

Global irrigation water demands biased by unreliable irrigation efficiencies

R code

Arnald Puy

Contents

1	Preliminary steps	2
2	Read in data	3
3	The model	7
3.1	Function to create sample matrix	7
3.2	Define distributions	8
3.3	Uncertainty in the proportion of large-scale irrigated areas	10
3.4	Function to create sample matrix and transform to appropriate distributions	11
3.5	Define the model	11
3.6	Define settings	13
3.7	Run the model in parallel	13
3.8	Extract model output	13
4	Uncertainty analysis	14
4.1	Coefficient of variation	14
4.2	Ranges	17
4.3	Overlap between irrigation efficiencies	22
4.4	Retrieve data from ISIMIP	36
4.5	Retrieve data from ISIMIP (climate change in 2050)	38
5	Sensitivity analysis	45

1 Preliminary steps

```
# 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)
    }
  }
}

# Load the packages
loadPackages(c("data.table", "tidyverse", "sensobol", "wesanderson",
              "cowplot", "parallel", "foreach", "doParallel",
              "countrycode", "gggridges", "scales", "overlapping",
              "sp", "rworldmap", "ncdf4", "benchmarkme"))

# Create custom theme
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),
          legend.position = "top",
          strip.background = element_rect(fill = "white"),
          plot.margin = margin(t = 0, r = 0.3, b = 0, l = 0.3, unit = "cm"))
}

# Set checkpoint

dir.create(".checkpoint")
library("checkpoint")

checkpoint("2021-08-02",
          R.version = "4.0.3",
          checkpointLocation = getwd())
```

