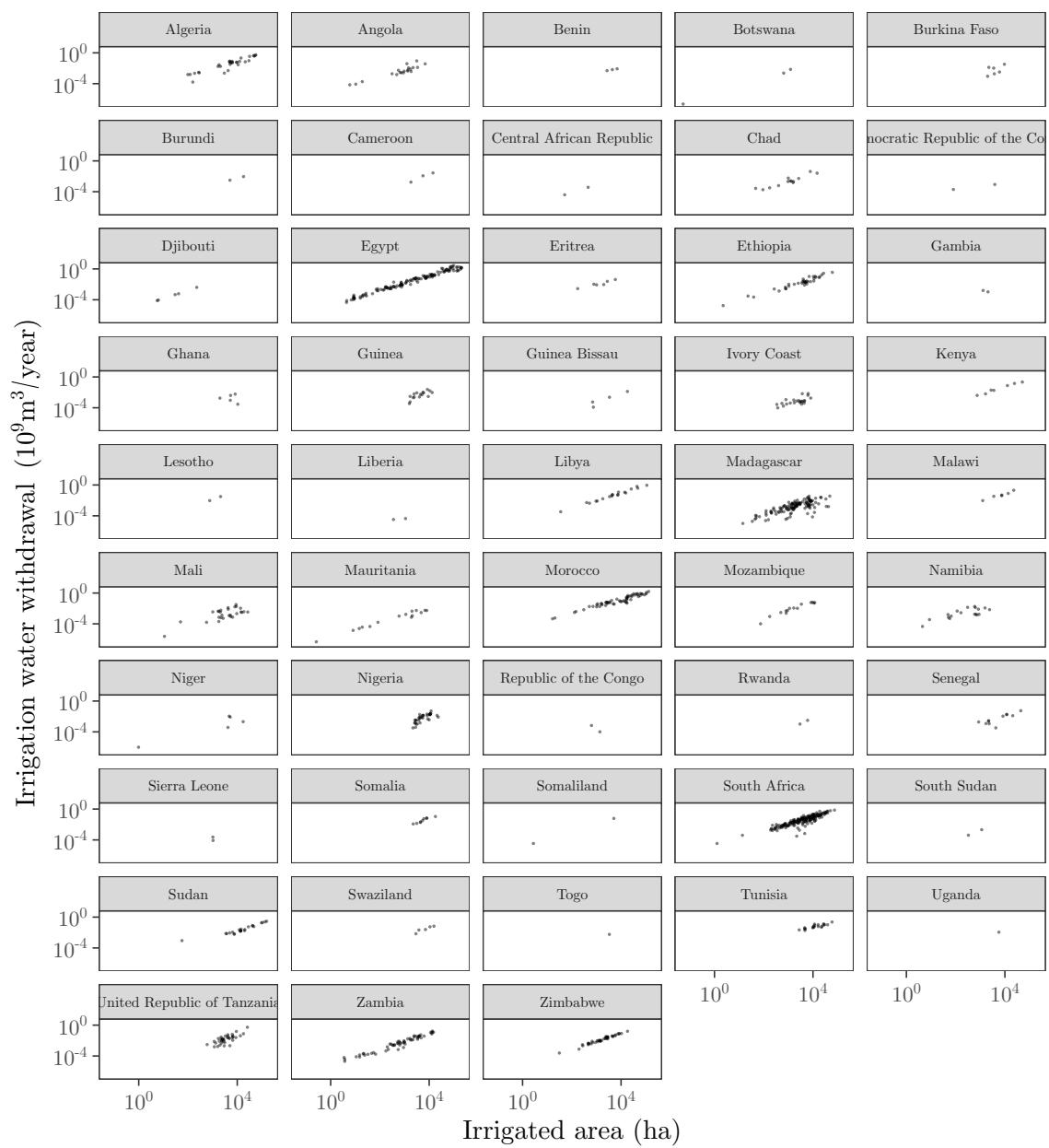
gg

## $Africa
## $Africa$CLM45

```

## Africa

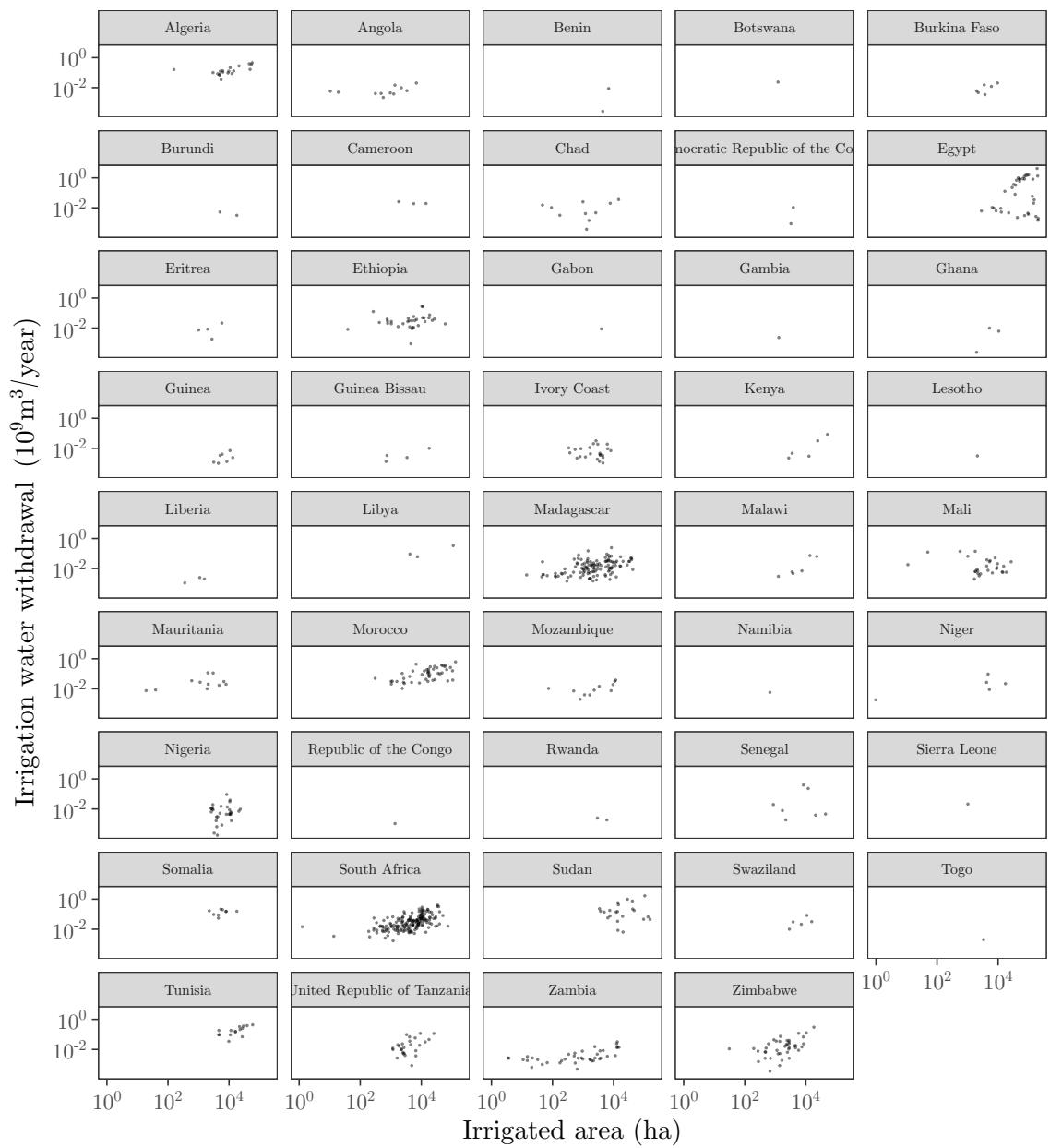
CLM45



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

## Africa

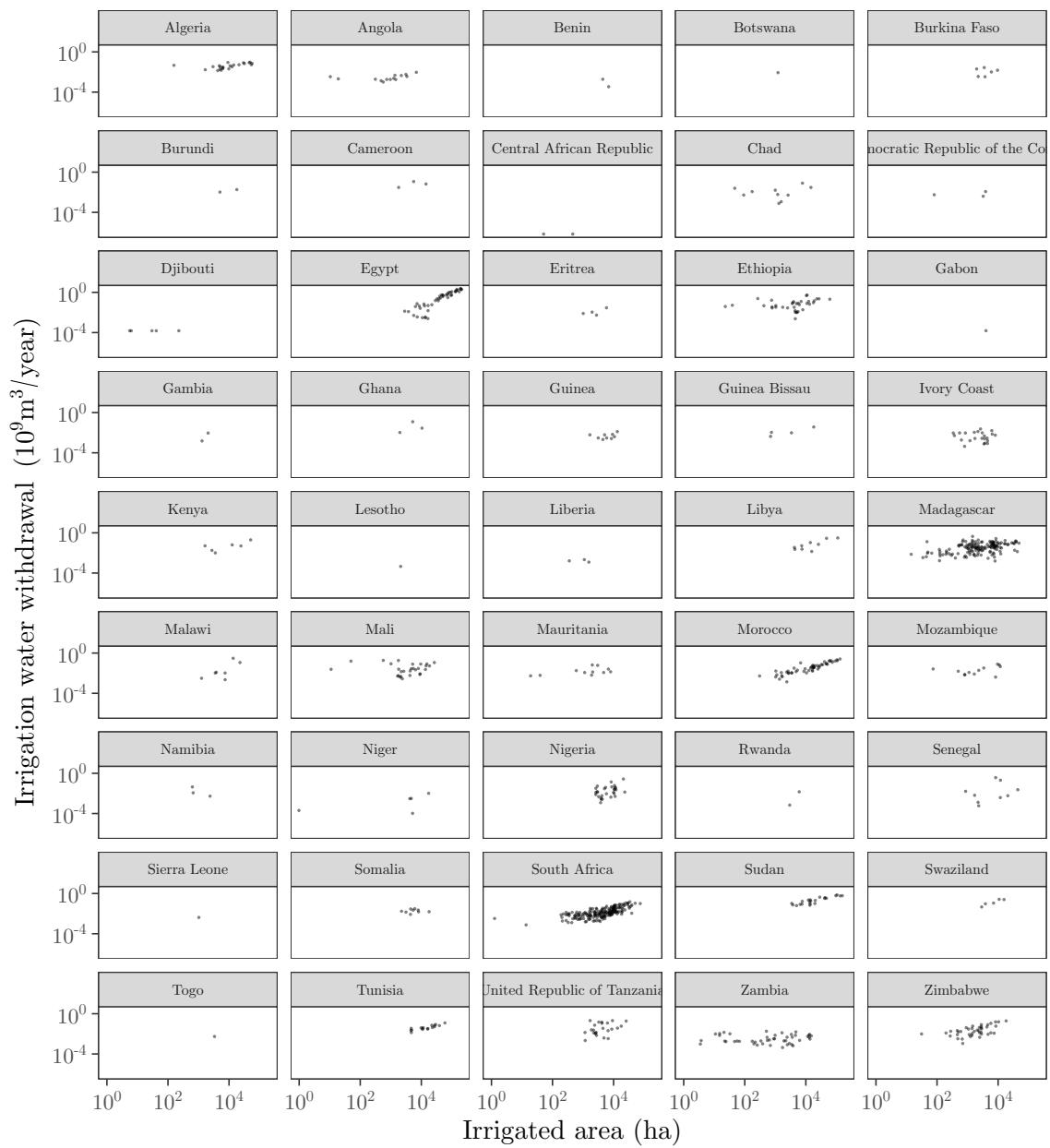
### DBHM



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

## Africa

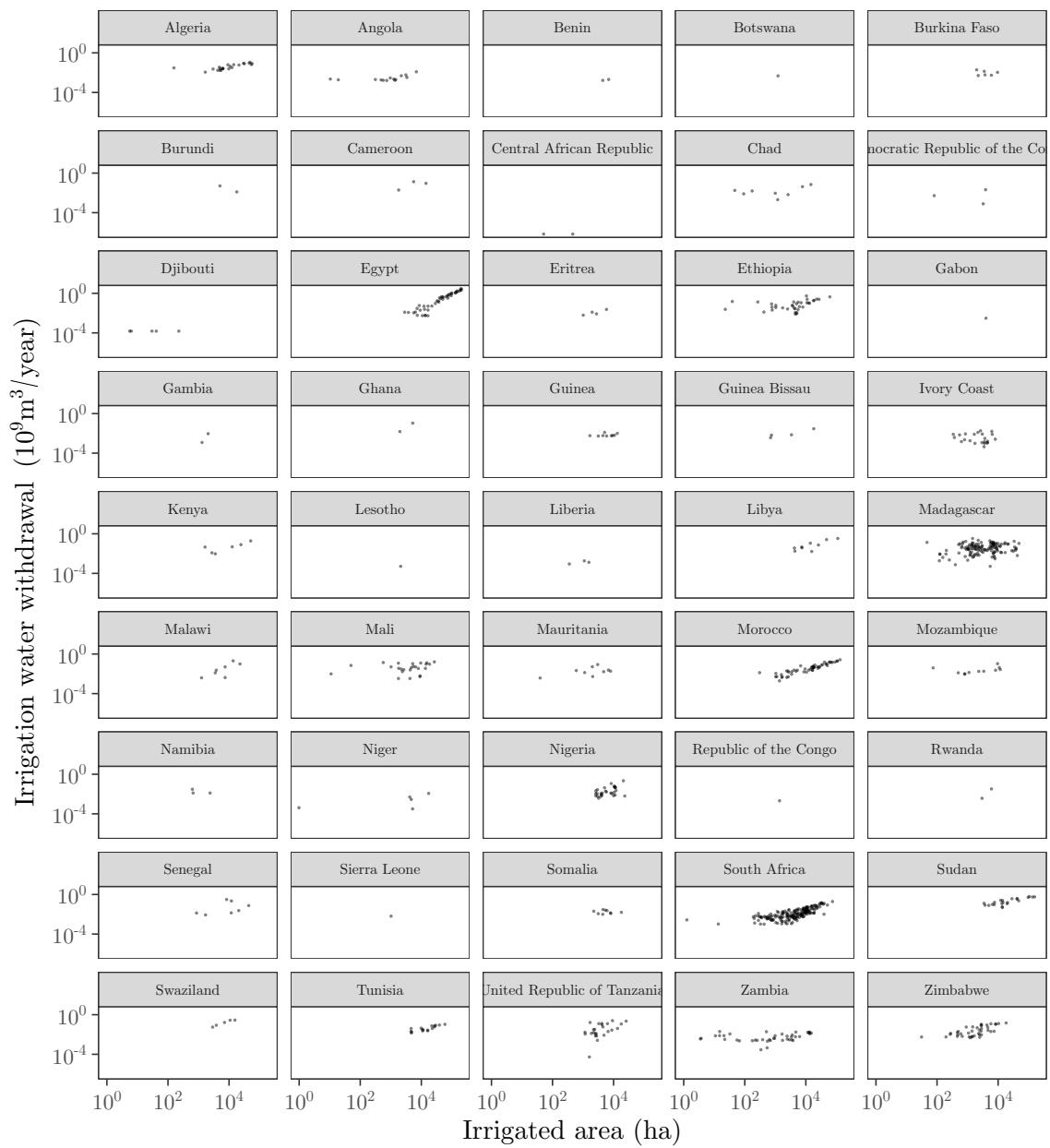
H08



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

## Africa

LPJmL

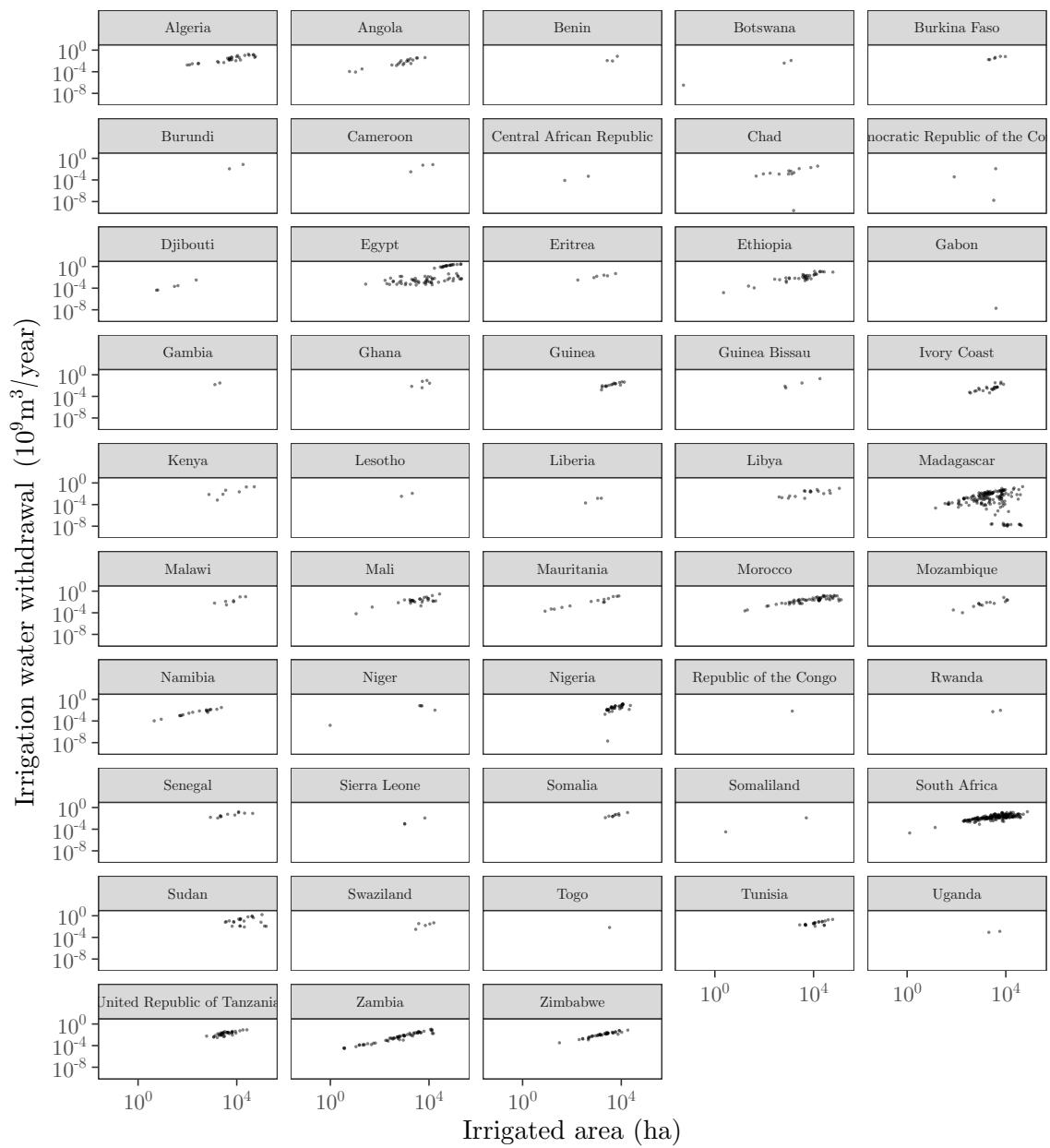


##

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

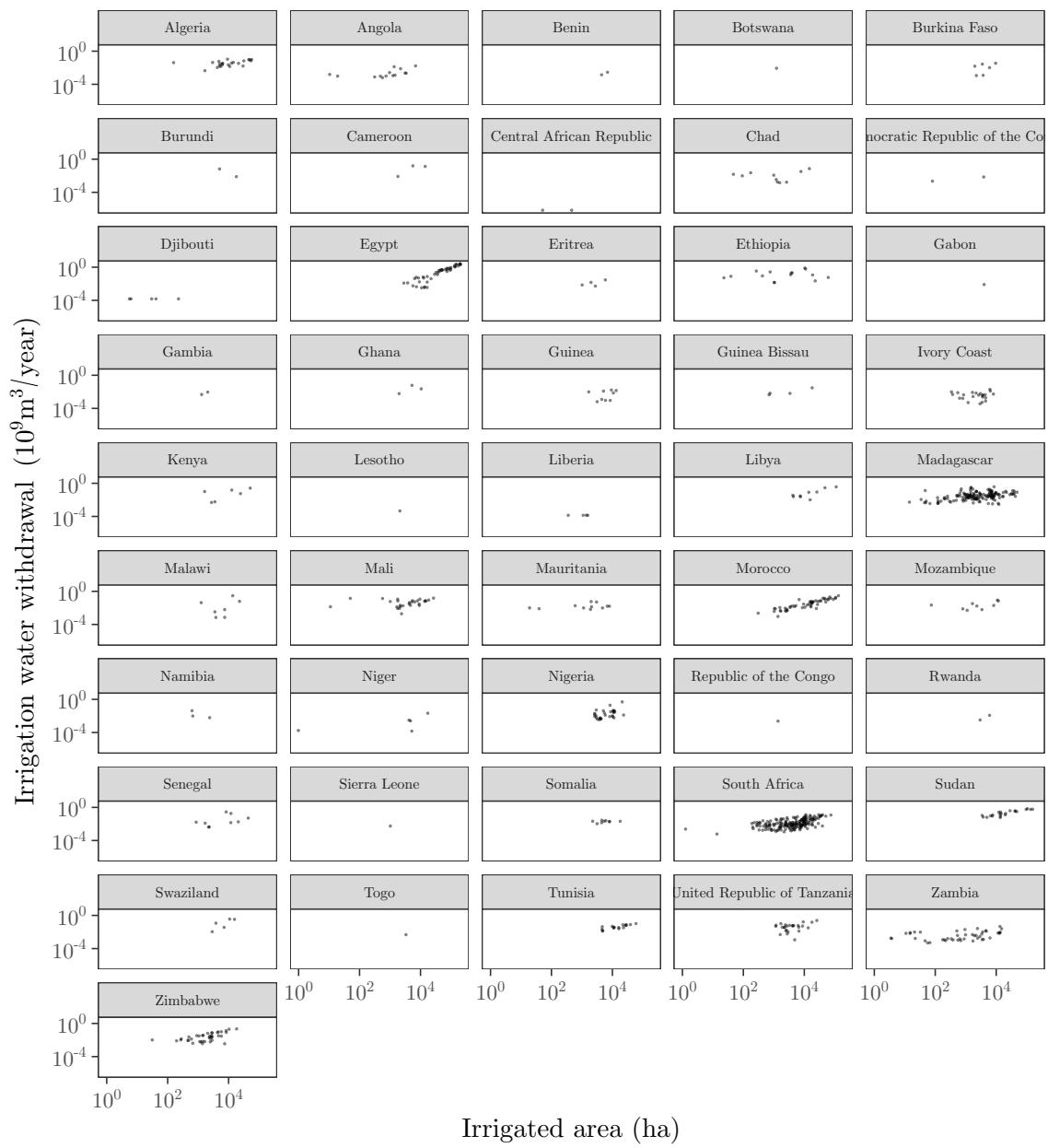
## Africa

MPI-HM



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

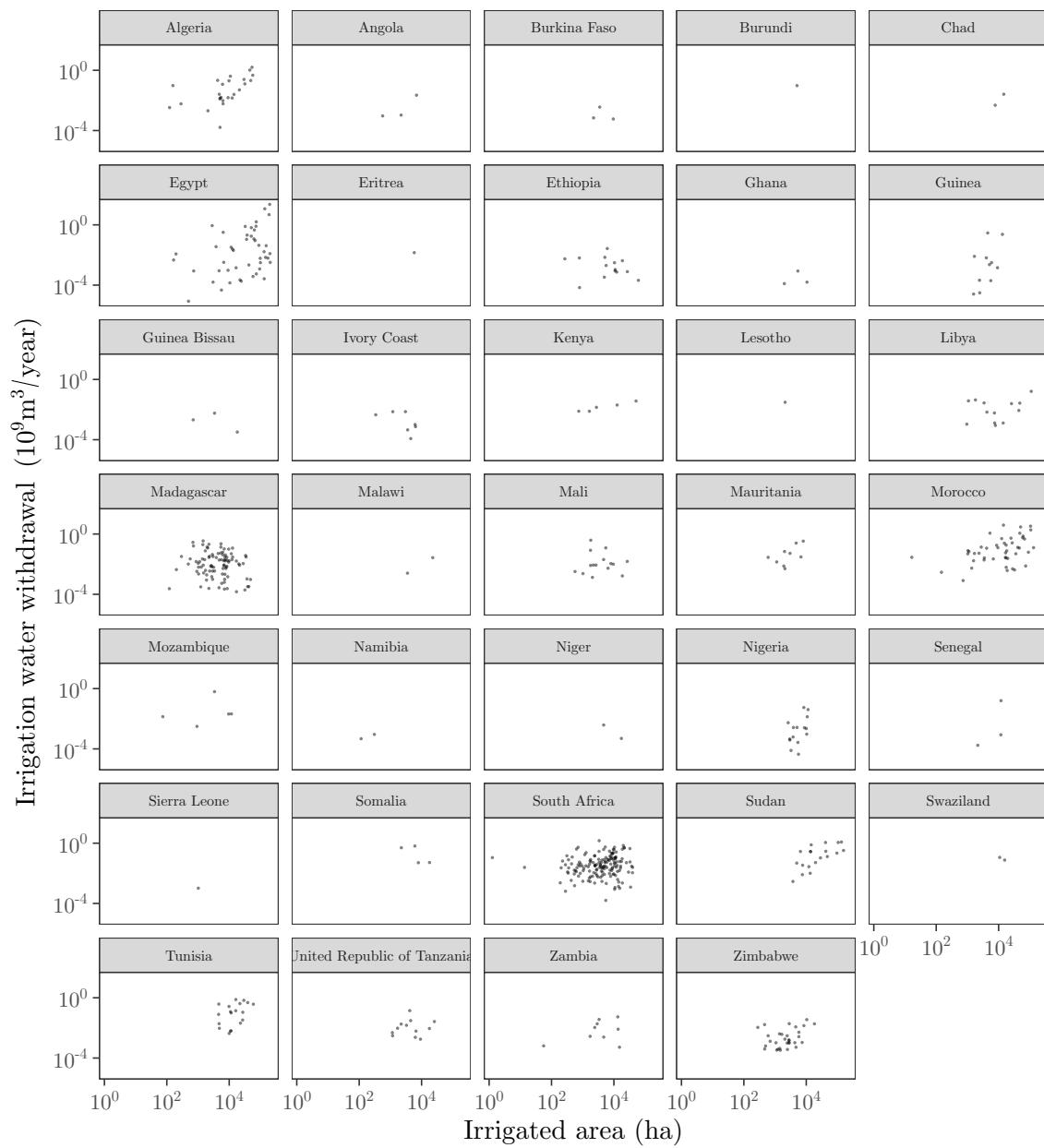
Africa  
PCR-GLOBWB



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

## Africa

### VIC

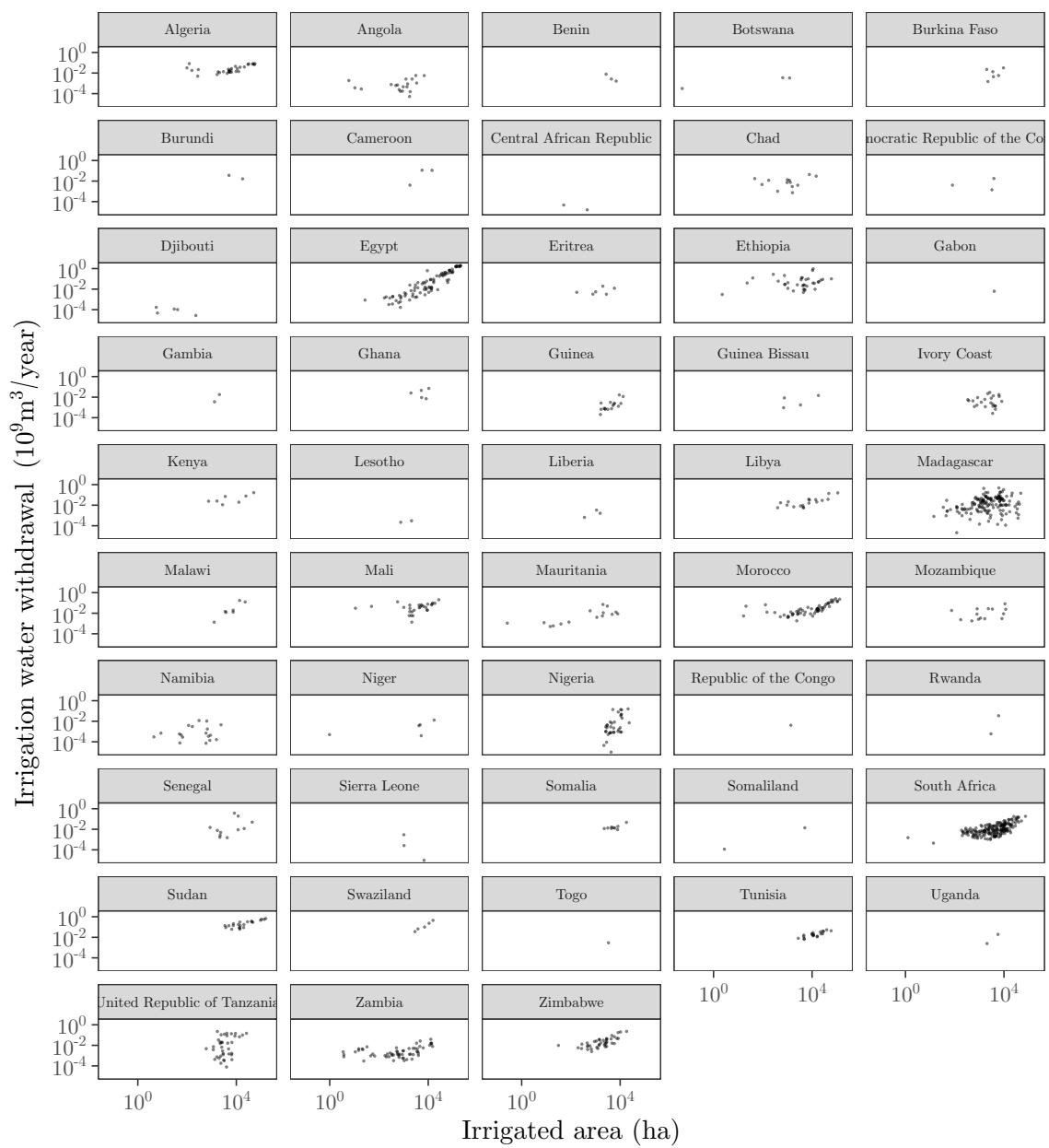


