

The Achilles heel of Global Hydrological Models

R code

Arnald Puy

Contents

1 Presentation	2
2 Preliminary functions	2
3 Read in datasets	3
3.1 Loading functions	3
3.2 Load GHM and FAO-based datasets	5
3.3 Load FAO-GMIA dataset	8
3.4 Plot FAO-GMIA against irrigation water withdrawal	8
4 Missing values	11
5 Regressions	13
6 Lookup table	20
7 Run the model	20
7.1 Define settings	20
7.2 Sample matrix	21
7.3 The model	22
8 Uncertainty analysis	23
9 Sensitivity analysis	26
10 Session information	33
References	35

1 Presentation

This document presents the workflow behind the paper “The Achilles heel of Global Hydrological Models”, by A. Puy, S. Lo Piano, and A. Saltelli, whose abstract is the following:

Global Hydrological Models (GHM) compute irrigation water demands based on sub-models on crop types and calendars, soil evapotranspiration processes, irrigation efficiencies, weather data and irrigated areas. Here we demonstrate that such design is convoluted as irrigation water withdrawals are largely determined by irrigated areas only. We show that modeling water withdrawals as a function of irrigated areas eases the exploration of model uncertainties and allows the production of statistics on the model output, something unattainable by GHM due to their computational demands. The results suggest that GHM need to improve the balance between complexity and error and explicitly integrate uncertainties to provide more robust insights on agrarian strategies 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).

2 Preliminary functions

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

```
# 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"))

# SET CHECKPOINT -----
dir.create(".checkpoint")
```

```

library("checkpoint")

checkpoint("2020-05-17",
           R.version ="3.6.3",
           checkpointLocation = getwd())

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

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

}

```

3 Read in datasets

In this section we will load all datasets and products required for the analysis. We will retrieve irrigation water withdrawal data from six Gobal Hydrological Models (GHM) (PCR-GLOBWB, H08, LPMjL, WaterGap, MPI-HM, DBHM) and from two FAO-based datasets (Aquastat and Aquastat without missing values).

For PCR-GLOBWB, H08, LPMjL and WaterGap we will use the .nc files compiled by Huang et al. (2018). For DBHM (Tang et al. 2007) and MPI-HM (Stacke and Hagemann 2012), we will rely on the .nc files available in the Inter-Sectoral Impact Model Intercomparison Project (ISIMIP) webpage, <https://www.isimip.org>.

For the FAO-based datasets, we use the irrigation water withdrawal values reported by Aquastat (Frenken and Gillet 2012). We also use the dataset by Liu et al. (2016), which is similar to Aquastat's but without missing values.

3.1 Loading functions

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

`country_code` will unify the country names and provide each country with its United Nations code to avoid any confusion during the processing of the data.

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

`get_nc_data` is tailored to specifically extract irrigation water withdrawal data at the country level from the datasets compiled by Huang et al. (2018). Note that inside the function there are calls to `country_code` and `coors2country` to transform each longitude and latitude value to a country name.

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

```
# CREATE FUNCTIONS -----
# Function to obtain UN code, Continent and Country names
country_code <- function(dt) {
  dt[, `:=` (Code = countrycode(dt[, Country],
                                origin = "country.name",
                                destination = "un"),
             Continent = countrycode(dt[, Country],
                                origin = "country.name",
                                destination = "continent"))]
  dt[, Country:= countrycode(dt[, Code],
                                origin = "un",
                                destination = "country.name")]
  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")
  water <- rowSums(ww[, 469:ncol(ww)]) # Obtain year values for 2010 only
  ww.df <- data.frame(cbind(lon, lat, water))
  countries <- coords2country(ww.df[1:nrow(ww.df), 1:2])
  df <- cbind(countries, ww.df)
  setDT(df)
  final <- df[, .(Water.Withdrawn = sum(water)), countries]
  setnames(final, "countries", "Country")
```

```

country_code(final)
out <- na.omit(final[order(Continent)])
out[, Water.Withdrawn:= Water.Withdrawn / 1000] # From mm to m
return(out)
}

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

```

3.2 Load GHM and FAO-based datasets

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

We show at the end of the code chunk that MPI-HM produced an outlier for Gabon. The difference with the values outputted by the other GHM is very large and suggests that it is actually a miscalculation. To prevent this outlier from biasing the computations, we transformed it to `NA` in

order to impute it later on.

```
# READ IN DATASETS ON IRRIGATION WATER WITHDRAWAL -----  
  
# Define vector to reorder columns  
cols_order <- c("Continent", "Country", "Code", "Water.Dataset", "Water.Withdrawn")  
  
# FAO data (Table 4) -----  
# UNIT IS KM3/YEAR  
table4.tmp <- fread("table_4.csv", skip = 3, nrows = 167) %>%  
  .[, .(Country, Year, Water.withdrawal)] %>%  
  setnames(., "Water.withdrawal", "Water.Withdrawn")  
  
# Extract the selected years  
table4.dt <- country_code(table4.tmp[Year %in% 1999:2012])[  
  , Water.Dataset := "Aquastat"] [  
  , Year := NULL] %>%  
  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"] %>%  
  setcolororder(., cols_order)  
  
# Huang et al datasets -----  
names_nc_files <- c("withd_irr_lpjml.nc", "withd_irr_pcrglobwb.nc",  
  "withd_irr_h08.nc", "withd_irr_watergap.nc")  
out.nc <- mclapply(names_nc_files, function(x) get_nc_data(x), mc.cores = detectCores() * 0.75)  
names(out.nc) <- c("LPJml", "PCR-GLOBWB", "H08", "WaterGap")  
out.final <- rbindlist(out.nc, idcol = "Water.Dataset")  
  
# ISIMIP datasets -----  
files <- list("dbh_wfdei_nobc_hist_varsoc_co2_airrrww_global_monthly_1971_2010.nc",  
  "mpi-hm_miroc5_ewembi_picontrol_histsoc_co2_airrrww_global_monthly_1861_2005.nc")  
  
dname <- "airrrww"  
  
isimip.dt <- mclapply(files, function(x) open_nc_files(file = x, dname = dname),  
  mc.cores = detectCores() * 0.75)  
  
names(isimip.dt) <- c("DBHM", "MPI-HM")  
  
isimip.final <- rbindlist(isimip.dt, idcol = "Water.Dataset") %>%  
  setnames(., "countries", "Country") %>%
```

```

country_code(.) %>%
  na.omit()

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

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

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

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

```

Next we row-bind the GHM and FAO-based datasets, and check if there are duplicated countries resulting from a bad conversion of longitude/latitude to country names. We observed that there were only two repeated countries, Cyprus and Palestinian Territories. Since we aimed at obtaining only one irrigation water withdrawal value per country and dataset, we retrieved for these countries the mean irrigation water withdrawal values.

We finally produce the dataset containing GHM and FAO-based irrigation water withdrawal data: `water.dt`.

```

# CREATE THE FINAL IRRIGATION WATER WITHDRAWAL DATASET -----
water.dt <- rbind(liu.dt, table4.dt, GHM.dt)

# Check if there are duplicated countries
duplicated.countries <- water.dt[, .N, .(Country)][N > 8][, Country]

# Show duplicated countries
water.dt[Country %in% duplicated.countries]