2 Read in data

```
# READ IN DATA -----

# Rohwer data
rohwer <- fread("rohwer_data_all.csv")
rohwer[rohwer == ""] <- NA
rohwer <- rohwer[, Large_fraction:= Large_fraction / 100]

# Jager data
jager <- fread("jager_data.csv")
jager.list <- split(jager, jager$Country)

# Bos data
bos <- fread("bos_data.csv")
bos <- bos[, Scale := ifelse(Irrigated_area < 10000, "<10.000 ha", ">10.000 ha")]

# Solley data (USA)
usa.dt <- fread("usa_efficiency.csv")
usa.dt <- usa.dt[, Efficiency:= consumptive.use / total.withdrawal]

# FAO 1997 data (Irrigation potential in Africa)
fao_dt <- fread("fao_1997.csv")
fao_dt <- fao_dt[, Efficiency:= Efficiency / 100]

# Create data set with E_a values as defined by Rohwer
bos.rohwer.ea <- data.table("Irrigation" = c("Surface", "Sprinkler"),
                           "Value" = c(0.6, 0.7),
                           "variable" = "E[a]")

# Create data set with E_c values as defined by Rohwer
bos.rohwer.ec <- data.table("Irrigation" = c("Surface", "Sprinkler"),
                           "Value" = c(0.8, 0.95),
                           "variable" = "E[c]")

bos.rohwer.all <- rbind(bos.rohwer.ec, bos.rohwer.ea)

# As a function of scale
bos.rohwer.mf.ec <- data.table("Scale" = c("<10.000 ha", ">10.000 ha"),
                              "Value" = c(0.85, 0.59),
                              "variable" = "E[c]")

bos.rohwer.mf.ed <- data.table("Scale" = c("<10.000 ha", ">10.000 ha"),
                              "Value" = c(0.81, 0.72),
                              "variable" = "E[d]")

bos.rohwer.mf.all <- rbind(bos.rohwer.mf.ec, bos.rohwer.mf.ed)
```

```

# PLOT -----

bos2 <- copy(bos)
bos2 <- setnames(bos2, c("E_a", "E_c", "E_d"), c("E[a]", "E[c]", "E[d]"))

# Field and conveyance efficiency -----

a <- bos2 %>%
  melt(., measure.vars = c("E[a]", "E[c]")) %>%
  ggplot(., aes(value, fill = Irrigation, color = Irrigation)) +
  geom_histogram(position = "identity", alpha = 0.4, bins = 15) +
  facet_wrap(~variable, labeller = label_parsed) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  geom_vline(data = bos.rohwer.all, aes(xintercept = Value,
                                       color = Irrigation,
                                       group = variable),
            lty = 2,
            size = 1) +
  labs(x = "", y = "Counts") +
  theme_AP()

# As a function of scale -----

b <- melt(bos2, measure.vars = c("E[c]", "E[a]", "E[d]")) %>%
  na.omit() %>%
  ggplot(., aes(value, fill = Scale, color = Scale)) +
  geom_histogram(bins = 15, position = "identity", alpha = 0.6) +
  labs(x = "Irrigation efficiency", y = "Counts") +
  facet_wrap(~ variable, labeller = label_parsed) +
  geom_vline(data = bos.rohwer.mf.all, aes(xintercept = Value,
                                       color = Scale,
                                       group = variable),
            lty = 2,
            size = 1) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  scale_color_manual(values = wes_palette(2, name = "Chevalier1"),
                    name = "Scale",
                    labels = c("<10.000 ha", ">10.000 ha")) +
  scale_fill_manual(values = wes_palette(2, name = "Chevalier1"),
                   name = "Scale",
                   labels = c("<10.000 ha", ">10.000 ha")) +
  theme_AP()

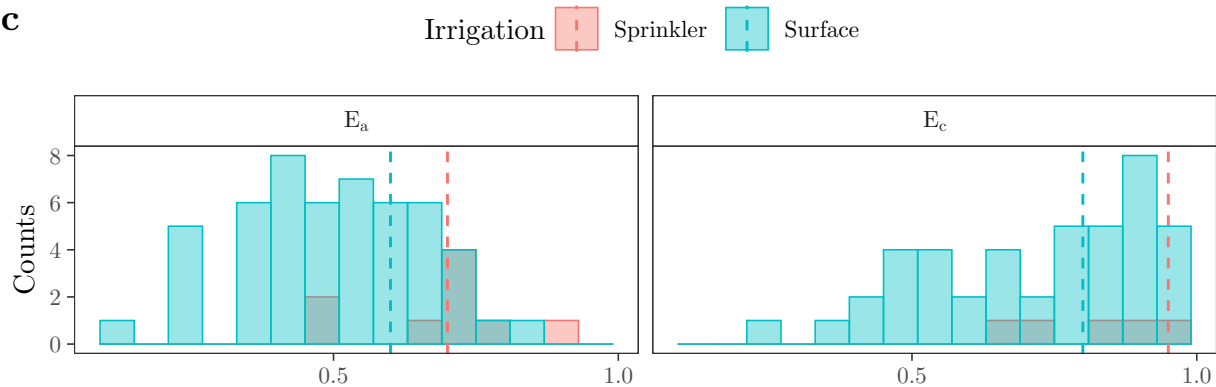
bottom <- plot_grid(a, b, ncol = 1, labels = c("c", "d"))

## Warning: Removed 74 rows containing non-finite values (stat_bin).

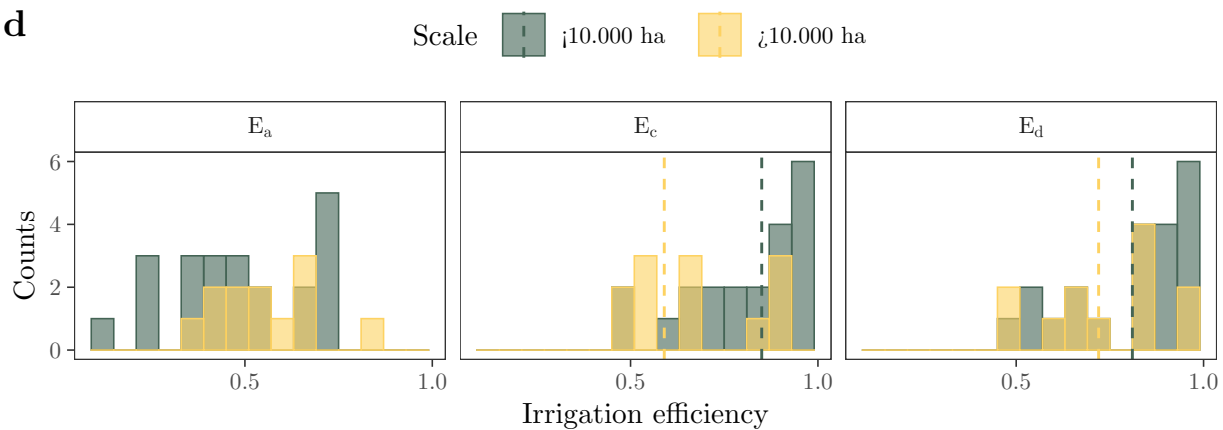
```