```
##
```

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

## Africa

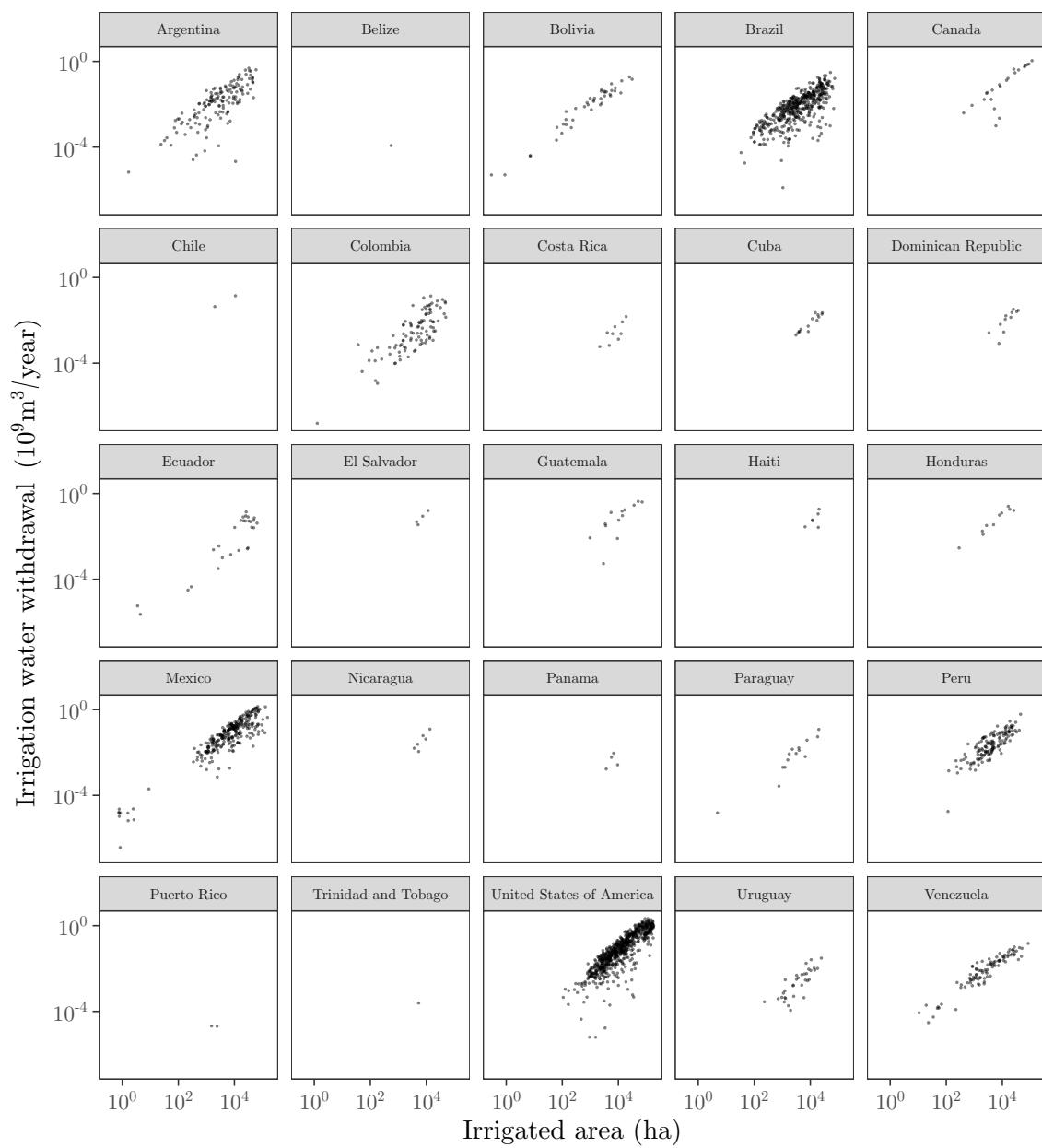
### WaterGap



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

## Americas

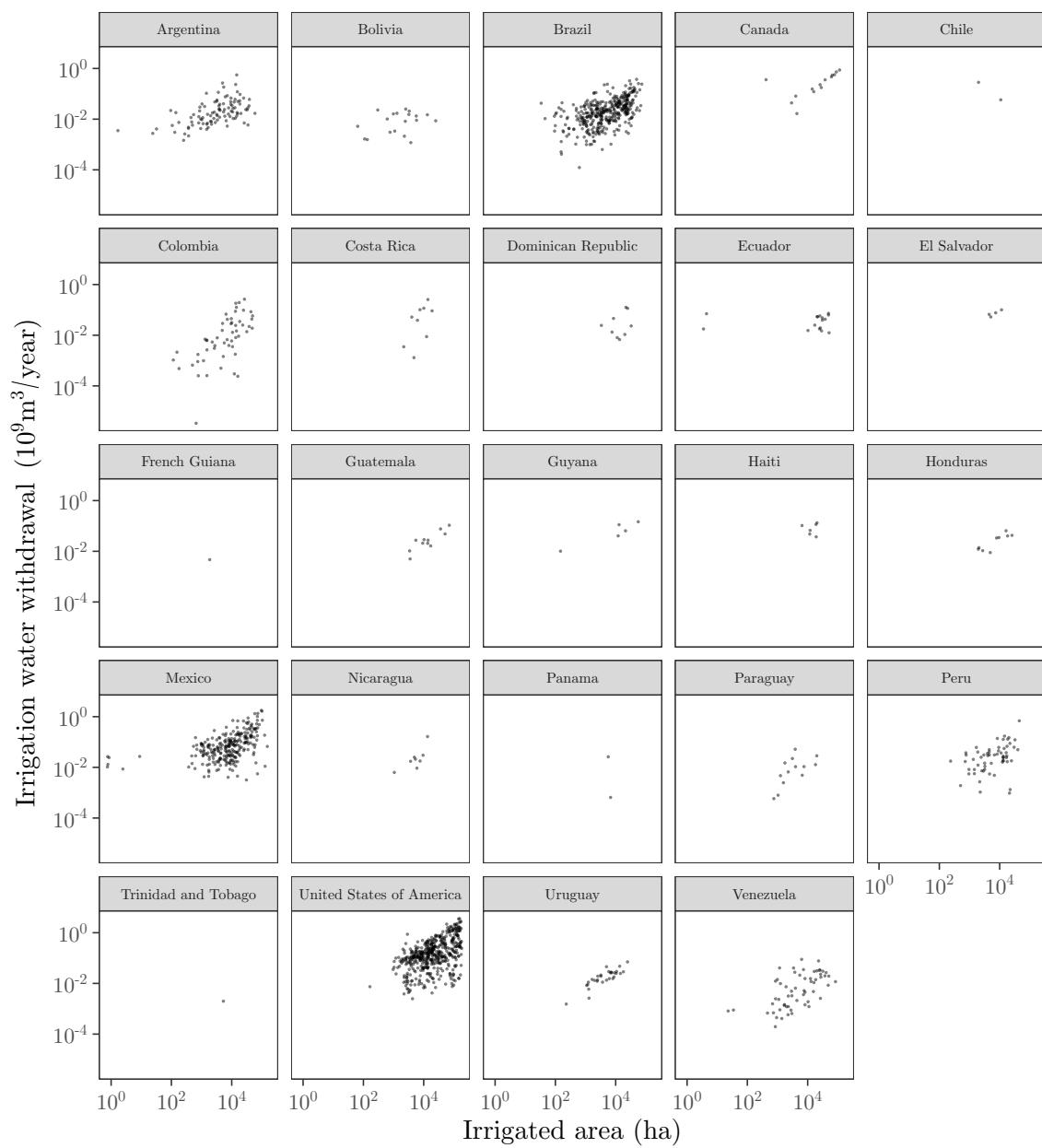
CLM45



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

## Americas

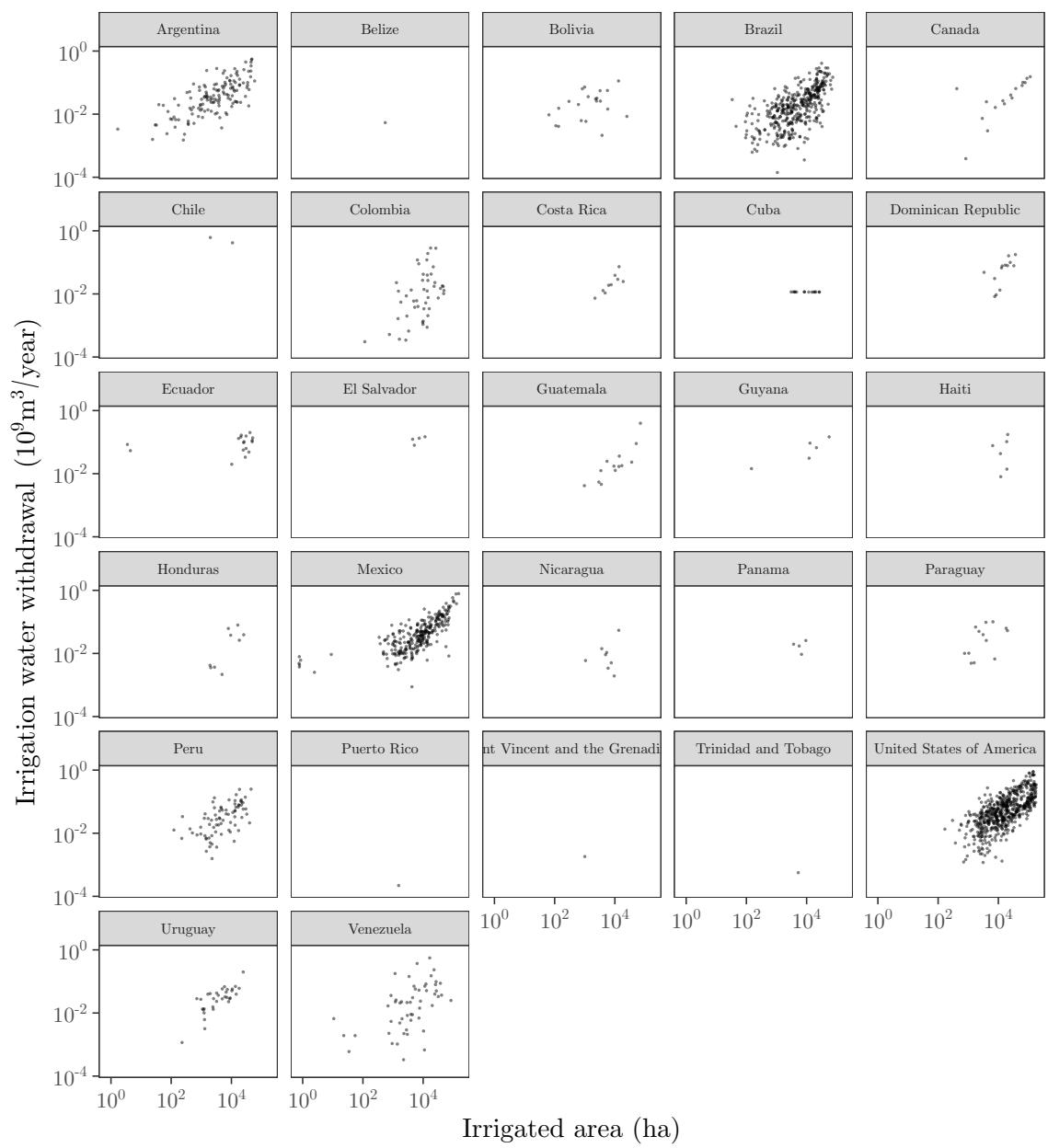
### DBHM