```

	Continent	Country	Code	Water.Dataset	Water.Withdrawn
## 1:	Asia	Cyprus	196	Liu et al. 2016	0.160000000
## 2:	Asia	Palestinian Territories	275	Liu et al. 2016	0.250000000
## 3:	Asia	Cyprus	196	Aquastat	0.148000000
## 4:	Asia	Palestinian Territories	275	Aquastat	0.189000000
## 5:	Asia	Cyprus	196	LPJmL	0.016139367
## 6:	Asia	Cyprus	196	LPJmL	0.022694116
## 7:	Asia	Cyprus	196	PCR-GLOBWB	0.014070284

```

## 8:    Asia          Cyprus 196    PCR-GLOBWB 0.024366953
## 9:    Asia          Cyprus 196    H08        0.016676997
## 10:   Asia          Cyprus 196    H08        0.019444796
## 11:   Asia          Cyprus 196    WaterGap   0.006359497
## 12:   Asia          Cyprus 196    WaterGap   0.010202374
## 13:   Asia          Cyprus 196    DBHM      0.315209567
## 14:   Asia          Cyprus 196    DBHM      0.140490805
## 15:   Asia          Cyprus 196    MPI-HM    0.110097728
## 16:   Asia          Cyprus 196    MPI-HM    0.020111216
## 17:   Asia Palestinian Territories 275    LPJmL    0.245201230
## 18:   Asia Palestinian Territories 275    LPJmL    0.100188175
## 19:   Asia Palestinian Territories 275    PCR-GLOBWB 0.163368097
## 20:   Asia Palestinian Territories 275    PCR-GLOBWB 0.105801784
## 21:   Asia Palestinian Territories 275    H08        0.178804828
## 22:   Asia Palestinian Territories 275    H08        0.063406375
## 23:   Asia Palestinian Territories 275    WaterGap   0.187555342
## 24:   Asia Palestinian Territories 275    WaterGap   0.084385338
## 25:   Asia Palestinian Territories 275    DBHM      0.000000000
## 26:   Asia Palestinian Territories 275    DBHM      0.520364324
## 27:   Asia Palestinian Territories 275    MPI-HM    0.021298826
## 28:   Asia Palestinian Territories 275    MPI-HM    0.068825864
##     Continent          Country Code  Water.Dataset Water.Withdrawn
# Compute mean
water.dt <- water.dt[, .(Water.Withdrawn = mean(Water.Withdrawn)),
                      .(Water.Dataset, Country, Code, Continent)]

```

3.3 Load FAO-GMIA dataset

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

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

```

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

```

3.4 Plot FAO-GMIA against irrigation water withdrawal

In order to obtain the final dataset for the computations, we merge `water.dt` with `meier.dt`. We forced `water.dt` to have the same countries as `meier.dt` (186). Note that there are currently 193 countries recognized by the United Nations. If some GHM or FAO-based dataset within `water.dt` had no matching row in `meier.dt`, a NA was added. The method to impute these missing values is a source of uncertainty, and will be dealt with later on. For now, we only print the list of countries that include some missing irrigation water withdrawal value.

```

# MERGE DATASETS ----

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

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

## [1] "Bosnia and Herzegovina"
## [2] "Burma"
## [3] "Czech Republic"
## [4] "Iran (Islamic Republic of)"
## [5] "Korea, Republic of"
## [6] "Lao People's Democratic Republic"
## [7] "Libyan Arab Jamahiriya"
## [8] "Palestine"
## [9] "Republic of Moldova"
## [10] "Saint Lucia"
## [11] "Syrian Arab Republic"
## [12] "The former Yugoslav Republic of Macedonia"
## [13] "Trinidad and Tobago"
## [14] "United Republic of Tanzania"
## [15] "Viet Nam"
## [16] "Belize"
## [17] "Cape Verde"
## [18] "Croatia"
## [19] "Cuba"
## [20] "El Salvador"
## [21] "Grenada"
## [22] "Guyana"
## [23] "Haiti"
## [24] "Ireland"
## [25] "Lebanon"
## [26] "Luxembourg"
## [27] "Malaysia"
## [28] "Panama"
## [29] "Peru"
## [30] "Sudan"
## [31] "Suriname"
## [32] "Uruguay"
## [33] "Malta"
## [34] "Seychelles"
## [35] "Argentina"
## [36] "Canada"
## [37] "Portugal"

```

```
## [38] "Russia"
## [39] "Mauritius"
```

Finally, we plot irrigated areas against irrigation water withdrawals. As shown in Figure 1 below, they are clearly related and their trend suggests that the former might actually describe very much the behaviour of the latter. We used this plot as preliminary evidence for the overparametrization of GHM, since a single parameter drives most of the variance in their model output. Given the shape of the scatterplots, the relation between irrigated area and irrigation water withdrawal can very well be modeled with a simple linear regression model.

PLOT -----

```
full.dt %>%
  na.omit() %>%
  ggplot(., aes(Irrigated.Area, Water.Withdrawn,
                color = Continent)) +
  geom_point(size = 0.7) +
  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 = 4) +
  theme_AP() +
  theme(legend.position = "top",
        strip.text = element_text(size = 6.7))
```

Warning: Transformation introduced infinite values in continuous x-axis

Warning: Transformation introduced infinite values in continuous y-axis

In order to set the ground for the computations, which will be launched hereafter, we log-transform both irrigation water withdrawals and irrigated area values, and export all files created so far to .csv.

TRANSFORM DATASET -----

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

EXPORT FULL DATASET WITH MISSING VALUES -----

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

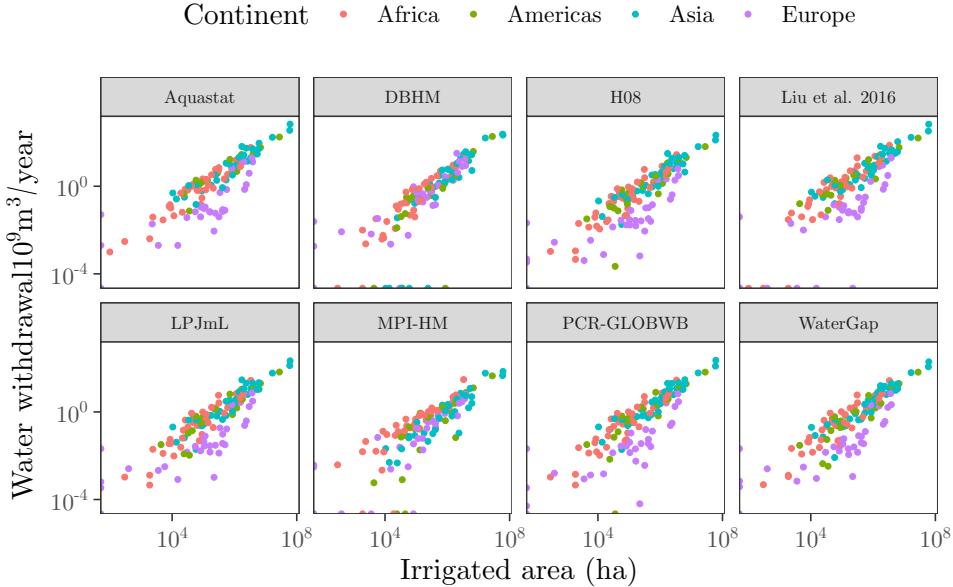


Figure 1: Scatterplots of irrigated areas reported by FAO-GMIA against irrigation water withdrawals. Each dot is a country.

4 Missing values

In multiple imputation methods, the proportion of missing values conditions the number of imputations required to obtain an appropriate estimate of the true missing value. Graham, Olchowski, and Gilreath (2007), for instance, recommended 20 iterations for 20%–30% missing data, and 40 iterations for 50% missing data. Which iteration to use in the regressions is also a source of uncertainty, which will be duly accounted for in due time. For now, we just calculate the fraction of missing values in each GHM and FAO-based dataset. Figure 2 below shows the results, and indicates that most datasets have c. 10-20% of missing values. The fact that one dataset shows a much larger proportion of missing values (Aquastat in the Americas, c. 40%) made us set the number of iterations at 40 to prevent loss of statistical power (see below).

```
# 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")
```

Now we impute the missing values for each continent and irrigation water withdrawal dataset. Given

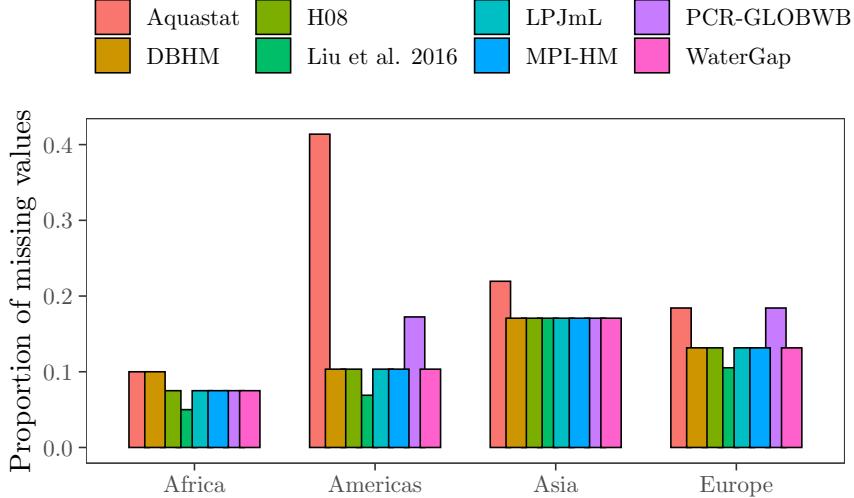


Figure 2: Proportion of missing values in irrigation water withdrawal per dataset.