bottom

c



d



PLOT USA AND AFRICA -----

```
c1 <- ggplot(usa.dt, aes(Efficiency)) +
  geom_histogram() +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  geom_vline(xintercept = 0.6, lty = 2) +
  labs(x = "", y = "Counts") +
  theme_AP()

d1 <- ggplot(fao.dt, aes(Efficiency)) +
  geom_histogram() +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  labs(x = "", y = "") +
  theme_AP()

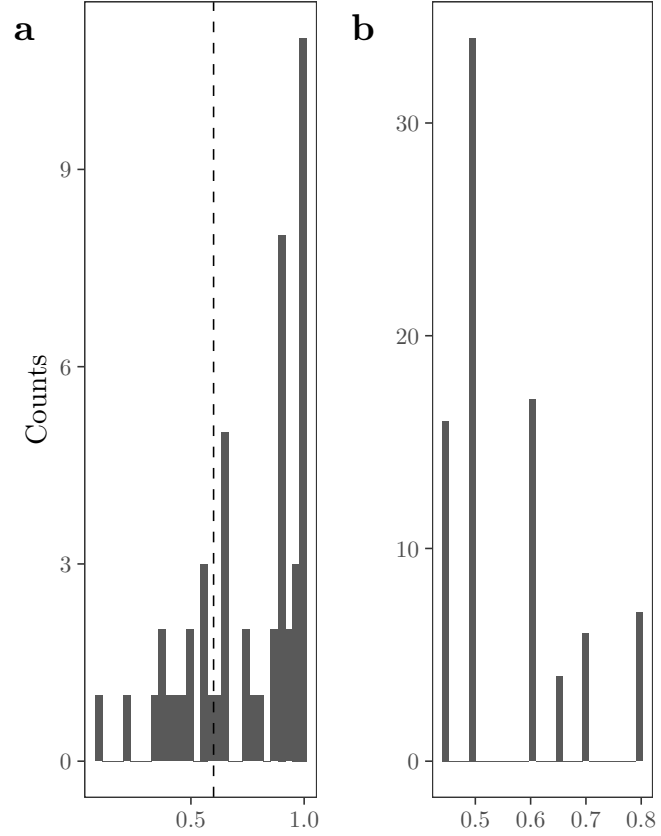
top <- cowplot::plot_grid(c1, d1, ncol = 2, labels = "auto")
```

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

Warning: Removed 3 rows containing non-finite values (stat_bin).