```
##  
## $Americas$H08
```

## Americas

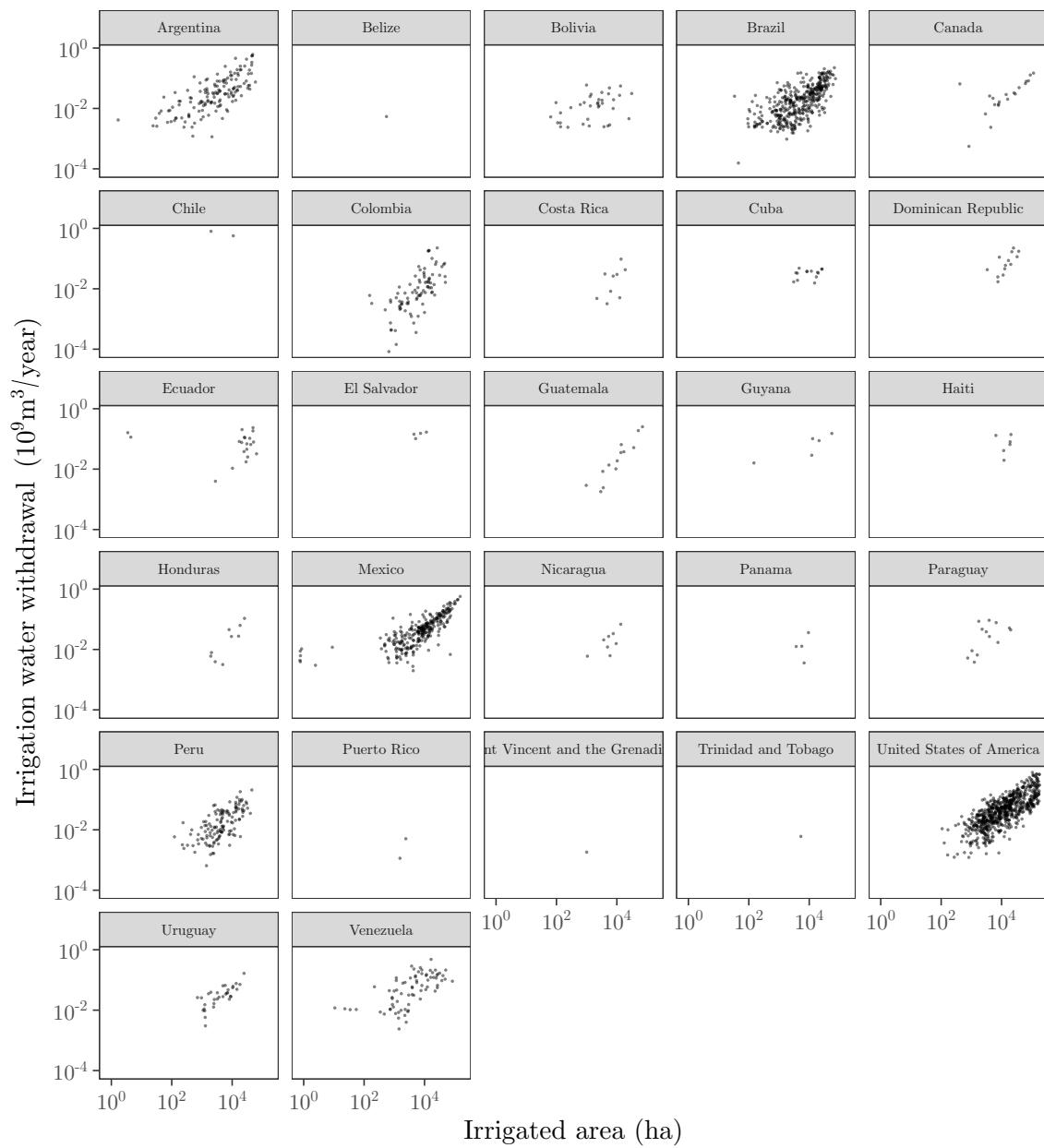
H08



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

## Americas

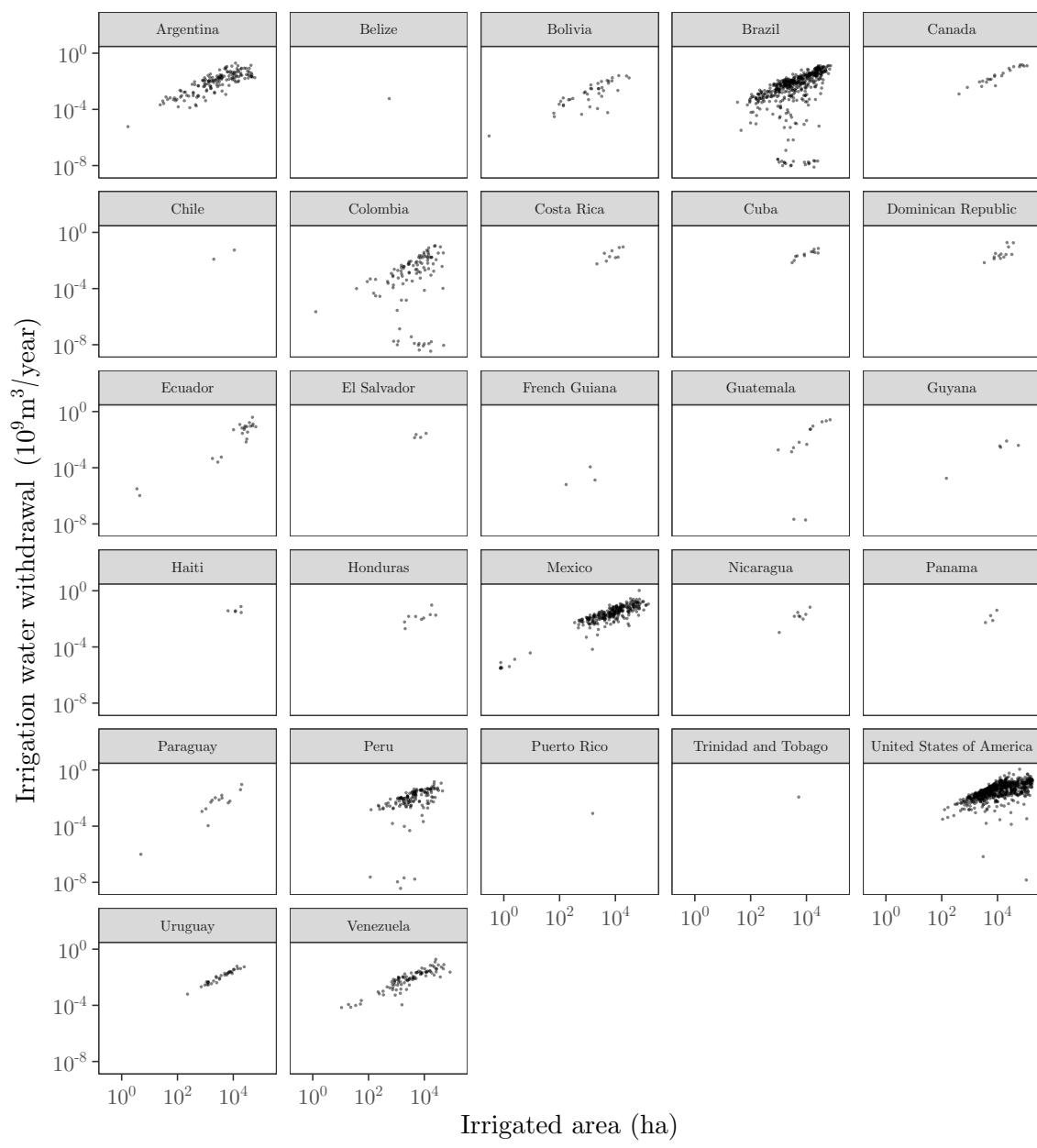
LPJmL



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

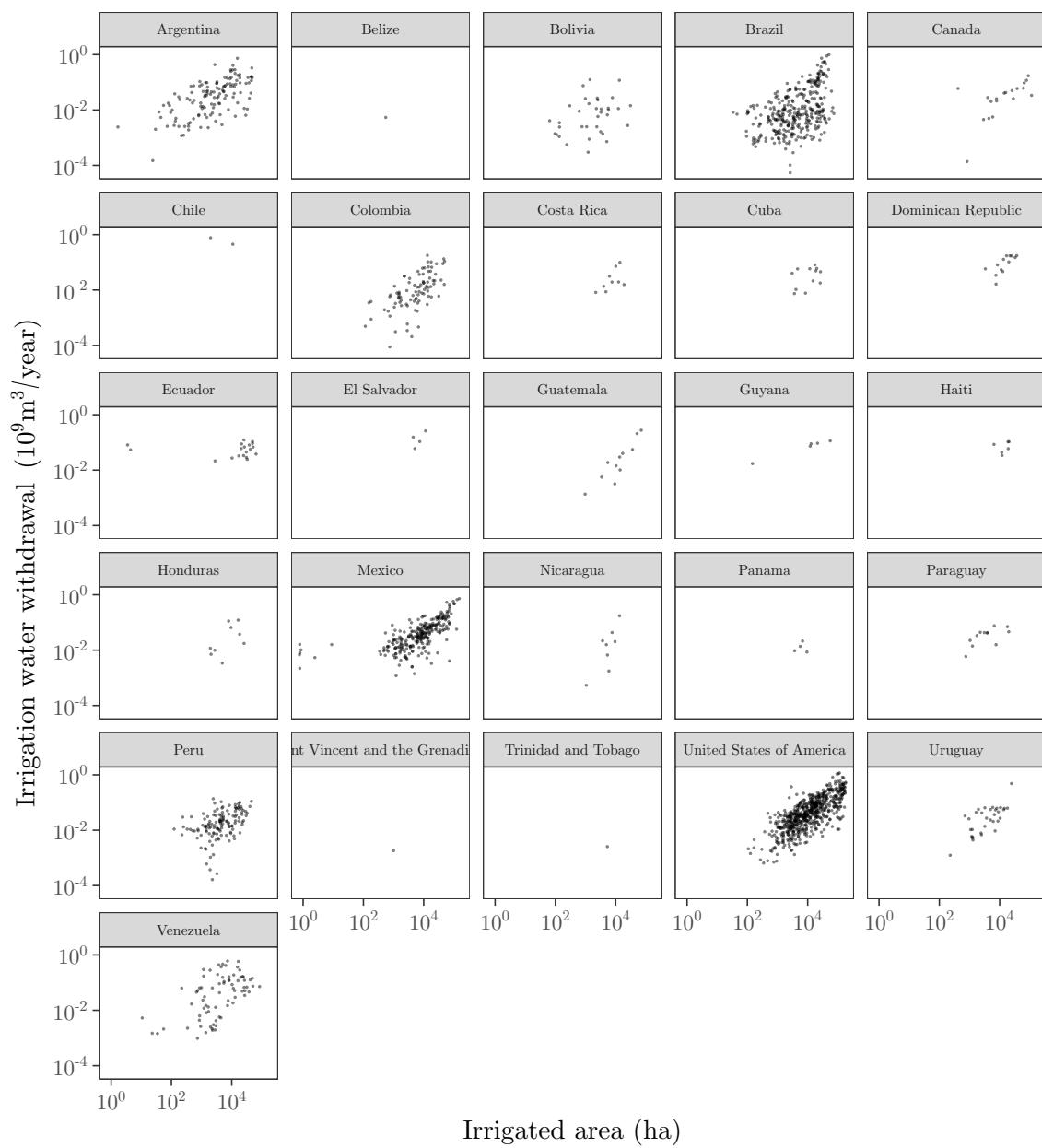
## Americas

MPI-HM



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

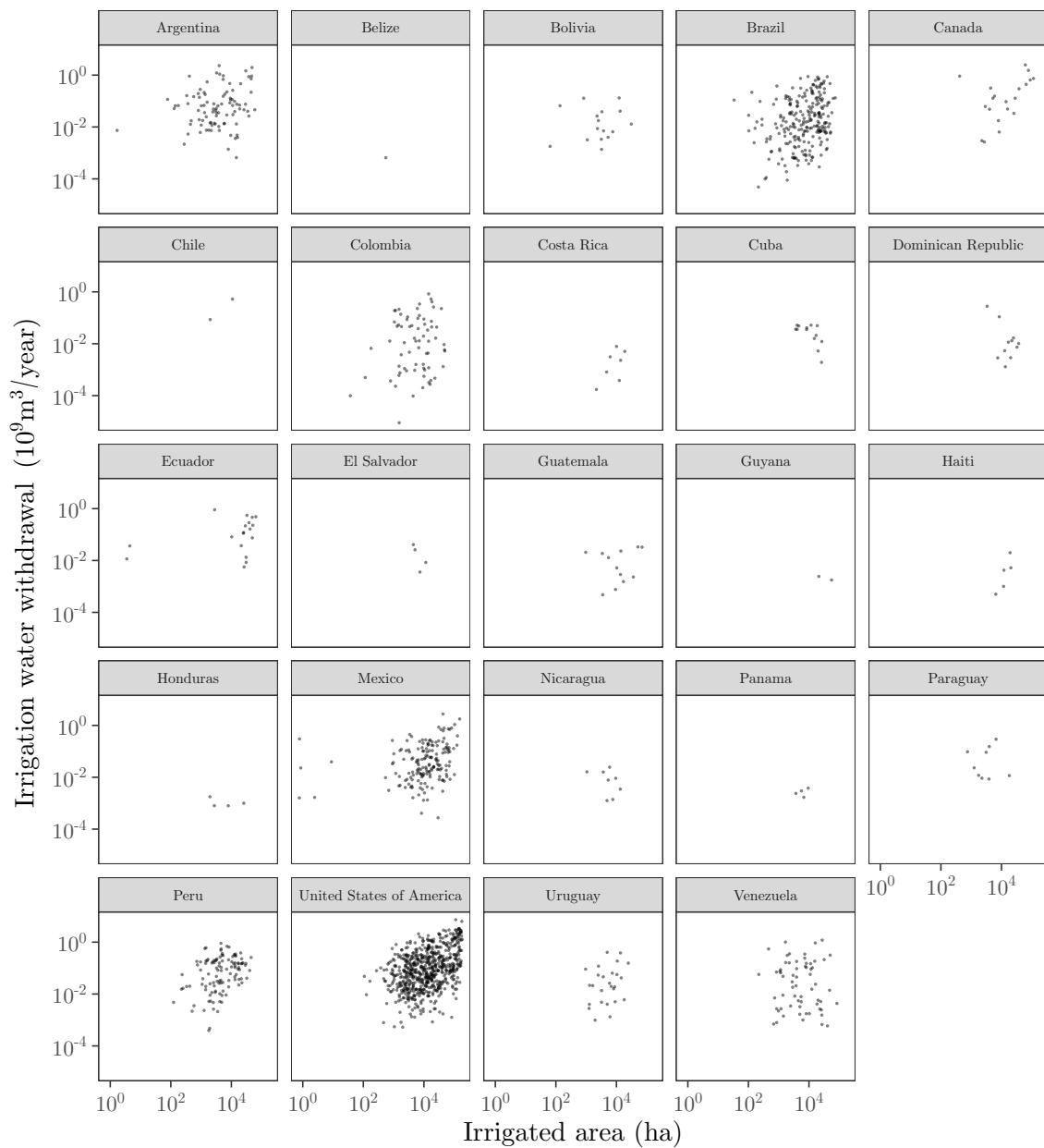
Americas  
PCR-GLOBWB



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

## Americas

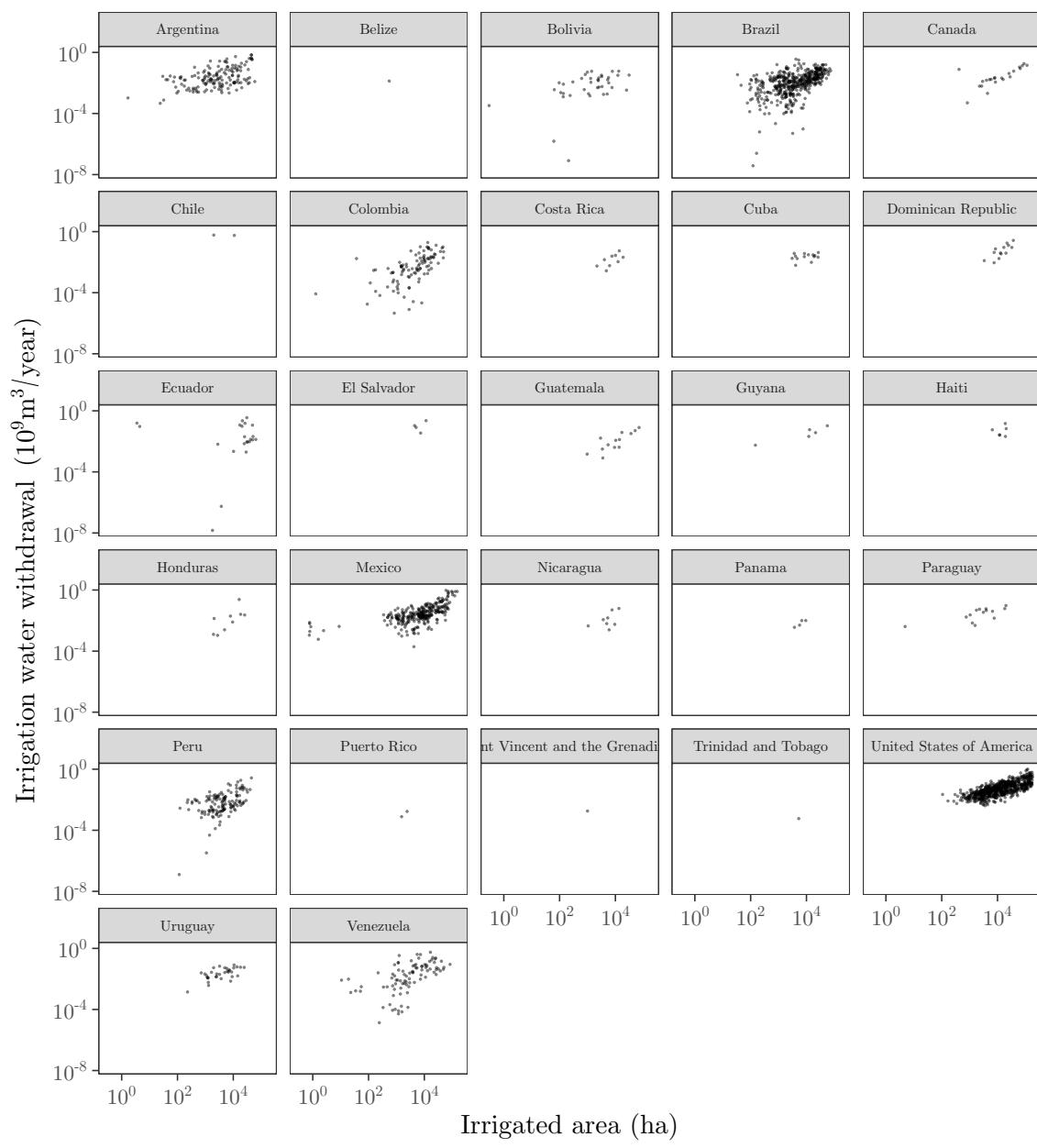
### VIC



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

## Americas

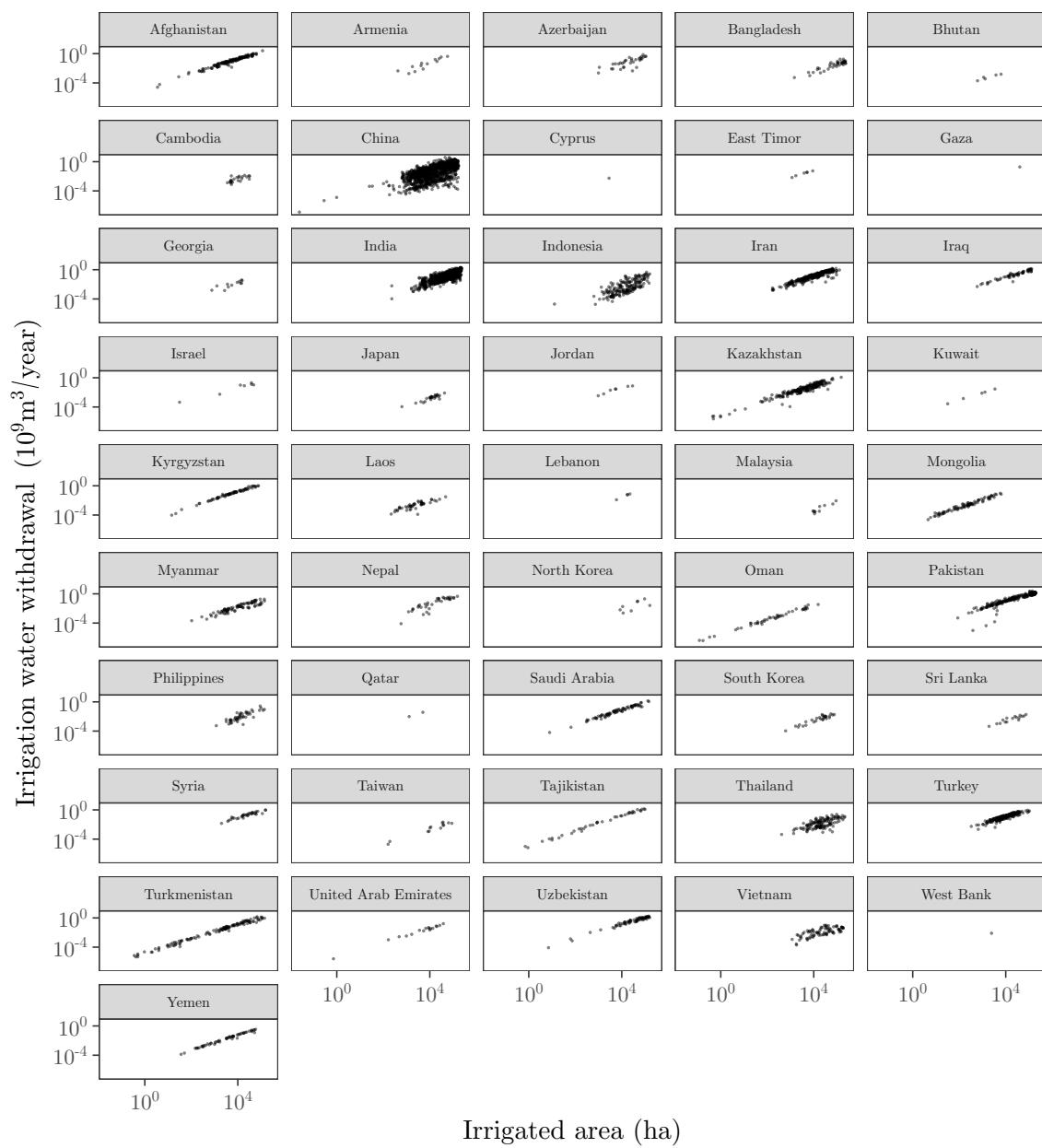
### WaterGap



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

## Asia

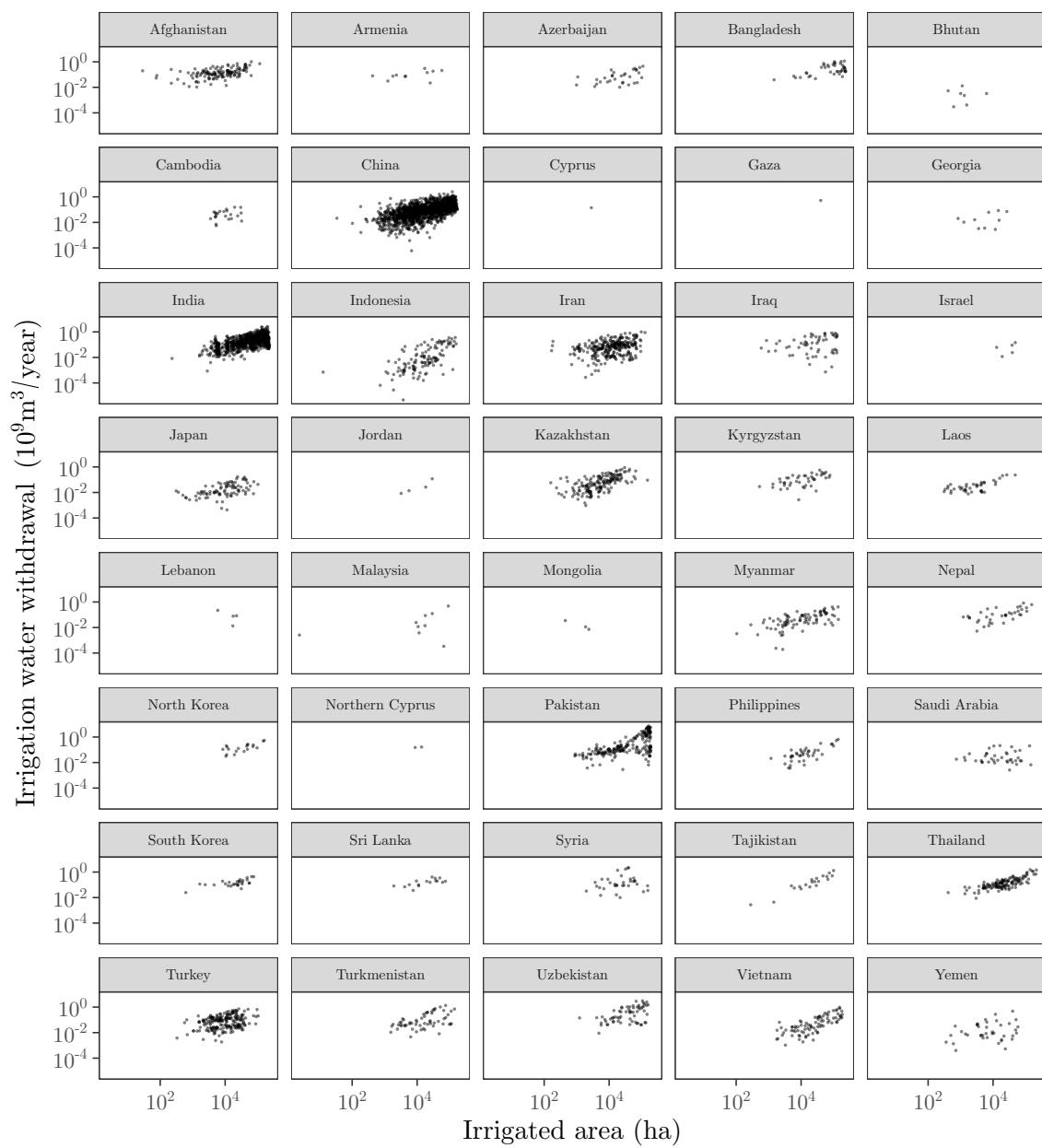
CLM45