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

```
# IMPUTATION OF MISSING VALUES -----
# Substitute Inf values for NA
for (j in 1:ncol(full.dt)) set(full.dt, which(is.infinite(full.dt[[j]])), j, NA)

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

##      Water.Dataset Country Code Continent Water.Withdrawn Irrigated.Area
## 1:          0        0    0           0          0            0
# Imputation settings
m.iterations <- 40
imputation.methods <- c("norm", "norm.boot", "norm.nob")

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

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

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

```

 .[, Data:= lapply(Datasets, function(x) rbindlist(x, idcol = "Iteration"))]

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

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

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

```

5 Regressions

The following code snippet computes all possible regressions for all combinations of irrigation water withdrawals datasets (8), imputation methods (3) and iterations (40). These three uncertain factors will be the triggers that we will use in our model to select the r^2 value (see below). In total, we have $8 * 3 * 40 = 960$ possible regressions and r^2 values in each continent. The result is the `regressions` dataset.

Then we extract `r.squared` values and other statistics from the `regressions` dataset, and create the `results` dataset. We do the same with the residuals, and create the `residuals` dataset.

```

# COMPUTE LINEAR REGRESSIONS -----
# Compute regressions in each combination
regressions <- full.imput %>%
  group_by(Continent, Water.Dataset, Imputation.Method, Iteration) %>%
  nest() %>%
  mutate(fit = map(.x = data, .f = ~lm(Water.Withdrawn ~ Irrigated.Area, data = .)),
    results = map(fit, glance),
    residuals = map(fit, augment))

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

```

```
# Extract residuals
residuals <- regressions %>%
  dplyr::select(Continent, Water.Dataset,
                Imputation.Method, Iteration, residuals) %>%
  unnest(residuals) %>%
  data.table()
```

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

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

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

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

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

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

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

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

```

    data.table()
})

out <- 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]]))]
}

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

# Compute quantiles
water.quantiles <- water.predicted[, .(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)]

```

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

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

```

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

```

```

    labs(y = expression(paste("Water withdrawal ", " ", "(", 10^9, m^3/year, "", "))),
      x = "") +
  coord_flip() +
  theme_AP()
}

## $Africa

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

```

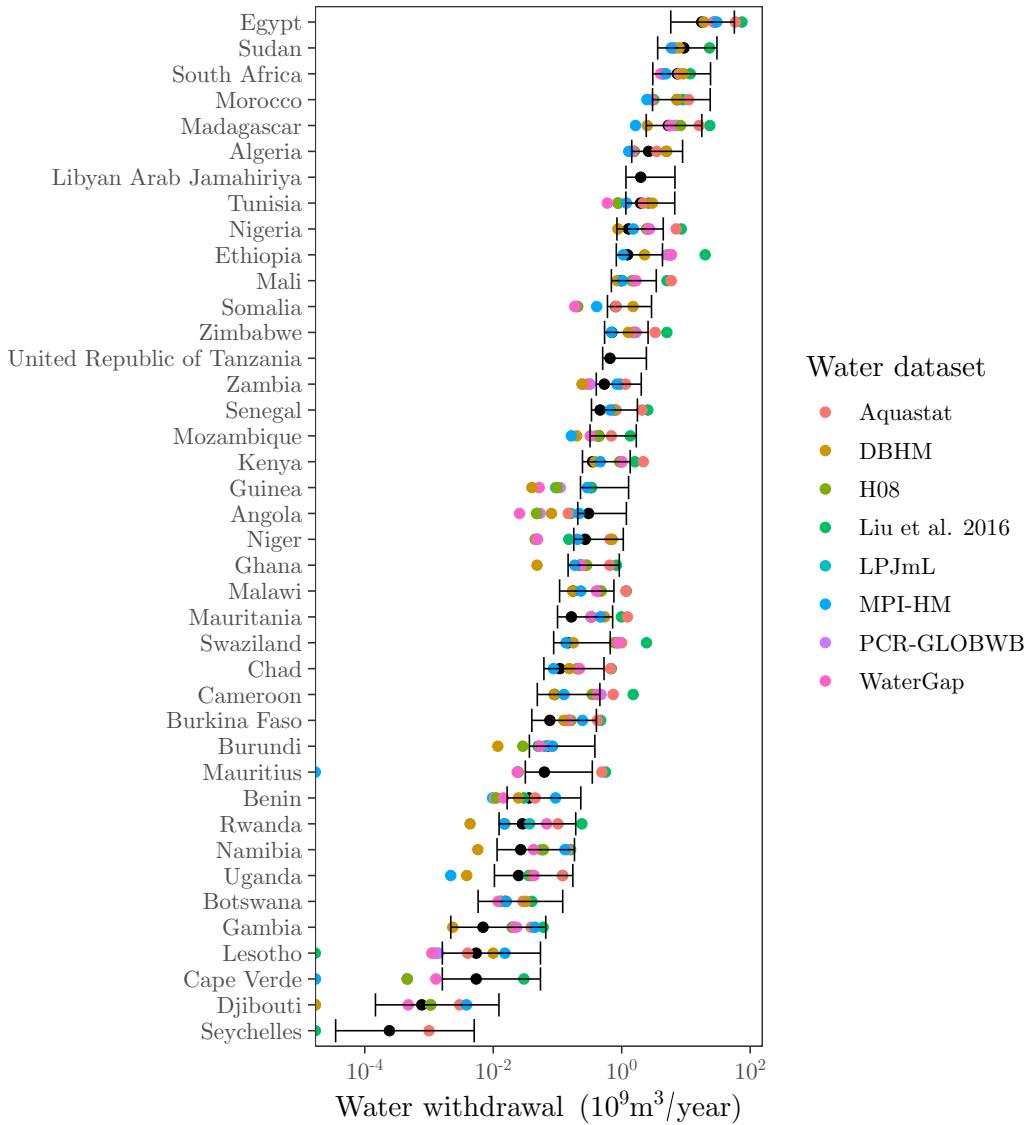


Figure 3: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models and FAO-based datasets.