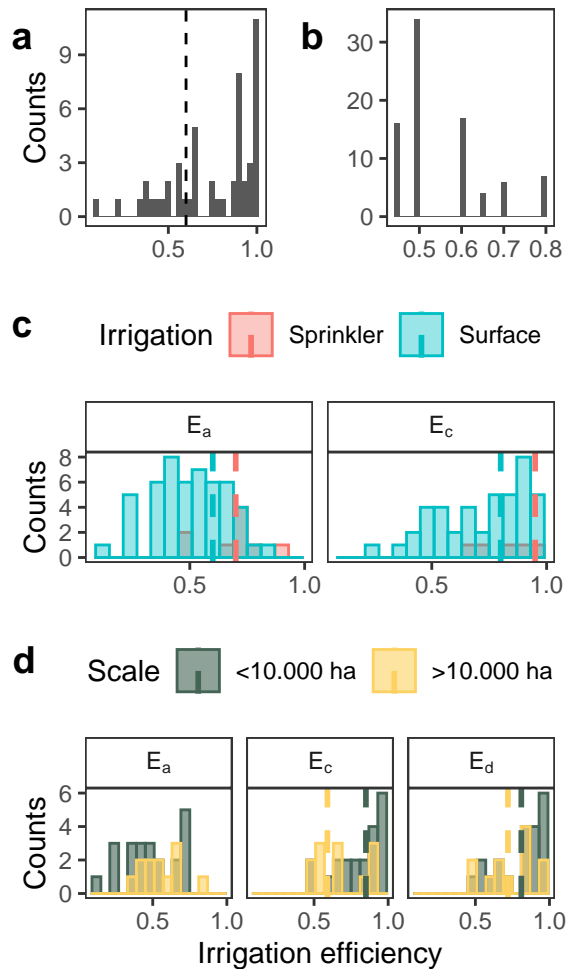
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

top



PLOT MERGED

```
plot_grid(top, bottom, ncol = 1, rel_heights = c(0.3, 0.7))
```



3 The model

3.1 Function to create sample matrix

```
# CREATE FUNCTION TO DESIGN SAMPLE MATRIX -----

params_algo <- list(
  "Surface" = c("Ea_surf", "Ec_surf", "m", "r_L", "X1", "X2"),
  "Sprinkler" = c("Ea_sprinkler", "Ec_sprinkler", "m"),
  "Micro" = c("Ea_micro", "Ec_micro", "m"),
  "Mixed" = c("Ea_surf", "Ea_sprinkler", "Ec_surf", "Ec_sprinkler", "m", "r_L", "X1", "X2"),
  "Jager" = c("Ea_surf", "Ea_sprinkler", "Ec_surf", "Ec_sprinkler",
              "Ea_micro", "Ec_micro", "m", "r_L", "X1", "X2")
)

params_fun <- function(IFT) {
  out <- params_algo[[IFT]]
  return(out)
}
```

```

sample_matrix_fun <- function(IFT) {
  params <- params_fun(IFT = IFT)
  mat <- sensobol::sobol_matrices(N = N, params = params)
  out <- list(params, mat)
  names(out) <- c("parameters", "matrix")
  return(out)
}

```

3.2 Define distributions

```

# DEFINE TRUNCATED DISTRIBUTIONS -----

# EA SURFACE -----

Ea.surface <- bos[Irrigation == "Surface"][, .(min = min(E_a, na.rm = TRUE),
                                                    max = max(E_a, na.rm = TRUE))]

shape <- 3.502469
scale <- 0.5444373
minimum <- Ea.surface$min
maximum <- Ea.surface$max
weibull_dist <- sapply(c(minimum, maximum), function(x)
  pweibull(x, shape = shape, scale = scale))

# EC SURFACE -----

Ec.surface <- bos[Irrigation == "Surface"][, .(min = min(E_c, na.rm = TRUE),
                                                    max = max(E_c, na.rm = TRUE))]

shape1 <- 5.759496
shape2 <- 1.403552
minimum.beta <- Ec.surface$min
maximum.beta <- Ec.surface$max
beta_dist <- sapply(c(minimum.beta, maximum.beta), function(x)
  pbeta(x, shape1 = shape1, shape2 = shape2))

# EA SPRINKLER -----

Ea.sprinkler <- bos[Irrigation == "Sprinkler"][, .(min = min(E_a, na.rm = TRUE),
                                                    max = max(E_a, na.rm = TRUE))]

shape.spr <- 6.9913711
scale.spr <- 0.7451178
minimum.spr <- Ea.sprinkler$min
maximum.spr <- Ea.sprinkler$max
weibull_dist_spr <- sapply(c(minimum.spr, maximum.spr), function(x)
  pweibull(x, shape = shape.spr, scale = scale.spr))

# MANAGEMENT FACTOR (m) -----