```
##  
## $Asia$DBHM
```

## Asia

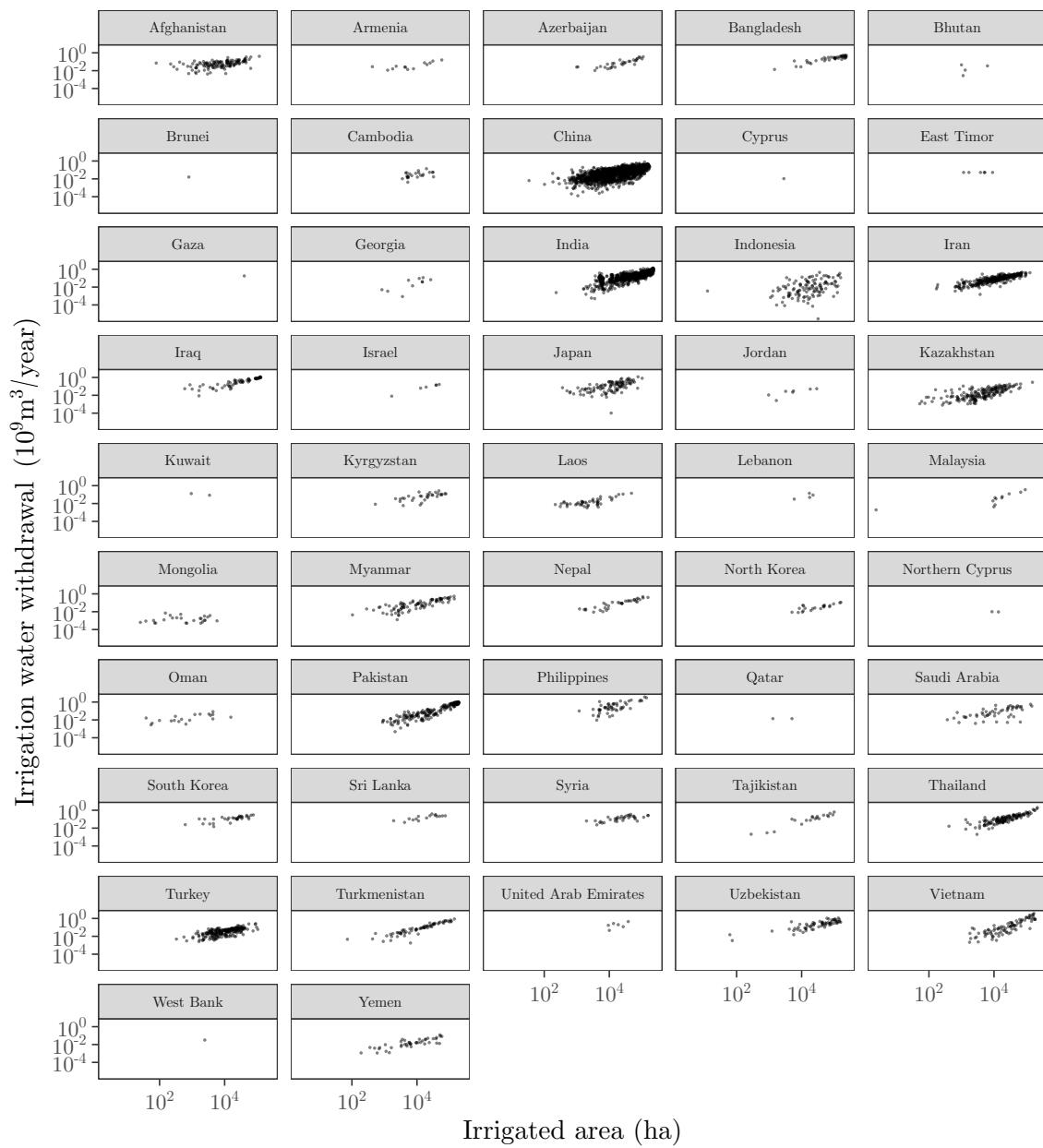
### DBHM



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

# Asia

H08

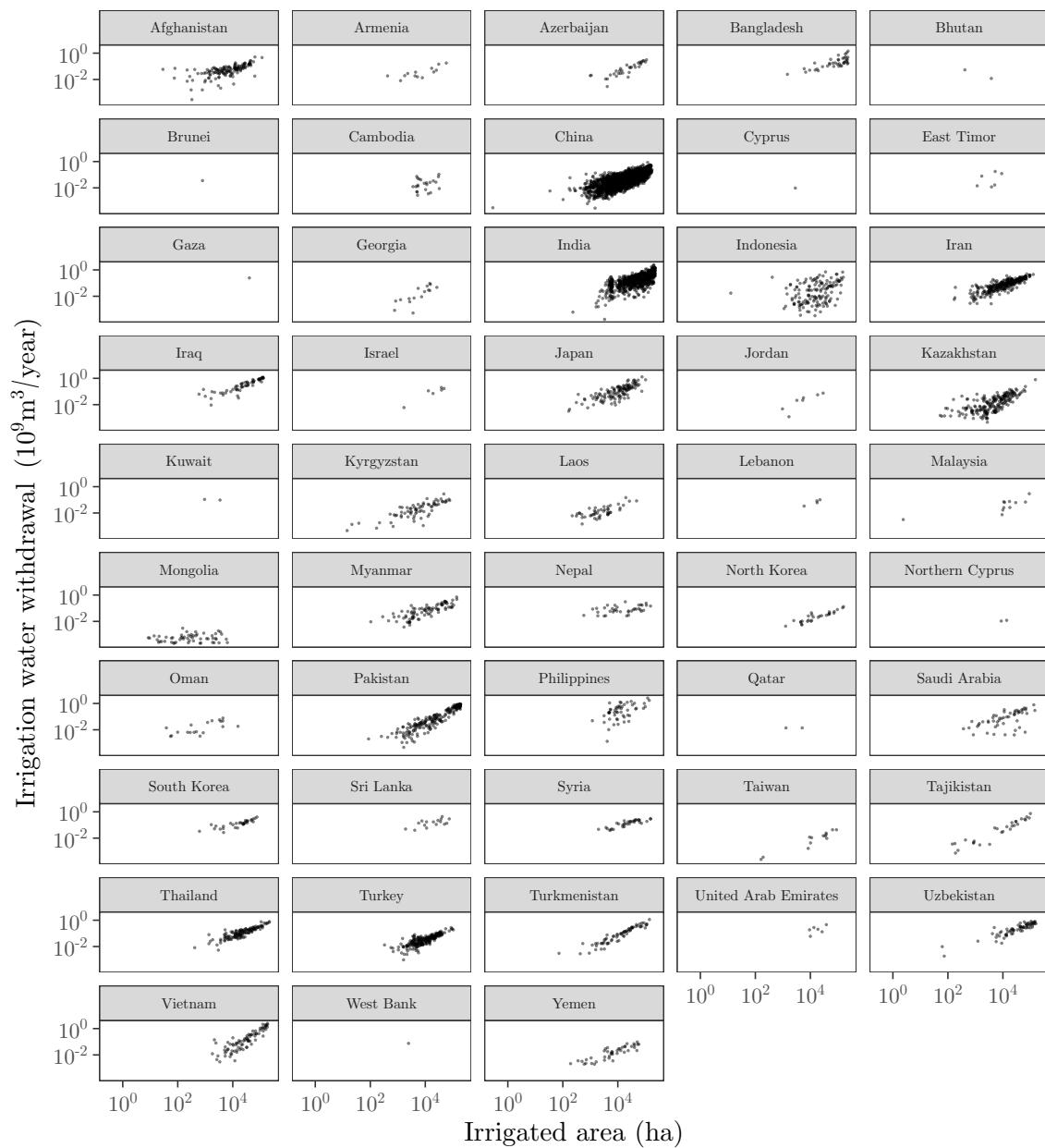


##

## \$Asia\$LPJmL

## Asia

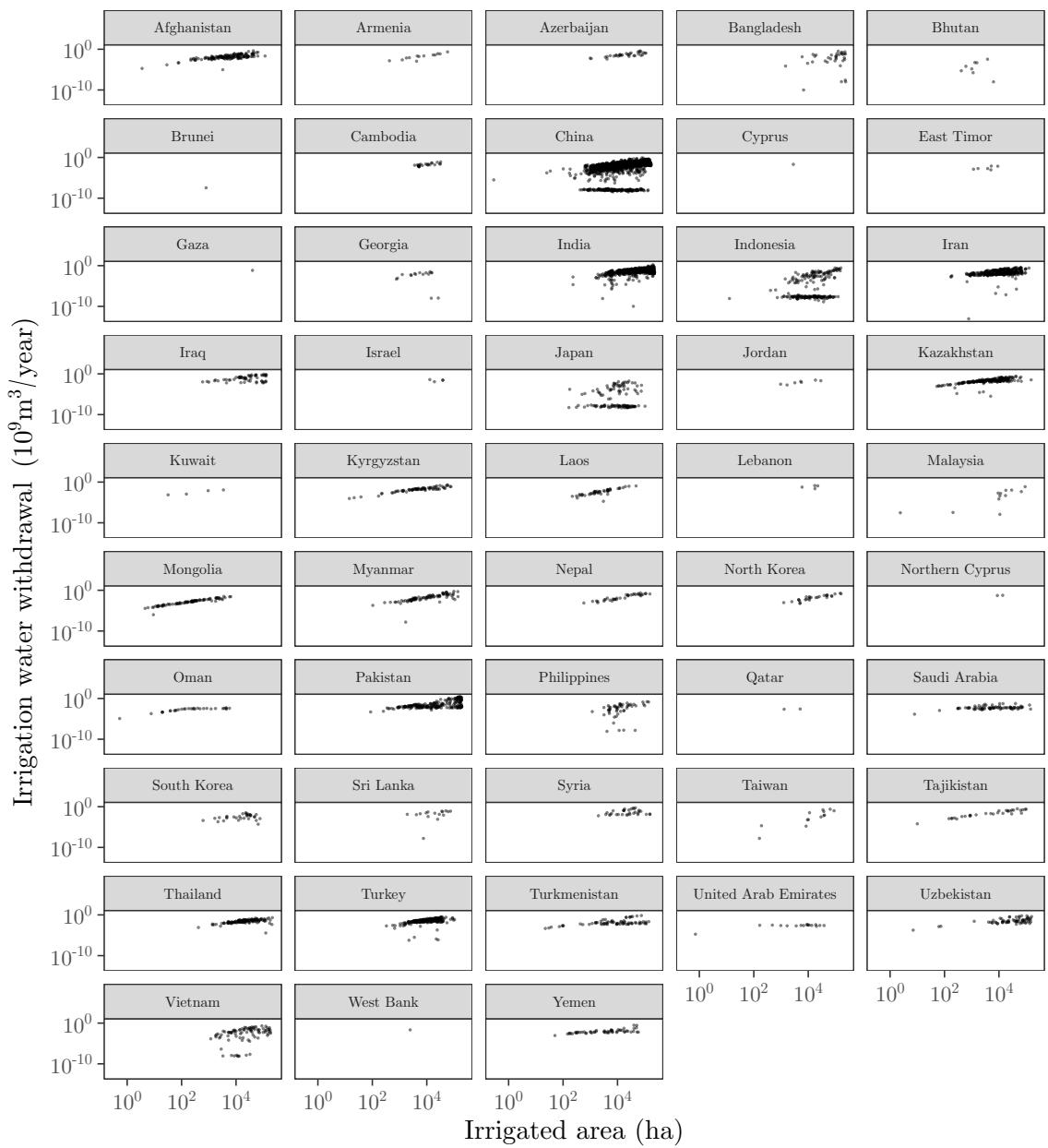
LPJmL



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

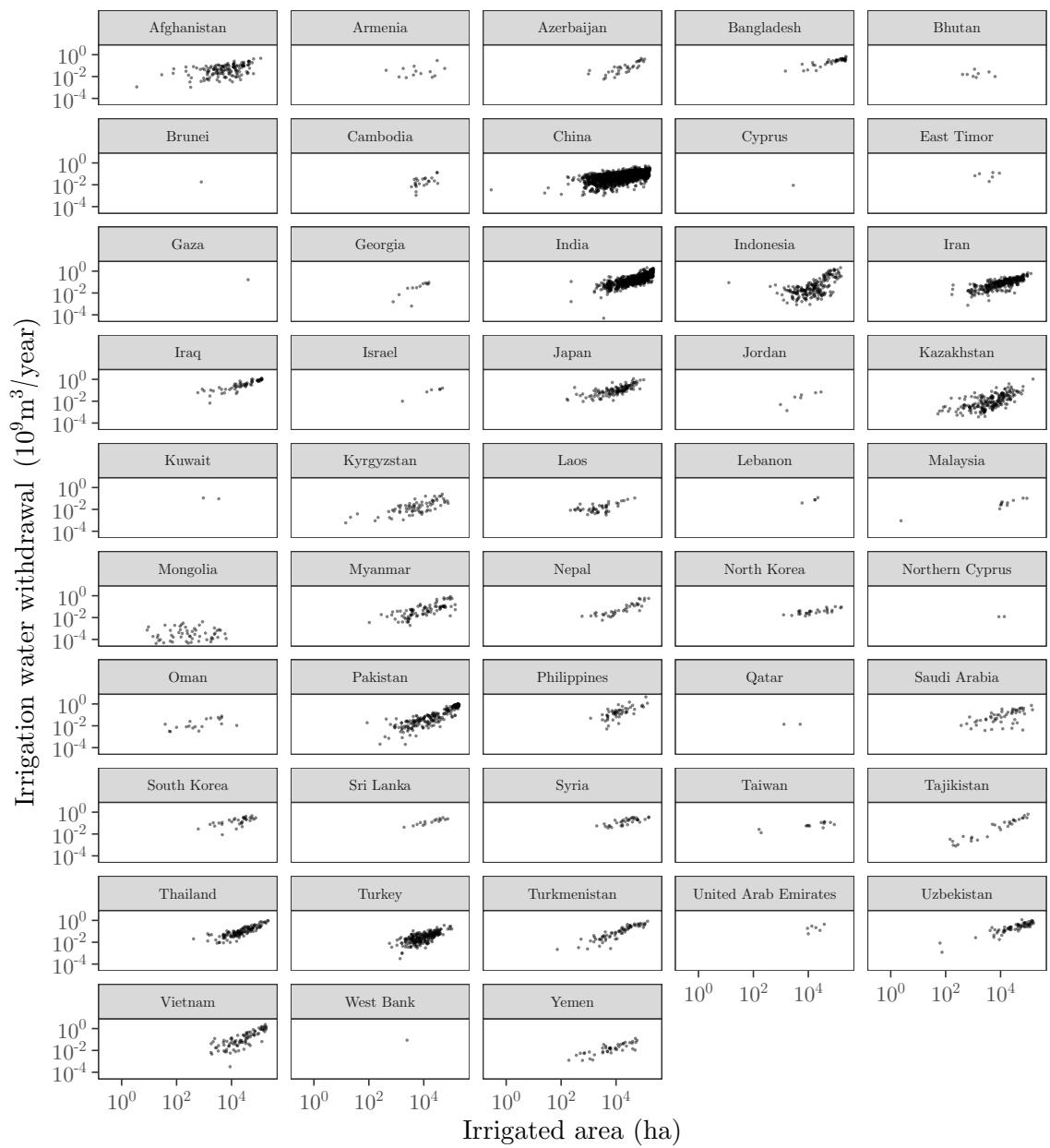
## Asia

MPI-HM



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

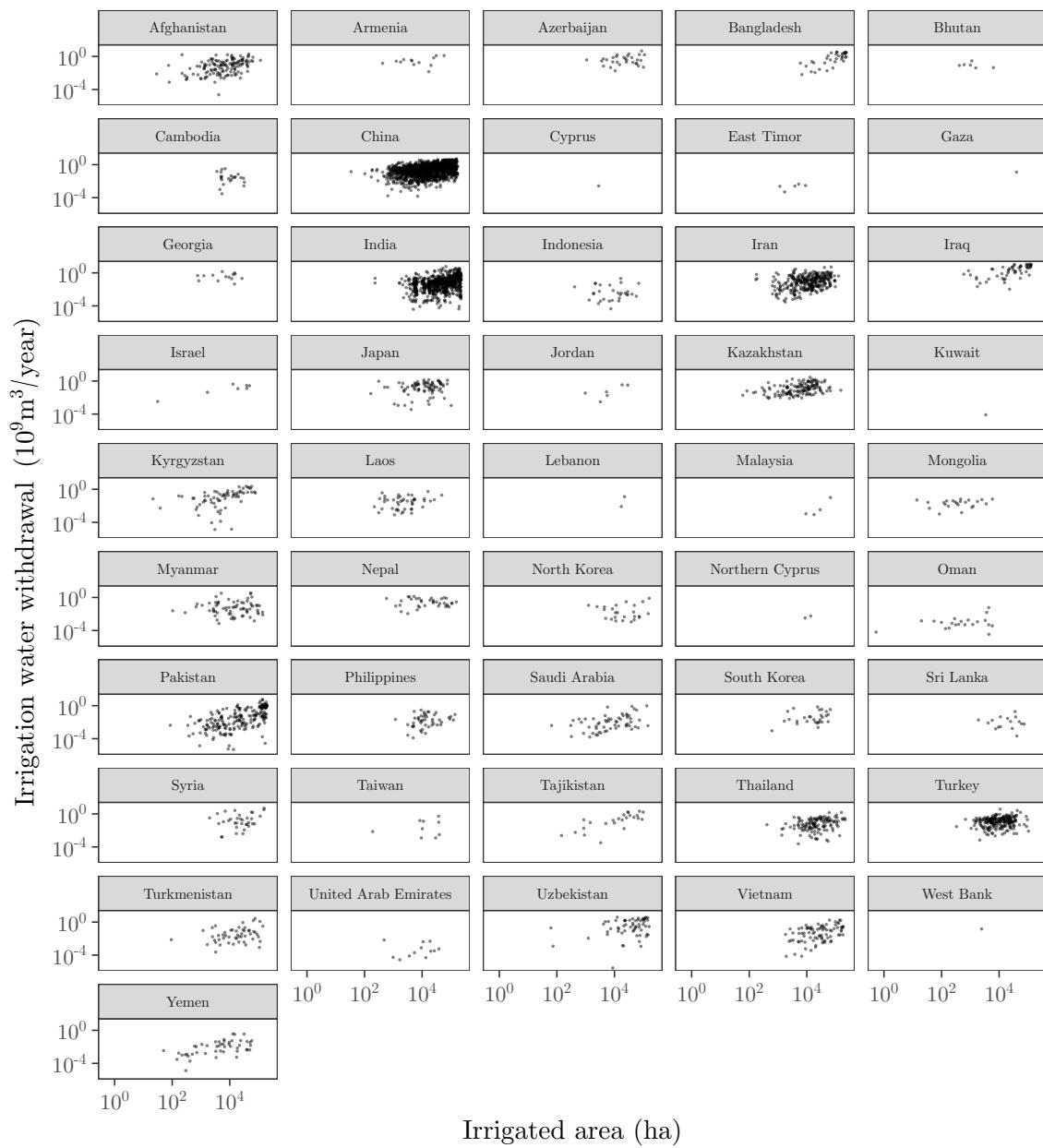