```

## 
## $Americas

```

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

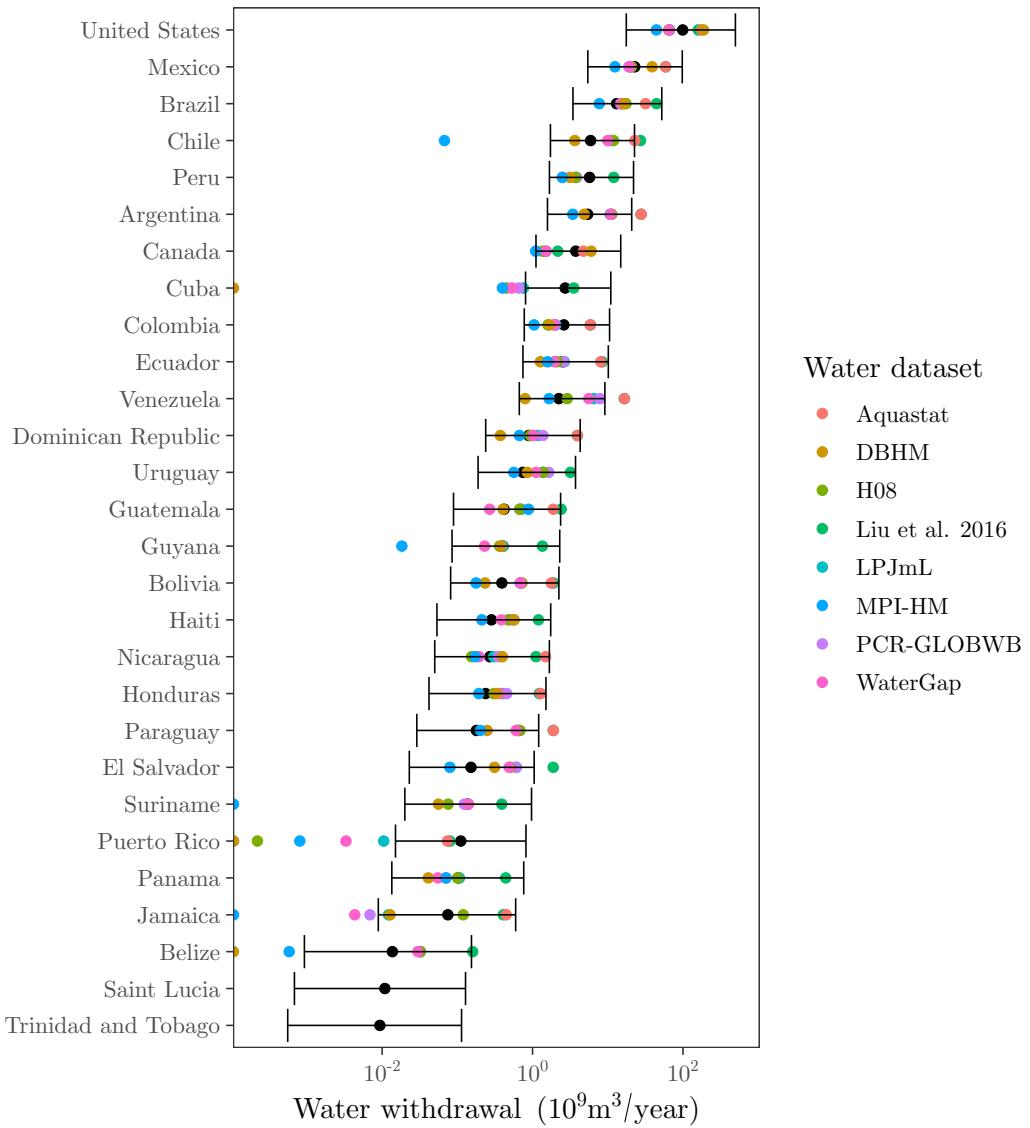


Figure 4: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models and FAO-based datasets.

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

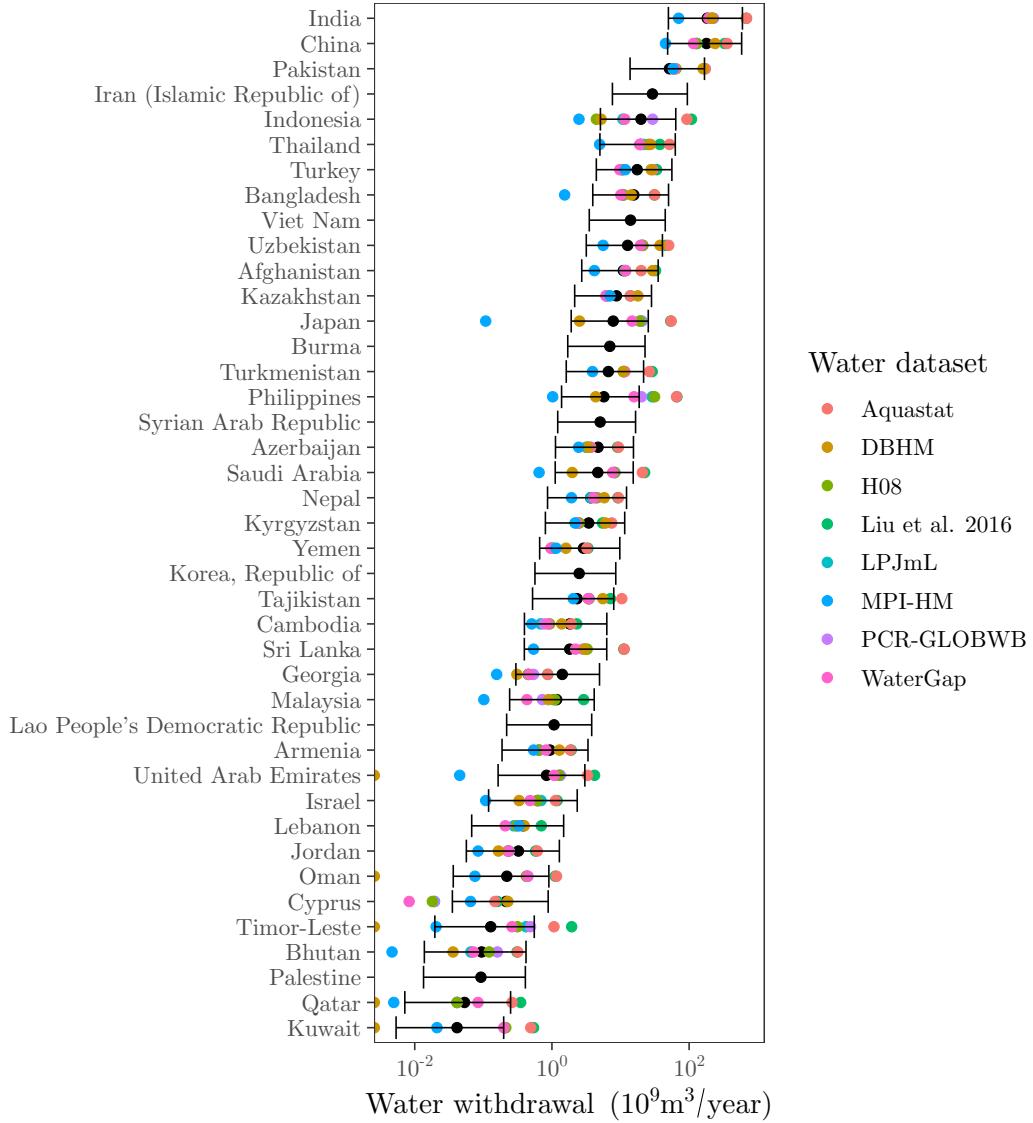


Figure 5: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models and FAO-based datasets.

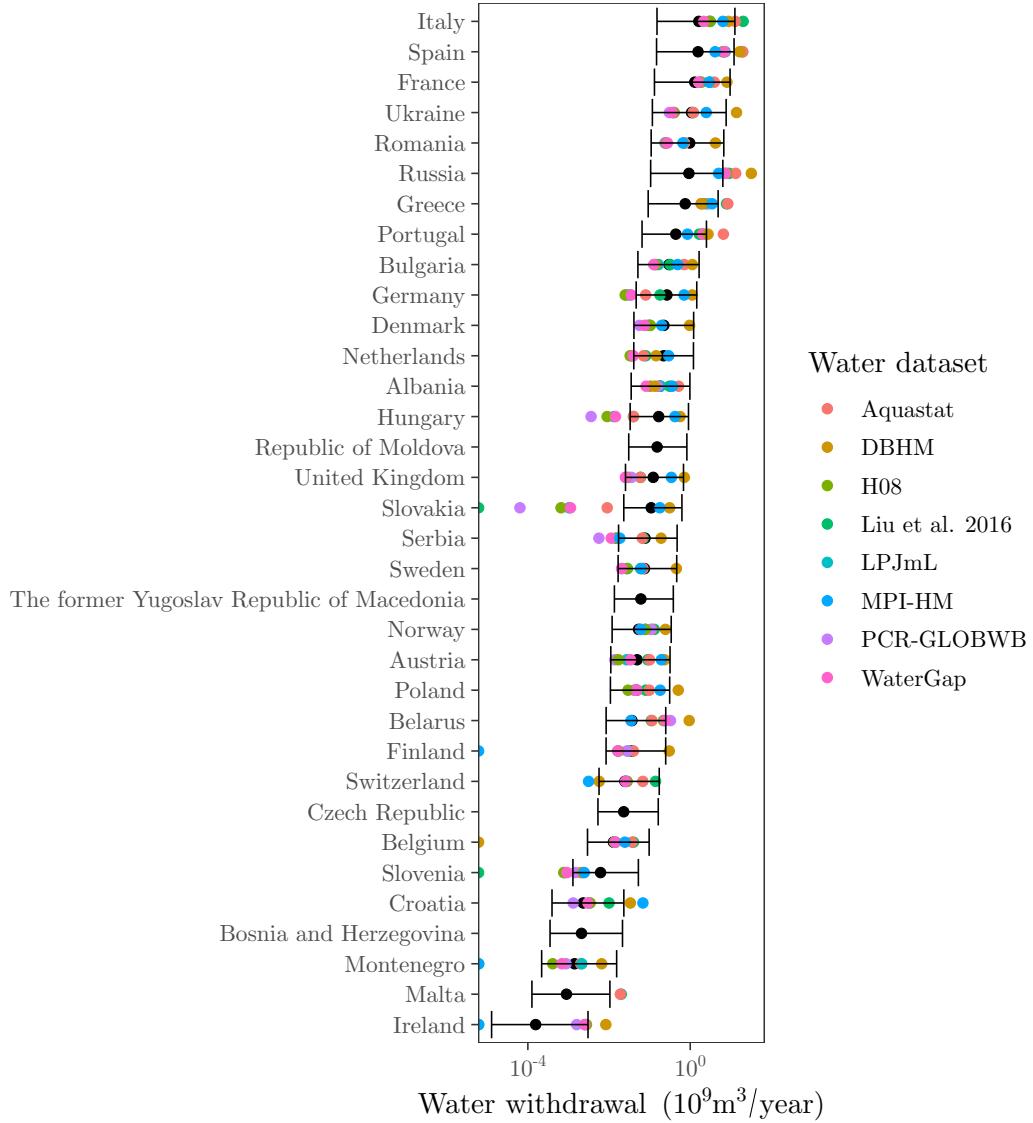


Figure 6: Validation of our approach. The black dots and the error bars show the range of irrigation water withdrawal values predicted from irrigated areas only. The colored dots show the irrigation water withdrawal values outputted by Global Hydrological Models and FAO-based datasets.

6 Lookup table

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

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

We finally export all datasets created so far to `.csv` files.

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

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

7 Run the model

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

7.1 Define settings

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