```



```

shape1.m <- 5.759496
shape2.m <- 1.403552
minimum.m <- 0.65
maximum.m <- 1
beta_dist.m <- sapply(c(minimum.m, maximum.m), function(x)
  pbeta(x, shape1 = shape1.m, shape2 = shape2.m))

# FUNCTION TO TRANSFORM TO APPROPRIATE DISTRIBUTIONS -----

distributions_fun <- list(

  # SURFACE IRRIGATION
  # -----

  "Ea_surf" = function(x) {

    out <- qunif(x, weibull_dist[[1]], weibull_dist[[2]])
    out <- qweibull(out, shape, scale)
  },

  "Ec_surf" = function(x) {

    out <- qunif(x, beta_dist[[1]], beta_dist[[2]])
    out <- qbeta(out, shape1, shape2)
  },

  # SPRINKLER IRRIGATION
  # -----

  "Ea_sprinkler" = function(x) {

    out <- qunif(x, weibull_dist_spr[[1]], weibull_dist_spr[[2]])
    out <- qweibull(out, shape.spr, scale.spr)
  },

  "Ec_sprinkler" = function(x) qunif(x, 0.64, 0.96),

  # MICRO (DRIP) IRRIGATION
  # -----

  "Ea_micro" = function(x) out <- qunif(x, 0.75, 0.95),
  "Ec_micro" = function(x) out <- qunif(x, 0.9, 0.95),

  # PROPORTION LARGE
  # -----

  "Proportion_large" = function(x) x,

```

```

# MANAGEMENT FACTOR
# -----

"m" = function(x) {
  out <- qunif(x, beta_dist.m[[1]], beta_dist.m[[2]])
  out <- qbeta(out, shape1.m, shape2.m)
},

# REDUCTION IN MANAGEMENT FACTOR DUE TO LARGE-SCALE
# -----
"r_L" = function(x) qunif(x, 0, 0.5),

# IRRIGATED AREA DATASET
# -----

"X1" = function(x) floor(x * (4 - 1 + 1)) + 1,

# THRESHOLD FOR LARGE-SCALE IRRIGATED AREAS
# -----

"X2" = function(x) floor(x * (3 - 1 + 1)) + 1
)

```

3.3 Uncertainty in the proportion of large-scale irrigated areas

```

# DEFINE THE UNCERTAINTY IN THE LARGE FRACTION AT THE COUNTRY LEVEL -----

eff10 <- fread("efficiency_10.csv")
eff30 <- fread("efficiency_30.csv")
eff100 <- fread("efficiency_100.csv")

# CHECK WHICH COUNTRIES FROM ROHWER ET AL. ARE MISSING IN THE
# LARGE-SCALE IRRIGATED AREA DATASETS -----

countryDiff <- setdiff(rohwer$Country, eff100$Country)
countryMissing <- data.table(Country = rep(countryDiff, each = 12),
                             X1 = rep(1:4, each = 3),
                             X2 = rep(1:3, times = 4),
                             Proportion_large = 0)

# ARRANGE DATASETS -----

largescale.dt <- rbind(eff10, eff30, eff100) %>%
  melt(., measure.vars = 3:6, variable.name = "X1",
       value.name = "Proportion_large") %>%
  .[, Code:= NULL] %>%
  setcolorder(., c("Country", "X1", "X2", "Proportion_large")) %>%