Asia  
PCR-GLOBWB



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

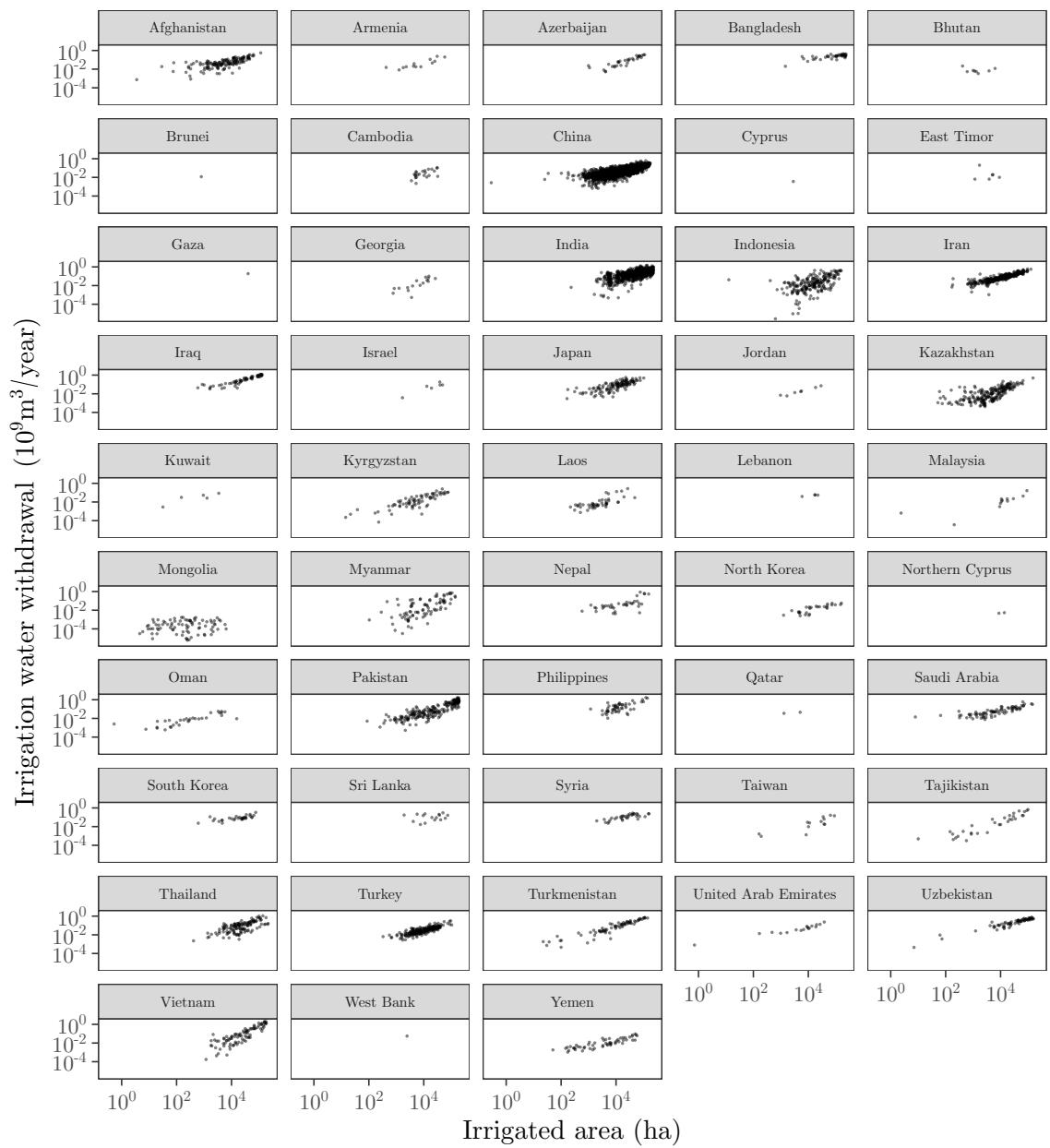
## Asia

### VIC



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

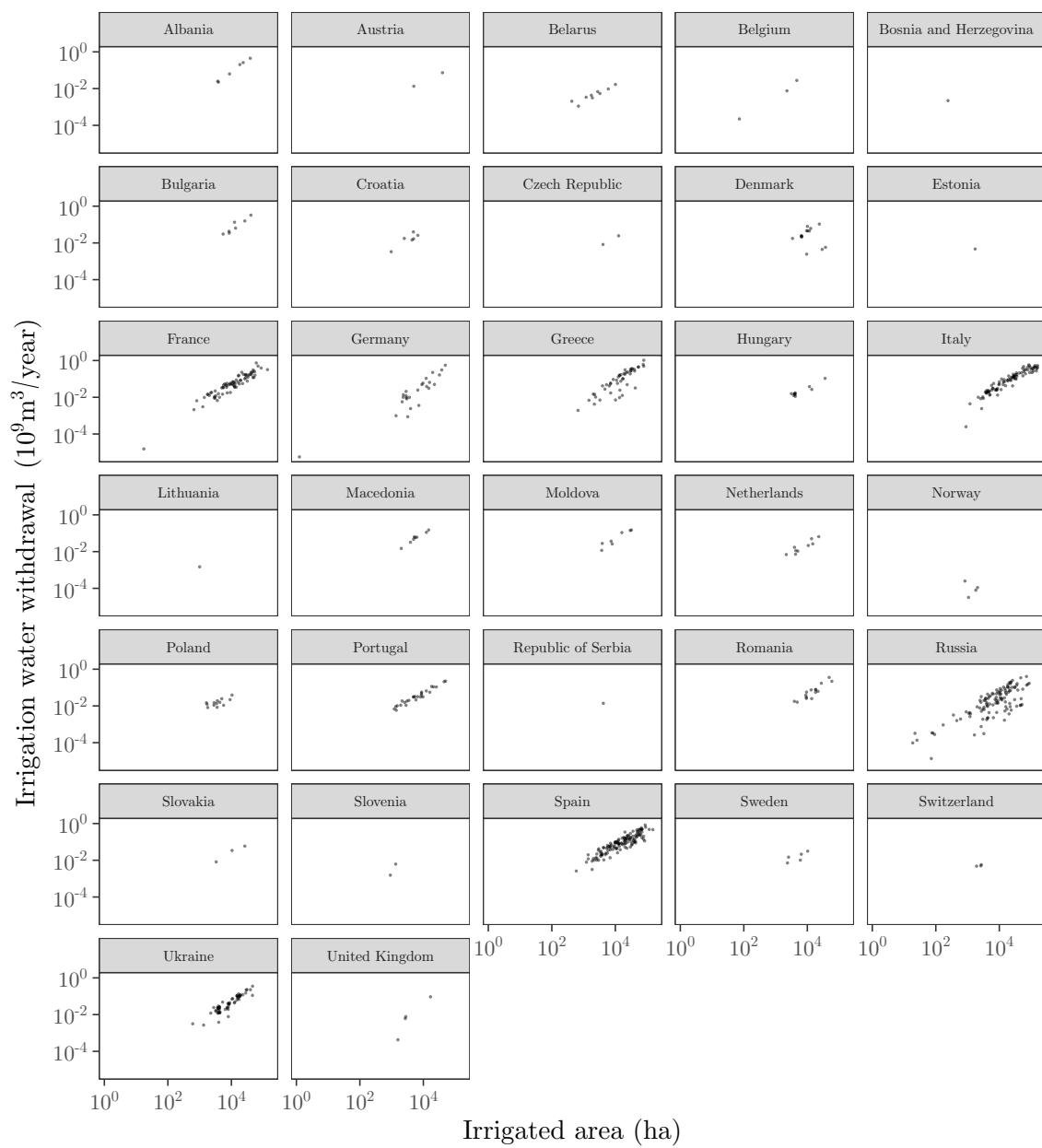
## Asia WaterGap



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

## Europe

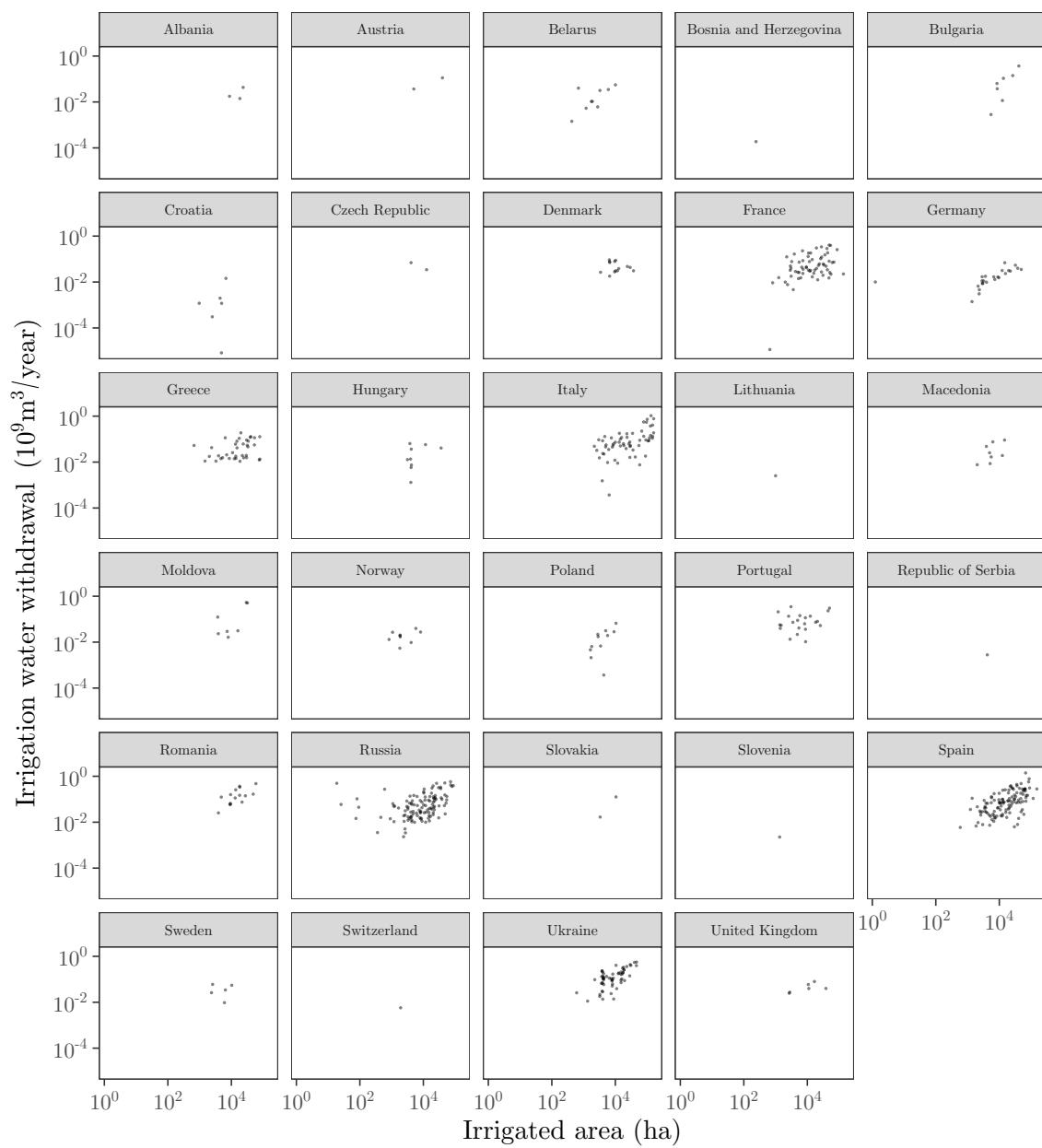
CLM45



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

## Europe

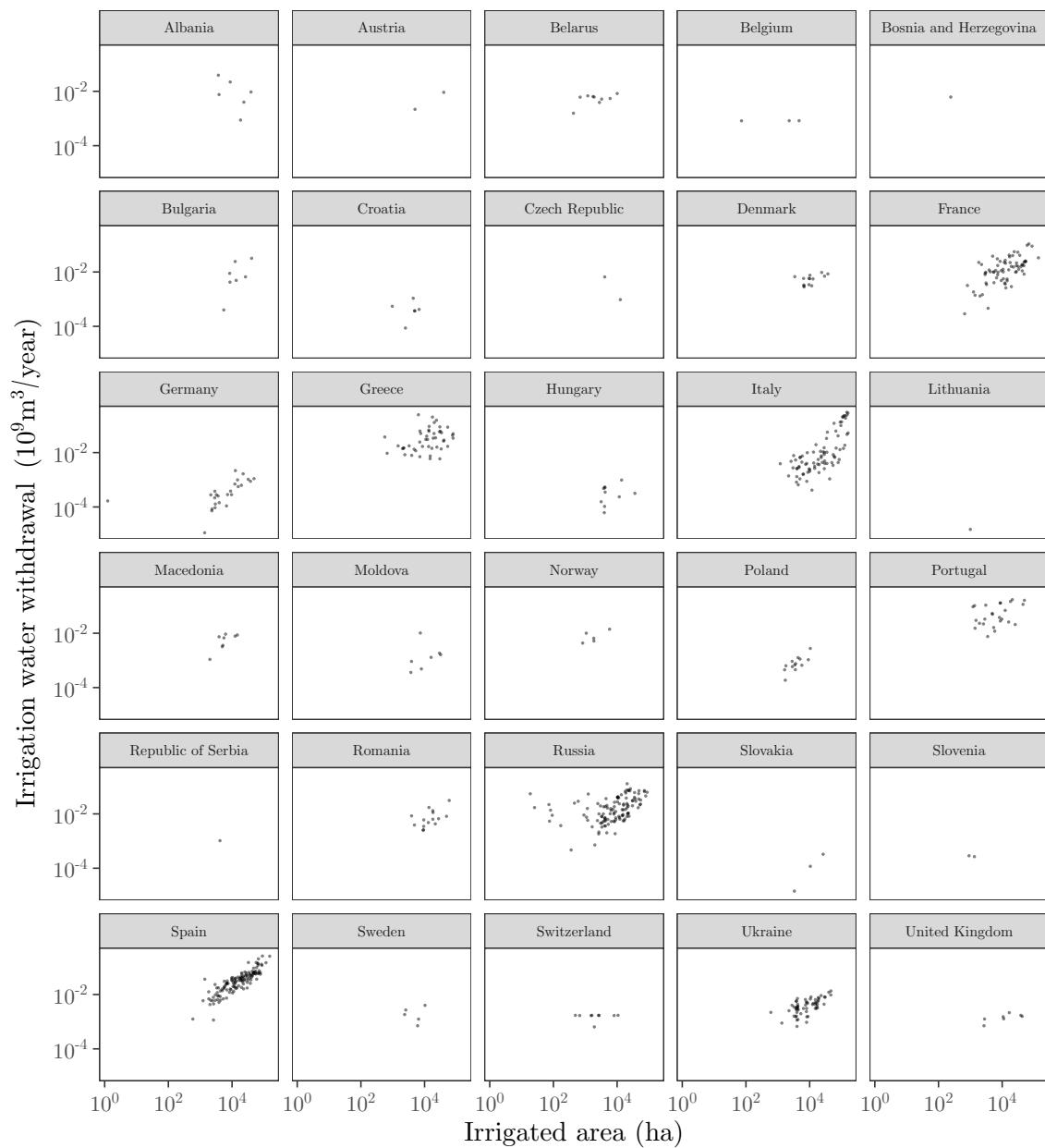
### DBHM