```
# 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", 1:3, sep = "")

# Select sample size
n <- 2 ^ 13

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

7.2 Sample matrix

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

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

```
# 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)
```

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

```
# TRANSFORM THE SAMPLE MATRIX -----
# Function to transform sample matrix to appropriate distributions
transform_sample_matrix <- function(dt) {
  dt[, X1 := floor(X1 * (8 - 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", "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]
}
```

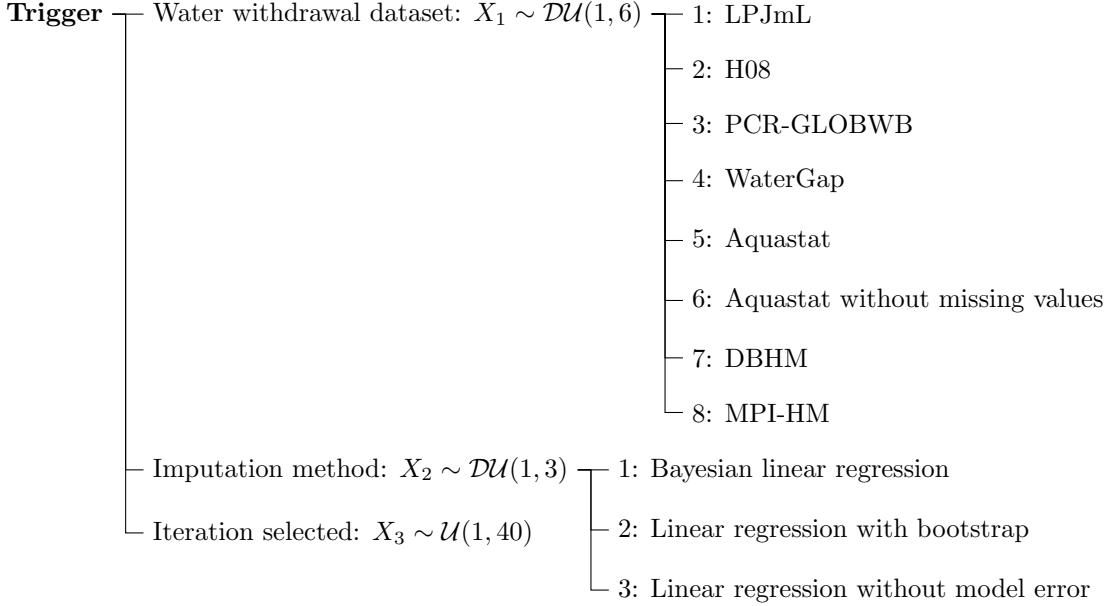


Figure 7: Tree diagram coding the discrete probability distributions of each trigger into each uncertainty level.

```

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

```

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

```

# PRINT SAMPLE MATRIX -----
print(sample.matrix.dt)

##          Continent      X1      X2 X3
## 1:    Africa Aquastat norm.boot 21
## 2:    Africa      DBHM     norm 31
## 3:    Africa  PCR-GLOBWB norm.nob 11
## 4:    Africa     WaterGap norm.boot 26
## 5:    Africa      MPI-HM norm.nob  6
## ---
## 294908:   Europe     WaterGap norm.boot 30
## 294909:   Europe      MPI-HM     norm 10
## 294910:   Europe        H08 norm.boot 40
## 294911:   Europe Liu et al. 2016 norm.nob 20
## 294912:   Europe      DBHM     norm 33

```

7.3 The model

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

```
# THE MODEL -----
model <- function(X) lookup[.(paste0(X[, 1:4], collapse = "_"))][, r.squared]
```

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

```
# RUN THE MODEL -----
# 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")
```

8 Uncertainty analysis

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

```
# 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 0.8215929 0.8372900 0.8517387 0.8663348 0.8878623 0.9133331
## 2: Americas 0.6762378 0.7291203 0.7937019 0.8659180 0.8984204 0.9456457
## 3: Asia 0.8334491 0.8478166 0.8598476 0.8722937 0.8846674 0.9299792
## 4: Europe 0.4958679 0.5863070 0.6621637 0.7199073 0.7899133 0.8783011
##          q0.99      q1      mean      median
## 1: 0.9181754 0.9211349 0.8684724 0.8663348
## 2: 0.9498877 0.9563170 0.8435401 0.8659180
## 3: 0.9350349 0.9400244 0.8739635 0.8722937
## 4: 0.8845060 0.9002252 0.7165019 0.7199073

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

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)),

```

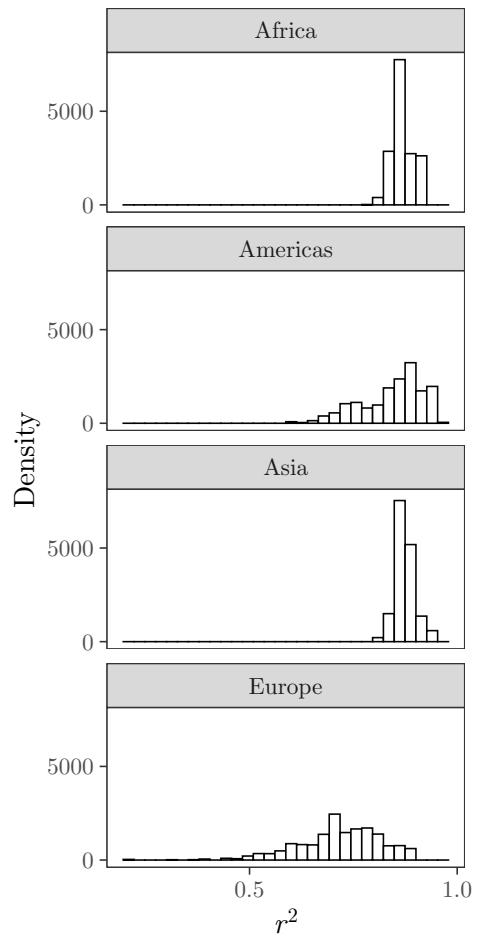


Figure 8: Uncertainty in the empirical distribution of r^2 .

```

    "MPI-HM" = expression(bold(MPI-HM)),
    "LPJmL" = expression(bold(LPJmL)),
    "H08" = expression(bold(H08)),
    "WaterGap" = expression(bold(WaterGap))) + 
  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")

## `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()

```

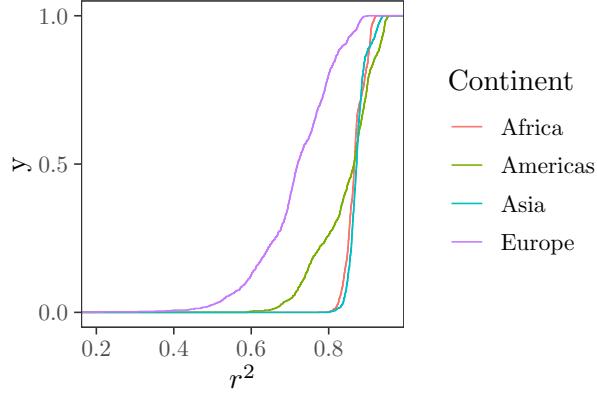


Figure 9: Cumulative empirical distribution for r^2 .