```

```

[, X1:= ifelse(X1 == "FAO-GMIA", 1,
              ifelse(X1 == "GRIPC", 2,
                    ifelse(X1 == "GIAM", 3, 4)))] %>%
[, X2:= ifelse(X2 == 1000, 1,
              ifelse(X2 == 3000, 2, 3))] %>%
# Add missing countries
rbind(., countryMissing) %>%
merge(rohwer[, .(Country, IFT)], ., by = "Country", all.x = TRUE) %>%
[, index:= paste(Country, X1, X2, sep = ".")]

largescale.dt[is.na(largescale.dt)] <- 0

# SETKEY -----

triggers.dt <- setkey(largescale.dt, index)

```

3.4 Function to create sample matrix and transform to appropriate distributions

```

# FULL ALGORITHM TO CREATE SAMPLE MATRIX -----

full_sample_matrix <- function(IFT, Country) {
  tmp <- sample_matrix_fun(IFT = IFT)
  mat <- tmp[["matrix"]]
  temp <- colnames(mat)
  mat <- sapply(seq_along(temp), function(x) distributions_fun[[temp[x]]](mat[, x]))
  colnames(mat) <- temp
  out <- list(tmp$parameters, mat)
  names(out) <- c("parameters", "matrix")
  return(out)
}

```

3.5 Define the model

```

# FULL MODEL -----

full_model <- function(IFT, Country, sample.size, R) {

  country.differences <- setdiff(rohwer$Country, jager$Country)
  tmp <- full_sample_matrix(IFT = IFT, Country = Country)
  mat <- tmp$matrix

  if(IFT == "Surface" | IFT == "Mixed" | IFT == "Jager") {
    X1 <- mat[, "X1"]
    X2 <- mat[, "X2"]
    index <- paste(Country, X1, X2, sep = ".")
  }
}

```

```

    Proportion_large <- triggers.dt[index][, Proportion_large]
  }

  if(IFT == "Surface") {

    Mf <- mat[, "m"] - mat[, "r_L"] * Proportion_large
    y <- mat[, "Ea_surf"] * mat[, "Ec_surf"] * Mf

  } else if(IFT == "Sprinkler") {

    Mf <- mat[, "m"]
    y <- mat[, "Ea_sprinkler"] * mat[, "Ec_sprinkler"] * Mf

  } else if(IFT == "Mixed") {

    Mf.surf <- mat[, "m"] - mat[, "r_L"] * Proportion_large
    y.surf <- mat[, "Ea_surf"] * mat[, "Ec_surf"] * Mf.surf

    Mf.sprink <- mat[, "m"]
    y.sprink <- mat[, "Ea_sprinkler"] * mat[, "Ec_sprinkler"] * Mf.sprink

    y <- 0.5 * y.surf + 0.5 * y.sprink

  } else if(IFT == "Micro") {

    Mf <- mat[, "m"]
    y <- mat[, "Ea_micro"] * mat[, "Ec_micro"] * Mf

  } else if(IFT == "Jager") {

    if(Country %in% country.differences == TRUE) {
      next
    }

    Mf.surf <- mat[, "m"] - mat[, "r_L"] * Proportion_large
    y.surf <- mat[, "Ea_surf"] * mat[, "Ec_surf"] * Mf.surf

    Mf.spr <- mat[, "m"]
    y.spr <- mat[, "Ea_sprinkler"] * mat[, "Ec_sprinkler"] * Mf.spr

    Mf.micro <- mat[, "m"]
    y.micro <- mat[, "Ea_micro"] * mat[, "Ec_micro"] * Mf.micro

    y <- jager.list[[Country]]$Surface_fraction * y.surf +
      jager.list[[Country]]$Sprinkler_fraction * y.spr +
      jager.list[[Country]]$Drip_fraction * y.micro
  }

```

```

}

ind <- sobol_indices(N = sample.size, Y = y, params = tmp$parameters,
                    boot = TRUE, R = R)
out <- list(y, ind)
names(out) <- c("output", "indices")
return(out)
}

```

3.6 Define settings

```

# DEFINE SETTINGS -----

N <- 2^14
R <- 10^2
list_continents <- list(c("Africa", "Asia"), c("Americas", "Europe"))