```
##  
## $Europe$H08
```

## Europe

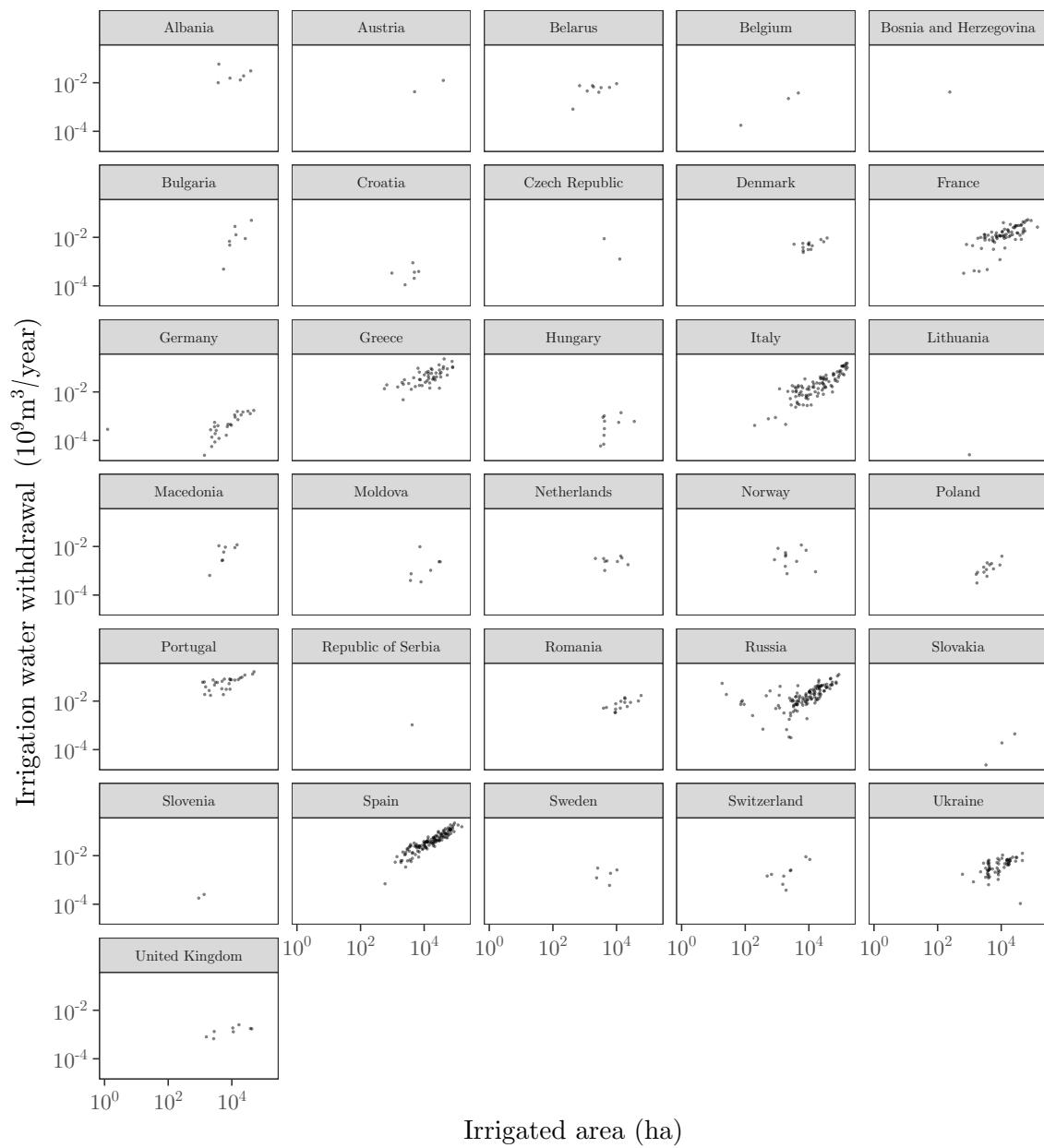
H08