9 Sensitivity analysis

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

```

# 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", 1:3, sep = ""))

# R squared
ggplot(scatter.dt, aes(r.squared, value)) +
  geom_point(alpha = 0.1, size = 0.5) +
  facet_grid(variable ~ Continent,
             scales = "free_y") +
  labs(y = "",
       x = expression(italic(r)^2)) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  scale_color_manual(values = c("#00BFC4", "#F8766D")) +
  theme_AP() +
  theme(axis.text.y = element_text(angle = 45, hjust = 1, size = 6),
        legend.position = "top")

```

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

```

# SENSITIVITY ANALYSIS ----

# Number of bootstrap replicas
R <- 1000

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

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

# PRINT AND EXPORT SENSITIVITY INDICES ----

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

##      Continent      original        bias std.error      low.ci      high.ci
## 1:    Africa  0.720619411 -7.994611e-05  0.005197866  0.71051173  0.73088699

```

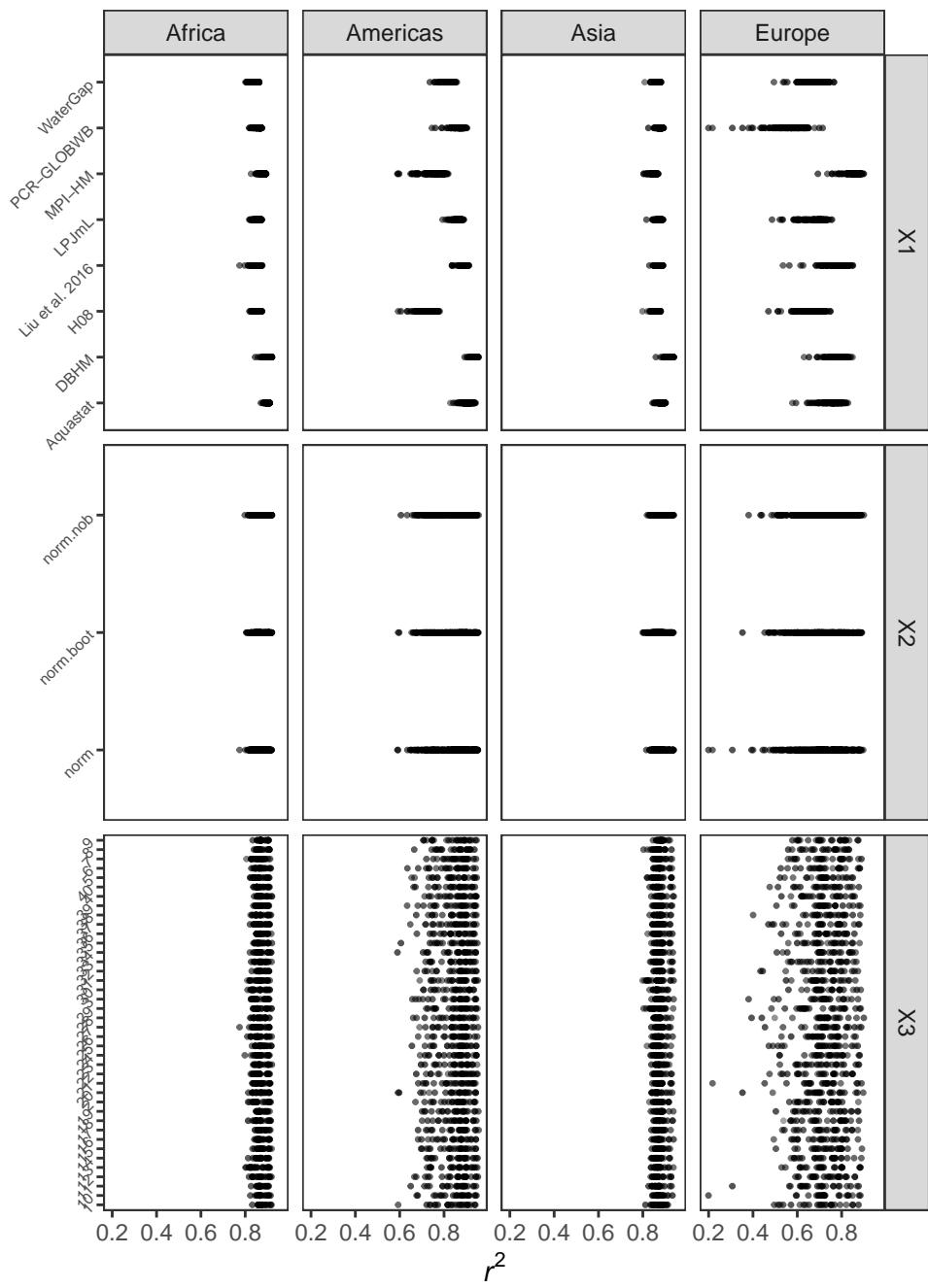


Figure 10: Scatterplots of r^2 against the triggers' levels.

```

## 2: Africa 0.010697268 -1.196400e-04 0.013049718 -0.01476007 0.03639388
## 3: Africa 0.044124350 2.275565e-04 0.013815199 0.01681950 0.07097409
## 4: Africa 0.893166906 -4.469596e-04 0.013910263 0.86635025 0.92087748
## 5: Africa 0.165582325 1.452862e-04 0.003899822 0.15779353 0.17308055
## 6: Africa 0.265048916 -2.040609e-04 0.005130178 0.25519801 0.27530794
## 7: Americas 0.874486367 -1.626353e-05 0.003309601 0.86801593 0.88098933
## 8: Americas -0.001168093 3.415676e-04 0.012566921 -0.02614037 0.02312105
## 9: Americas 0.011018352 6.343718e-04 0.013281883 -0.01564803 0.03641599
## 10: Americas 0.968019902 -3.192744e-04 0.013565312 0.94175165 0.99492670
## 11: Americas 0.087442878 1.321420e-04 0.002739294 0.08194182 0.09267965
## 12: Americas 0.122352770 2.560315e-05 0.003128528 0.11619536 0.12845897
## 13: Asia 0.730515628 -6.397168e-05 0.005942775 0.71893198 0.74222722
## 14: Asia -0.001026535 -2.264506e-04 0.013728241 -0.02770694 0.02610677
## 15: Asia 0.038043782 -2.956189e-04 0.014099072 0.01070573 0.06597308
## 16: Asia 0.846120801 8.057625e-04 0.013521626 0.81881314 0.87181694
## 17: Asia 0.185832640 9.646047e-05 0.004890859 0.17615027 0.19532209
## 18: Asia 0.265703075 -1.946358e-04 0.005593892 0.25493388 0.27686154
## 19: Europe 0.729714816 -2.159044e-04 0.006452730 0.71728360 0.74257784
## 20: Europe 0.003395427 -5.595039e-04 0.014352104 -0.02417468 0.03208454
## 21: Europe 0.013724452 3.706511e-05 0.016291964 -0.01824428 0.04561905
## 22: Europe 0.946109350 4.783269e-05 0.015915523 0.91486767 0.97725537
## 23: Europe 0.179895778 -2.034249e-04 0.005998639 0.16834209 0.19185632
## 24: Europe 0.264681637 -1.261745e-04 0.006426923 0.25221127 0.27740435
##   Continent      original          bias    std.error      low.ci     high.ci
##   sensitivity parameters
## 1: Si      $X_1$ 
## 2: Si      $X_2$ 
## 3: Si      $X_3$ 
## 4: Ti      $X_1$ 
## 5: Ti      $X_2$ 
## 6: Ti      $X_3$ 
## 7: Si      $X_1$ 
## 8: Si      $X_2$ 
## 9: Si      $X_3$ 
## 10: Ti     $X_1$ 
## 11: Ti     $X_2$ 
## 12: Ti     $X_3$ 
## 13: Si      $X_1$ 
## 14: Si      $X_2$ 
## 15: Si      $X_3$ 
## 16: Ti     $X_1$ 
## 17: Ti     $X_2$ 
## 18: Ti     $X_3$ 
## 19: Si      $X_1$ 
## 20: Si      $X_2$ 
## 21: Si      $X_3$ 
## 22: Ti     $X_1$ 
## 23: Ti     $X_2$ 