```

3.7 Run the model in parallel

```

# RUN MODEL -----

new.rohwer <- rohwer[Country %in% jager$Country][, IFT:= "Jager"]
all.dt <- list(rohwer, new.rohwer)

y <- list()
for(j in 1:length(all.dt)) {
  y[[j]] <- mclapply(1:nrow(all.dt[[j]]), function(x)
    full_model(IFT = all.dt[[j]][[x, "IFT"]],
              Country = all.dt[[j]][[x, "Country"]],
              sample.size = N,
              R = R),
    mc.cores = detectCores() * 0.75)
}

```

3.8 Extract model output

```

# EXTRACT MODEL OUTPUT -----

names(y) <- c("Rohwer et al. 2007", "Jägermeyr et al. 2015")

output <- tmp <- list()
for(i in names(y)) {
  output[[i]] <- lapply(y[[i]], function(x) x[["output"]][1:(2 * N)])

  if(i == "Rohwer et al. 2007") {

```

```

names(output[[i]]) <- rohwer$Country

} else if(i == "Jägermeyr et al. 2015") {

  names(output[[i]]) <- new.rohwer$Country

}
tmp[[i]] <- lapply(output[[i]], data.table) %>%
  rbindlist(., idcol = "Country")

if(i == "Rohwer et al. 2007") {

  tmp[[i]] <- merge(tmp[[i]], rohwer[, .(Country, IFT)], all.x = TRUE) %>%
    .[, IFT:= factor(IFT, levels = c("Surface", "Sprinkler", "Micro", "Mixed"))]

} else if(i == "Jägermeyr et al. 2015") {

  tmp[[i]] <- tmp[[i]][, IFT:= "Jager"]

}

tmp[[i]] <- tmp[[i]][, Continent:= countrycode(tmp[[i]][, Country],
                                              origin = "country.name",
                                              destination = "continent")]

}

```

```
## Warning in countrycode_convert(sourcevar = sourcevar, origin = origin, destination = dest,
```

```
## Warning in countrycode_convert(sourcevar = sourcevar, origin = origin, destination = dest,
```

```

uncertainty.dt <- rbindlist(tmp, idcol = "Approach")
uncertainty.dt <- uncertainty.dt[, Study:= ifelse(IFT == "Jager",
                                                "Jägermeyr et al. approach",
                                                "Rohwer et al. approach")]

```

```
# EXPORT UNCERTAINTY IN IRRIGATION EFFICIENCY -----
```

```
fwrite(uncertainty.dt, "uncertainty.dt.csv")
```

4 Uncertainty analysis

4.1 Coefficient of variation

```
# CALCULATE COEFFICIENT OF VARIATION -----
```

```

cv.dt <- uncertainty.dt[, .(sd = sd(V1), mean = mean(V1)),
                          .(Country, Approach, Continent)] %>%
  .[, cv:= sd / mean]

```

```

dd <- list()
for (i in 1:length(list_continents)) {
  dd[[i]] <- ggplot(cv.dt[Continent %in% list_continents[[i]]],
    aes(reorder(Country, cv), cv, color = Approach)) +
    geom_point() +
    scale_color_manual(values = wes_palette("Chevalier1"),
      labels = c("Jägermeyr et al. approach",
        "Rohwer et al. approach")) +
    labs(y = "Coefficient of variation",
      x = "") +
    facet_wrap(~Continent, scales = "free_y") +
    scale_y_continuous(limits = c(0, 1),
      breaks = pretty_breaks(n = 3)) +

    coord_flip() +
    theme_AP() +
    guides(color = guide_legend(nrow = 2, byrow = TRUE))
}

dd

## [[1]]

```



[[2]]



4.2 Ranges

```
# COMPUTE RANGES -----

calc <- uncertainty.dt[, .(min = min(V1), max = max(V1)), .(Continent, Country)] %>%
  .[, .(range = max - min), .(Continent, Country)] %>%
  .[order(range)]

print(calc, n = Inf)
```

```
##      Continent      Country      range
##  1:      Asia      Cyprus 0.4639176
##  2:      Asia United Arab Emirates 0.4854683
```