```
##  
## $Europe$LPJmL
```

## Europe

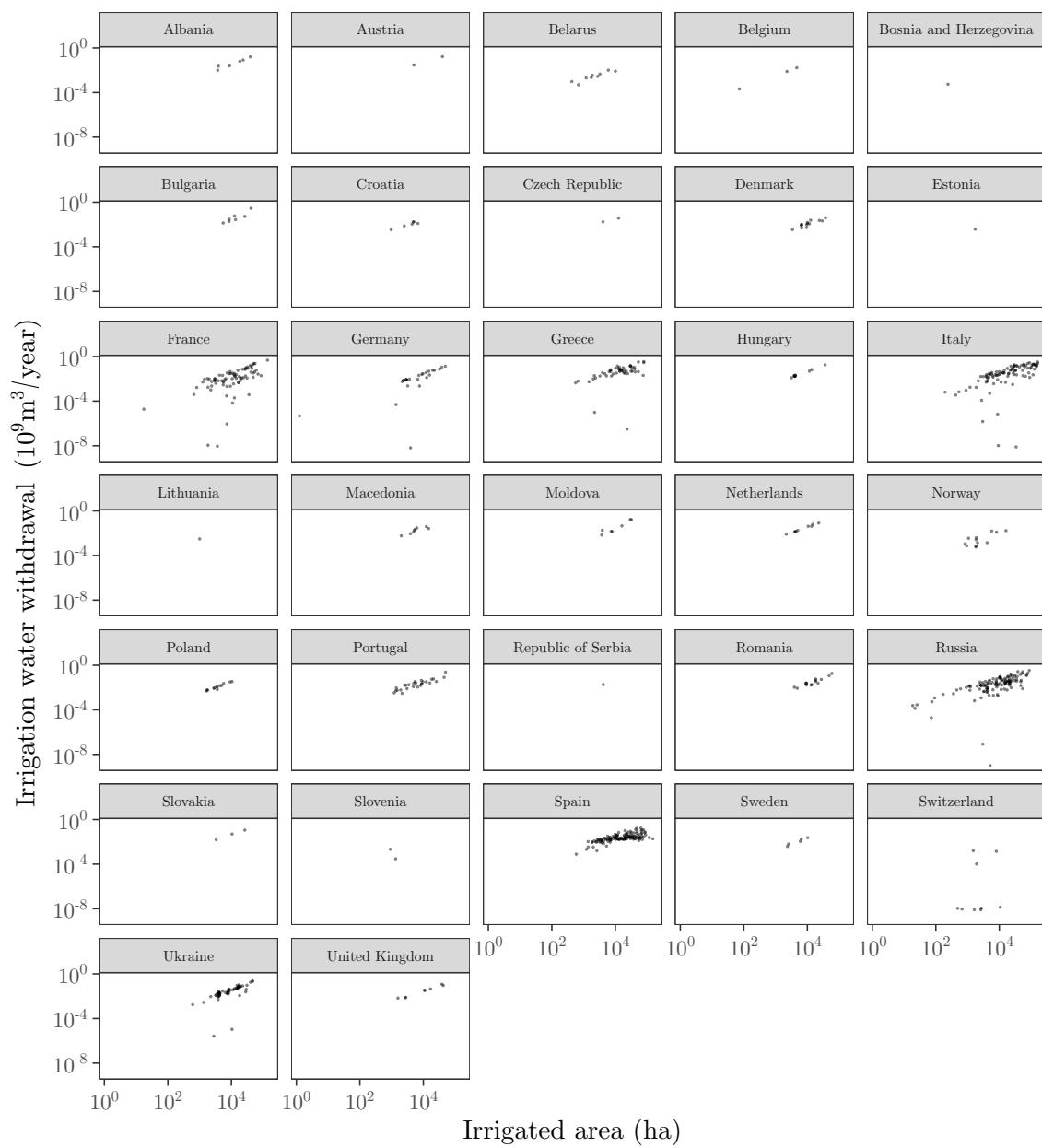
LPJmL



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

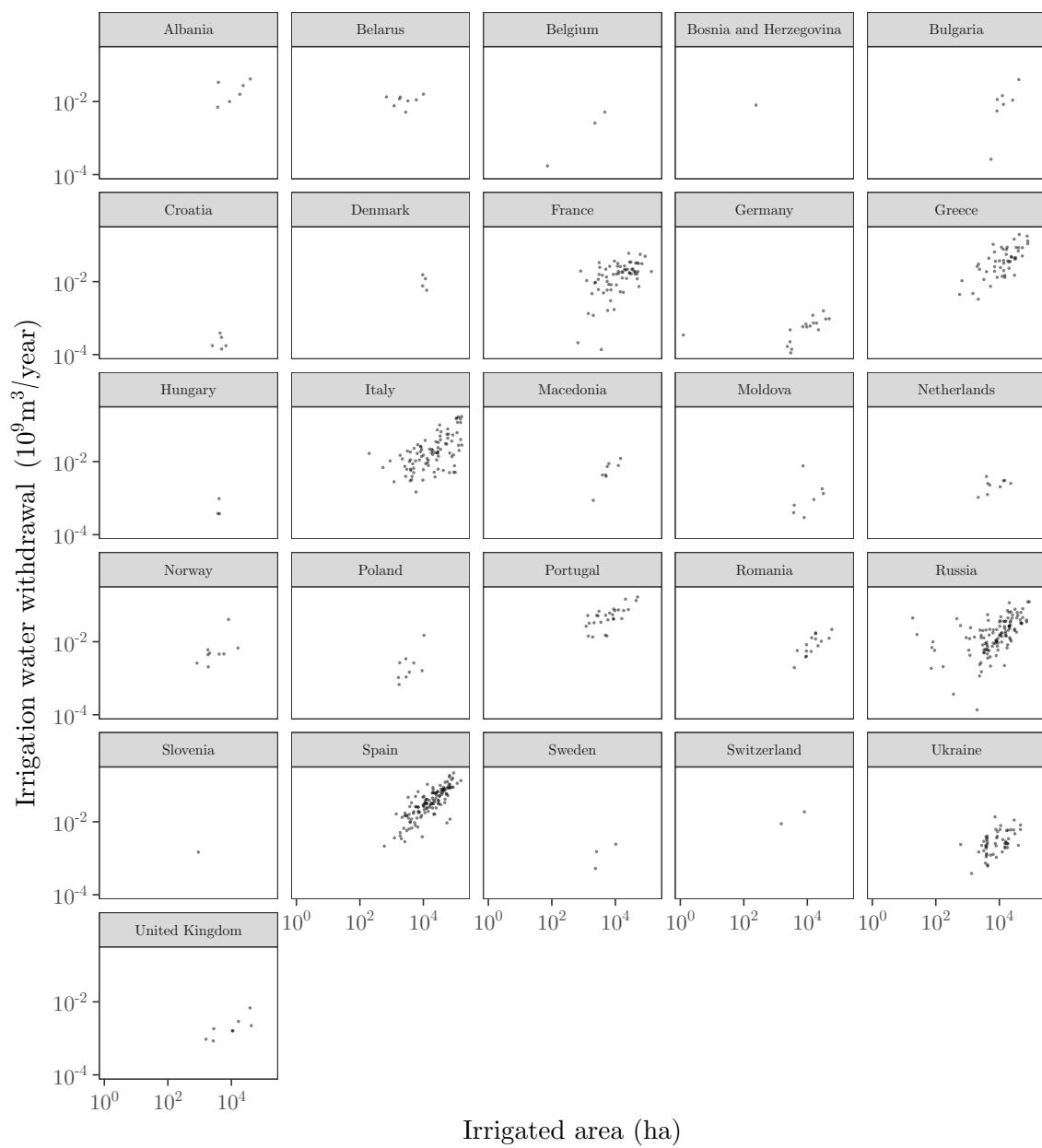
## Europe

MPI-HM



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

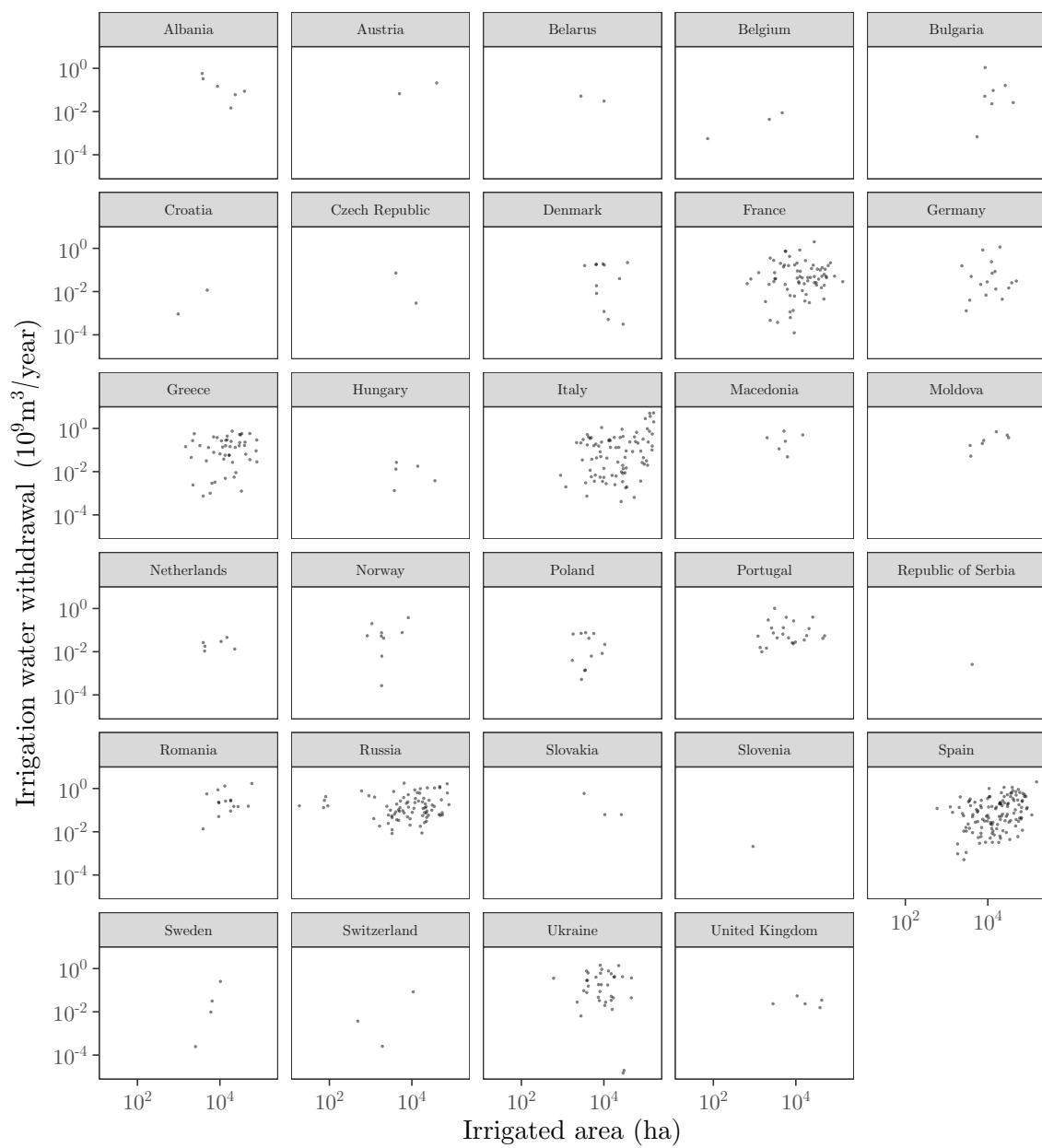
Europe  
PCR-GLOBWB



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

## Europe

### VIC

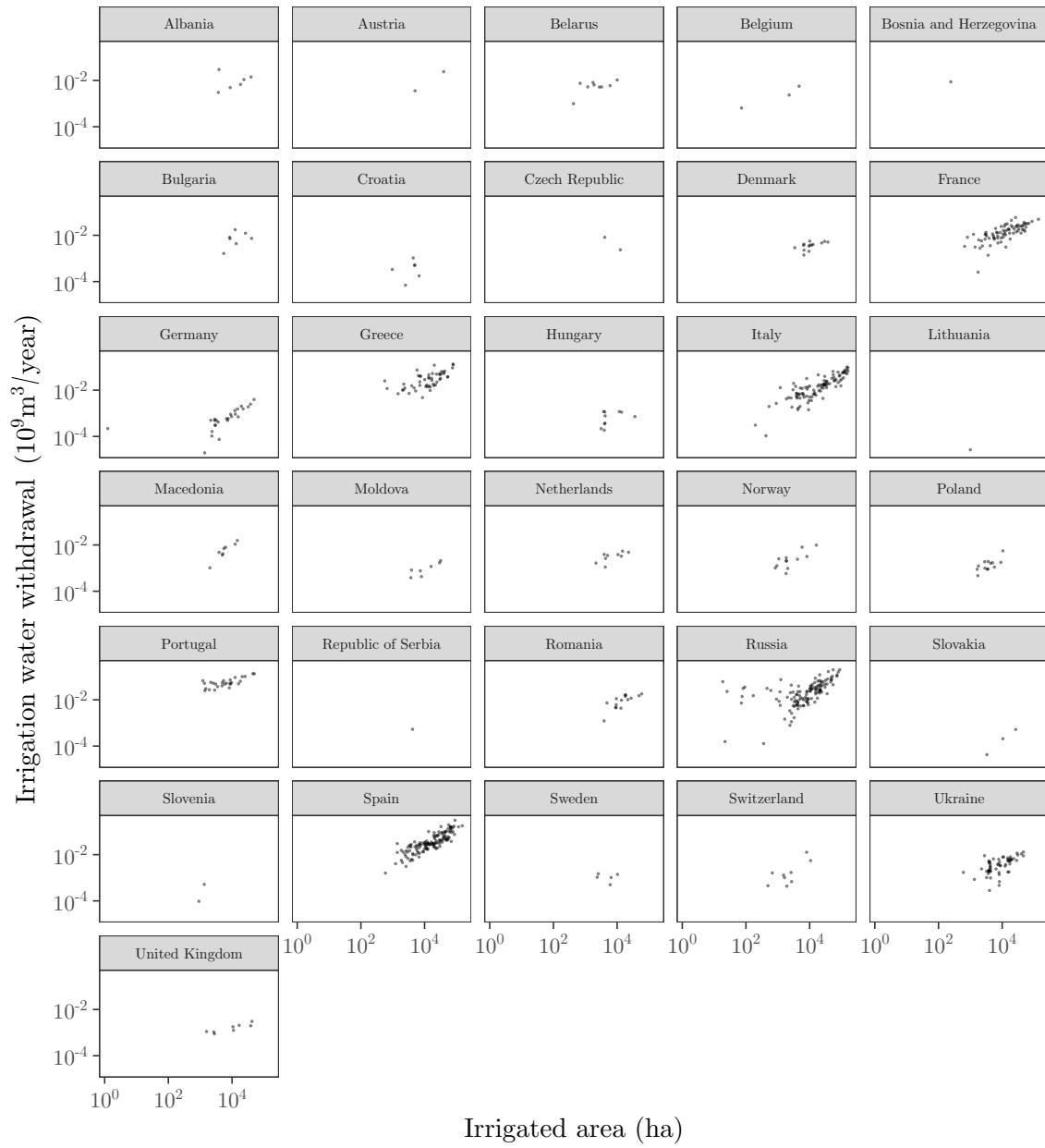


##

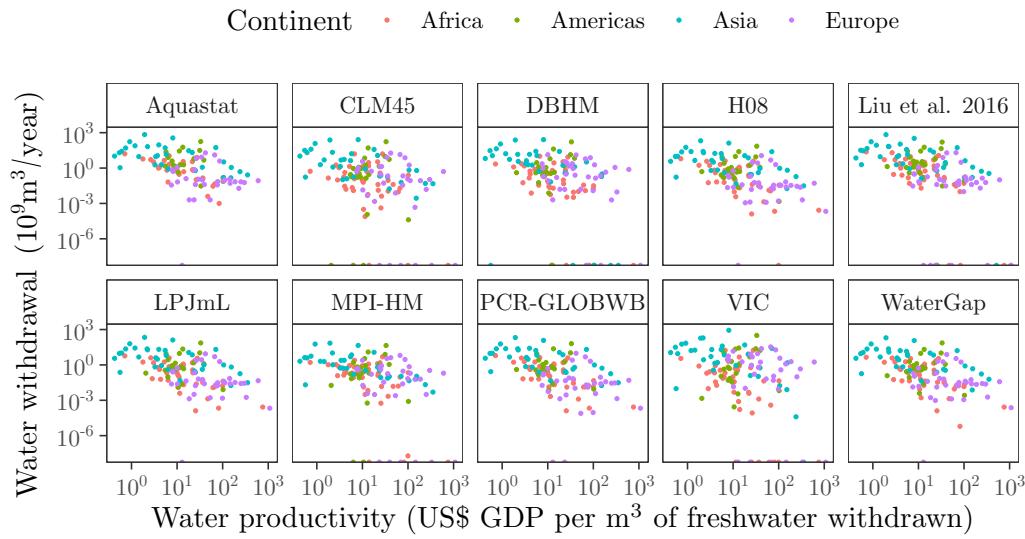
## \$Europe\$WaterGap

## Europe

### WaterGap