```

```

## 24:           Ti      $X_3$  

##   sensitivity parameters  

fwrite(indices, "indices.csv")  

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

bottom <- indices[sensitivity %in% c("Si", "Ti")] %>%  

  ggplot(., aes(parameters, original, fill = sensitivity)) +  

    geom_bar(stat = "identity",  

              position = position_dodge(0.6),  

              color = "black") +  

    geom_errorbar(aes(ymin = low.ci,  

                      ymax = high.ci),  

                  position = position_dodge(0.6)) +  

    scale_y_continuous(breaks = scales::pretty_breaks(n = 3)) +  

    facet_wrap(~Continent,  

               ncol = 4) +  

    labs(x = "",  

         y = "Sobol' index") +  

    scale_fill_discrete(name = "Sobol' indices",  

                         labels = c(expression(S[italic(i)]),  

                                    expression(T[italic(i)]))) +  

    theme_AP() +  

    theme(legend.position = "top")  

bottom

```

Sobol' indices █ S_i █ T_i

Continent	Parameter	S_i	T_i
Africa	X_1	~0.75	~0.90
	X_2	~0.05	~0.15
	X_3	~0.05	~0.25
Americas	X_1	~0.85	~0.95
	X_2	~0.05	~0.10
	X_3	~0.05	~0.15
Asia	X_1	~0.75	~0.85
	X_2	~0.05	~0.15
	X_3	~0.05	~0.25
Europe	X_1	~0.75	~0.95
	X_2	~0.05	~0.15
	X_3	~0.05	~0.25

```

# 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 -----  


```

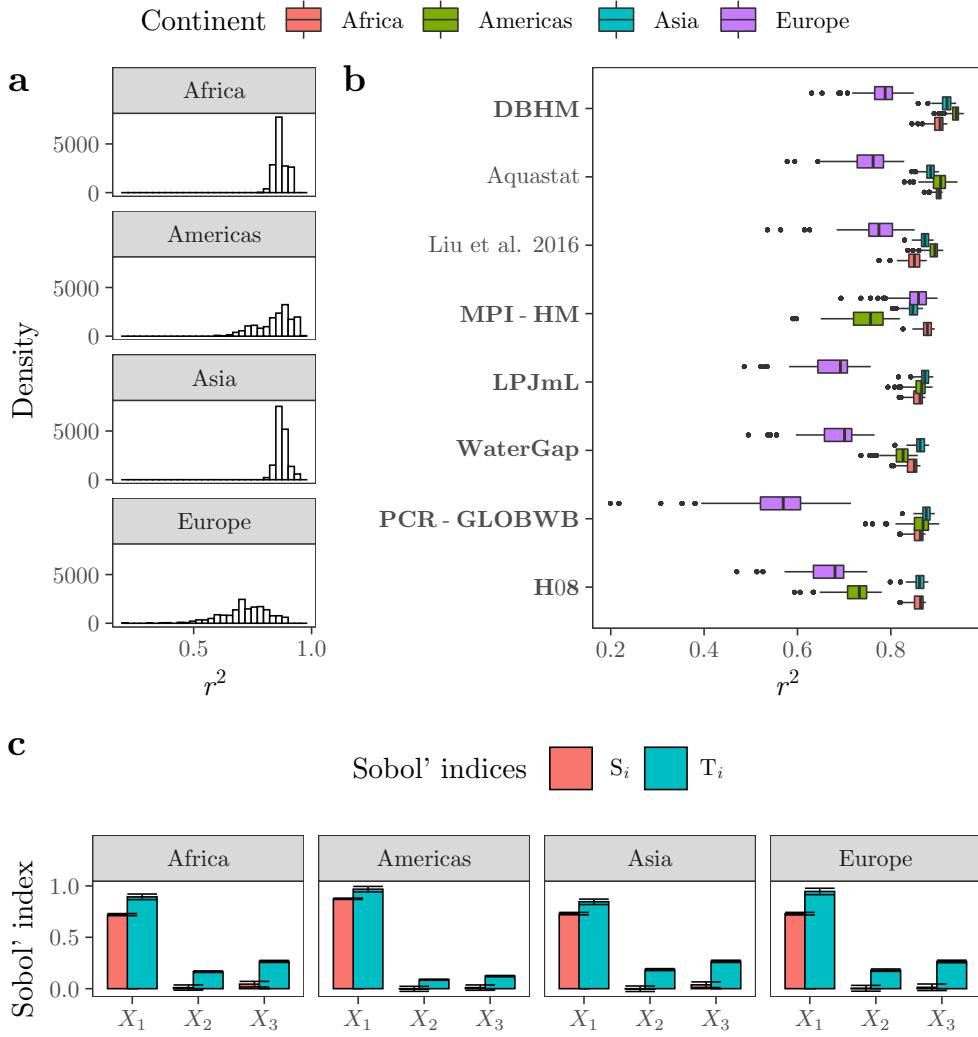


Figure 11: Uncertainty and sensitivity analysis. a) Empirical distribution for r^2 at the continental level. b) Boxplots of r^2 values obtained when regressions were run with GHM (in bold) and FAO-based datasets. c) Sobol' indices. S_i and T_i refer respectively to Sobol' first and total order indices. 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.

```

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

##   Continent      V1
## 1: Africa 0.7754410
## 2: Americas 0.8843366
## 3: Asia 0.7675329
## 4: Europe 0.7468347

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

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

```

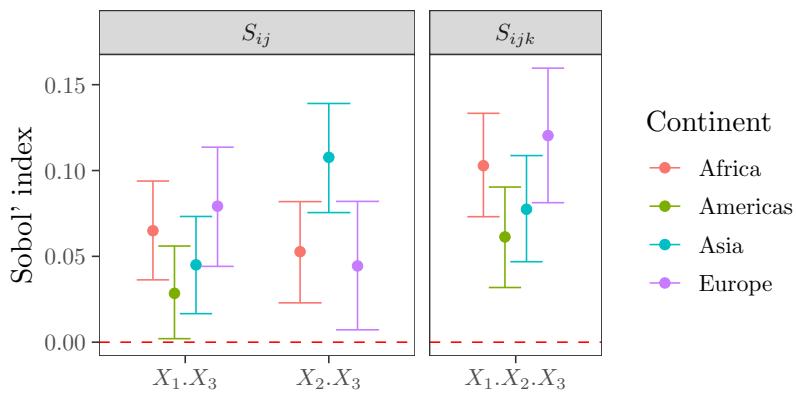


Figure 12: High-order interactions between the triggers.

10 Session information

```
# SESSION INFORMATION -----  
  
sessionInfo()  
  
## R version 3.6.3 (2020-02-29)  
## Platform: x86_64-apple-darwin15.6.0 (64-bit)  
## Running under: macOS Catalina 10.15.5  
##  
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib  
##  
## locale:  
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8  
##  
## attached base packages:  
## [1] parallel stats      graphics grDevices utils      datasets  methods  
## [8] base  
##  
## other attached packages:  
## [1] checkpoint_0.4.9 forcats_0.5.0   stringr_1.4.0    dplyr_0.8.5  
## [5] purrrr_0.3.4     readr_1.3.1     tidyverse_1.3.0  benchmarkme_1.0.3 cowplot_1.0.0  
## [9] ggplot2_3.3.0    broom_0.5.6     ggridges_0.5.2   mice_3.8.0  
## [13] naniar_0.5.0    zoo_1.8-7      sandwich_2.5-1   mvoutlier_2.0.9  
## [17] lmtest_0.9-37   complmrob_0.7.0 doParallel_1.0.15 iterators_1.0.12  
## [21] sgeostat_1.0-27 foreach_1.5.0    MASS_7.3-51.5    scales_1.1.1  
## [25] IDPmisc_1.1.20 countrycode_1.1.1 rworldmap_1.3-6   boot_1.3-24  
## [29] ncdf4_1.17     sensobol_0.3    data.table_1.12.8  
##  
## loaded via a namespace (and not attached):  
## [1] readxl_1.3.1      backports_1.1.6   spam_2.5-1  
## [4] plyr_1.8.6        splines_3.6.3     digest_0.6.25  
## [7] htmltools_0.4.0   fansi_0.4.1      magrittr_1.5  
## [10] cluster_2.1.0    openxlsx_4.1.4   modelr_0.1.6  
## [13] tikzDevice_0.12.3 colorspace_1.4-1  rvest_0.3.5  
## [16] rrcov_1.5-2      haven_2.2.0     xfun_0.13  
## [19] crayon_1.3.4    jsonlite_1.6.1   survival_3.1-12  
## [22] glue_1.4.0       gtable_0.3.0    car_3.0-7  
## [25] kernlab_0.9-29 prabclus_2.3-2  DEoptimR_1.0-8  
## [28] maps_3.3.0      abind_1.4-5     VIM_5.1.1  
## [31] mvtnorm_1.1-0   DBI_1.1.0      GGally_1.5.0  
## [34] bibtex_0.4.2.2  Rcpp_1.0.4.6   sROC_0.1-2  
## [37] laeken_0.5.1   foreign_0.8-76  mclust_5.4.6  
## [40] dotCall64_1.0-0  stats4_3.6.3   vcd_1.4-7  
## [43] truncnorm_1.0-8 httr_1.4.1    RColorBrewer_1.1-2
```

```

## [46] fpc_2.2-5           modeltools_0.2-23    ellipsis_0.3.0
## [49] farver_2.0.3         pkgconfig_2.0.3     reshape_0.8.8
## [52] NADA_1.6-1.1        flexmix_2.3-15     nnet_7.3-13
## [55] dbplyr_1.4.3         labeling_0.3       tidyselect_1.0.0
## [58] rlang_0.4.5          munsell_0.5.0     cellranger_1.1.0
## [61] tools_3.6.3          cli_2.0.2         generics_0.0.2
## [64] ranger_0.12.1        pls_2.7-2         evaluate_0.14
## [67] cvTools_0.3.2        yaml_2.2.1        knitr_1.28
## [70] fs_1.4.1             filehash_2.4-2   zip_2.0.4
## [73] robustbase_0.93-6   visdat_0.5.3      nlme_3.1-147
## [76] xml2_1.3.1           compiler_3.6.3    rstudioapi_0.11
## [79] curl_4.3              e1071_1.7-3      zCompositions_1.3.4
## [82] reprex_0.3.0          robCompositions_2.2.1 pcaPP_1.9-73
## [85] stringi_1.4.6         highr_0.8         fields_10.3
## [88] lattice_0.20-41       Matrix_1.2-18     vctrs_0.2.4
## [91] pillar_1.4.3          lifecycle_0.2.0   Rdpack_0.11-1
## [94] maptools_0.9-9        gbRd_0.4-11      R6_2.4.1
## [97] rio_0.5.16            codetools_0.2-16 benchmarkmeData_1.0.3
## [100] assertthat_0.2.1     withr_2.2.0       diptest_0.75-7
## [103] hms_0.5.3            grid_3.6.3        class_7.3-16
## [106] rmarkdown_2.1          carData_3.0-3    lubridate_1.7.8
## [109] tinytex_0.22

## 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

```

References

- FAO. 2016. “AQUASTAT website.” Food; Agriculture Organization of the United Nations. <http://www.fao.org/nr/water/aquastat/didyouknow/index3.stm>.
- . 2017. “FAOSTAT database.” Rome. <http://www.fao.org/faostat/en/>.
- Frenken, Karen, and Virginie Gillet. 2012. “Irrigation water requirement and water withdrawal by country.” Food; Agriculture Organization of the United Nations. <http://www.fao.org/3/abc824e.pdf>.
- Graham, John W., Allison E. Olchowski, and Tamika D. Gilreath. 2007. “How many imputations are really needed? Some practical clarifications of multiple imputation theory.” *Prevention Science* 8 (3): 206–13. <https://doi.org/10.1007/s11121-007-0070-9>.
- Huang, Zhongwei, Mohamad Hejazi, Xinya Li, QiuHong Tang, Chris Vernon, Guoyong Leng, Yaling Liu, et al. 2018. “Global gridded monthly sectoral water use dataset for 1971-2010: v2.” <https://doi.org/10.5281/ZENODO.1209296>.
- Jansen, M. 1999. “Analysis of variance designs for model output.” *Computer Physics Communications* 117 (1-2): 35–43. [https://doi.org/10.1016/S0010-4655\(98\)00154-4](https://doi.org/10.1016/S0010-4655(98)00154-4).
- Liu, Yaling, Mohamad Hejazi, Page Kyle, Son H. Kim, Evan Davies, Diego G. Miralles, Adriaan J. Teuling, Yujie He, and Dev Niyogi. 2016. “Global and regional evaluation of energy for water.” *Environmental Science and Technology* 50 (17): 9736–45. <https://doi.org/10.1021/acs.est.6b01065>.
- Meier, J., F. Zabel, and W. Mauser. 2018. “A global approach to estimate irrigated areas . A comparison between different data and statistics.” *Hydrology and Earth System Sciences* 22 (2): 1119–33. <https://doi.org/10.5194/hess-22-1119-2018>.
- Puy, Arnald, Samuele Lo Piano, Andrea Saltelli, and Simon Levin. n.d. “sensobol: an R package to compute high-order sensitivity indices.”
- Salmon, J.M., M. A. Friedl, S. Frolking, D. Wisser, and E. M. Douglas. 2015. “Global rain-fed, irrigated, and paddy croplands: A new high resolution map derived from remote sensing, crop inventories and climate data.” *International Journal of Applied Earth Observation and Geoinformation* 38. Elsevier B.V.: 321–34. <https://doi.org/10.1016/j.jag.2015.01.014>.
- Siebert, S., V. Henrich, K. Frenken, and J. Burke. 2013. “Update of the digital global map of irrigation areas to version 5.” Rome: Rheinische Friedrich-Wilhelms-University; Food; Agriculture Organization of the United Nations.
- Sobol’, Ilya. M. 1967. “On the distribution of points in a cube and the approximate evaluation of integrals.” *USSR Computational Mathematics and Mathematical Physics* 7 (4): 86–112. [https://doi.org/10.1016/0041-5553\(67\)90144-9](https://doi.org/10.1016/0041-5553(67)90144-9).
- . 1976. “Uniformly distributed sequences with an additional uniform property.” *USSR Computational Mathematics and Mathematical Physics* 16 (5): 236–42. [https://doi.org/10.1016/0041-5553\(76\)90154-3](https://doi.org/10.1016/0041-5553(76)90154-3).
- Stacke, T., and S. Hagemann. 2012. “Development and evaluation of a global dynamical wetlands extent scheme.” *Hydrology and Earth System Sciences* 16 (8): 2915–33. <https://doi.org/10.5194/hess-16-2915-2012>.

Tang, QiuHong, Taikan Oki, Shinjiro Kanae, and Heping Hu. 2007. "The influence of precipitation variability and partial irrigation within grid cells on a hydrological simulation." *Journal of Hydrometeorology* 8 (3): 499–512. <https://doi.org/10.1175/JHM589.1>.

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