```
## Warning in countrycode(wp[, Country], origin = "country.name", destination = "un"): Some val
## Warning in countrycode(wp[, Country], origin = "country.name", destination = "continent"): S
## Warning: Transformation introduced infinite values in continuous y-axis
```



## 13 Session information

```
# SESSION INFORMATION ----

sessionInfo()

## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/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_0.4.10  ggrepel_0.9.1    rsample_0.0.9    robustbase_0.93-7
## [5] gridExtra_2.3    forcats_0.5.1    stringr_1.4.0    dplyr_1.0.5
## [9] purrrr_0.3.4     readr_1.4.0     tidyverse_1.3.0  benchmarkme_1.0.7
## [13] ggplot2_3.3.3    tidyverse_1.3.0  broom_0.7.6     cowplot_1.1.1
## [17] naniar_0.6.0     broom_0.7.6     ggridges_0.5.3   mice_3.13.0
## [21] lmtest_0.9-38    zoo_1.8-9      sandwich_3.0-0   mvoutlier_2.0.9
## [25] sgeostat_1.0-27  complmrob_0.7.0 doParallel_1.0.16 iterators_1.0.13
## [29] foreach_1.5.1    MASS_7.3-53.1   scales_1.1.1    boot_1.3-27
## [33] IDPmisc_1.1.20   countrycode_1.2.0 rworldmap_1.3-6 sp_1.4-5
```

```

## [37] ncdf4_1.17           sensobol_1.0.1      data.table_1.14.0
##
## loaded via a namespace (and not attached):
## [1] readxl_1.3.1          backports_1.2.1    spam_2.6-0
## [4] plyr_1.8.6             splines_4.0.3     listenv_0.8.0
## [7] fda_5.1.9              digest_0.6.27    htmltools_0.5.1.1
## [10] fansi_0.4.2            magrittr_2.0.1   cluster_2.1.1
## [13] ks_1.12.0              hddrcde_3.4     openxlsx_4.2.3
## [16] globals_0.14.0          modelr_0.1.8    tikzDevice_0.12.3.1
## [19] fds_1.8                colorspace_2.0-0 rvest_1.0.0
## [22] rrcov_1.5-5            haven_2.3.1     rbibutils_2.1
## [25] xfun_0.22              crayon_1.4.1    RCurl_1.98-1.3
## [28] jsonlite_1.7.2          survival_3.2-10 glue_1.4.2
## [31] gtable_0.3.0            car_3.0-10     kernlab_0.9-29
## [34] prabclus_2.3-2          DEoptimR_1.0-8  maps_3.3.0
## [37] abind_1.4-5              VIM_6.1.0       mvtnorm_1.1-1
## [40] DBI_1.1.1              GGally_2.1.1   Rcpp_1.0.6
## [43] sROC_0.1-2              laeken_0.5.1   foreign_0.8-81
## [46] proxy_0.4-25             mclust_5.4.7   dotCall64_1.0-1
## [49] stats4_4.0.3             vcd_1.4-8     truncnorm_1.0-8
## [52] httr_1.4.2              RColorBrewer_1.1-2 fpc_2.2-9
## [55] modeltools_0.2-23        ellipsis_0.3.1 rainbow_3.6
## [58] pkgconfig_2.0.3           reshape_0.8.8  NADA_1.6-1.1
## [61] flexmix_2.3-17            nnet_7.3-15    dbplyr_2.1.1
## [64] utf8_1.2.1              tidyselect_1.1.0 rlang_0.4.10
## [67] munsell_0.5.0             cellranger_1.1.0 tools_4.0.3
## [70] cli_2.4.0                generics_0.1.0  ranger_0.12.1
## [73] pls_2.7-3                evaluate_0.14 cvTools_0.3.2
## [76] yaml_2.2.1              fs_1.5.0       knitr_1.31
## [79] filehash_2.4-2             zip_2.1.1     visdat_0.5.3
## [82] future_1.21.0             xml2_1.3.2   rstudioapi_0.13
## [85] compiler_4.0.3             curl_4.3      e1071_1.7-6
## [88] zCompositions_1.3.4        reprex_2.0.0  robCompositions_2.3.0
## [91] pcaPP_1.9-73              stringi_1.5.3 fields_11.6
## [94] lattice_0.20-41            Matrix_1.3-2  vctrs_0.3.7
## [97] furrr_0.2.2              pillar_1.5.1  lifecycle_1.0.0
## [100] Rdpack_2.1.1             bitops_1.0-6  maptools_1.1-1
## [103] R6_2.5.0                 KernSmooth_2.23-18 rio_0.5.26
## [106] parallelly_1.24.0         codetools_0.2-18 benchmarkmeData_1.0.4
## [109] assertthat_0.2.1           withr_2.4.1   diptest_0.75-7
## [112] hms_1.0.0                 class_7.3-18 rmarkdown_2.7
## [115] carData_3.0-4              lubridate_1.7.10

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

## Machine:
## [1] "Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz"

```

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

## Num cores:
## [1] 8

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

## Num threads:
## [1] 16

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

## RAM:
## 34.4 GB
```

## 14 References