##	3:	Asia	Israel	0.4944147
##	4:	Asia	Jordan	0.5042251
##	5:	Africa	Tunisia	0.5706087
##	6:	Asia	Saudi Arabia	0.5707274
##	7:	Europe	Italy	0.5721363
##	8:	Africa	Swaziland	0.5726990
##	9:	Americas	United States	0.5775350
##	10:	Europe	Netherlands	0.5782524
##	11:	Americas	Cuba	0.5818560
##	12:	Africa	Mozambique	0.5822536
##	13:	Americas	Brazil	0.5855978
##	14:	Africa	Benin	0.5856878
##	15:	Africa	Namibia	0.5856878
##	16:	Africa	Zimbabwe	0.5856878
##	17:	Asia	Lebanon	0.5864389
##	18:	Europe	France	0.5869800
##	19:	Africa	South Africa	0.5876042
##	20:	Africa	Zambia	0.5890368
##	21:	Asia	Kazakhstan	0.5894648
##	22:	Africa	Ivory Coast	0.5903014
##	23:	Asia	Brunei	0.5922976
##	24:	Americas	Canada	0.5944970
##	25:	Europe	Spain	0.5984780
##	26:	Asia	Kuwait	0.5998517
##	27:	Europe	Austria	0.6014893
##	28:	Europe	Belgium	0.6014893
##	29:	<NA>	Byelarus	0.6014893
##	30:	Europe	Denmark	0.6014893
##	31:	Europe	Finland	0.6014893
##	32:	Europe	Germany	0.6014893
##	33:	Europe	Greece	0.6014893
##	34:	Europe	Hungary	0.6014893
##	35:	Europe	Latvia	0.6014893
##	36:	Europe	Lithuania	0.6014893
##	37:	Europe	Luxembourg	0.6014893
##	38:	Africa	Malawi	0.6014893
##	39:	Europe	Slovakia	0.6014893
##	40:	Europe	Sweden	0.6014893
##	41:	Europe	Switzerland	0.6014893
##	42:	Africa	Botswana	0.6071124
##	43:	Europe	Ukraine	0.6122925
##	44:	Asia	Qatar	0.6222104
##	45:	Europe	Czech Republic	0.6394465
##	46:	Europe	Croatia	0.6449088
##	47:	Europe	Russia	0.6468832
##	48:	Europe	United Kingdom	0.6527024
##	49:	Europe	Slovenia	0.6529393
##	50:	Europe	Romania	0.6658015

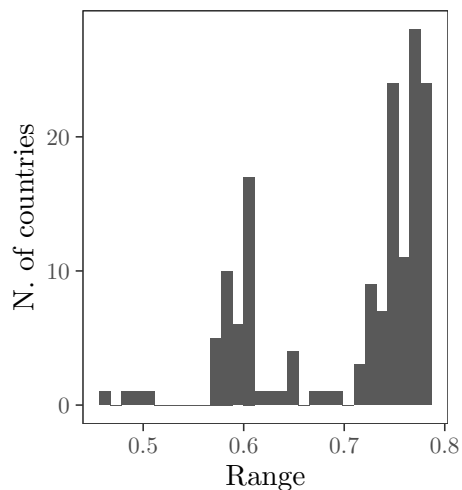
## 51:	Africa	Burkina Faso	0.6810580
## 52:	Africa	Libya	0.6956438
## 53:	Asia	Azerbaijan	0.7146797
## 54:	Asia	Syria	0.7165713
## 55:	Africa	Algeria	0.7176049
## 56:	Asia	Iraq	0.7217608
## 57:	Africa	Morocco	0.7219235
## 58:	Oceania	New Zealand	0.7243644
## 59:	Asia	Turkey	0.7249771
## 60:	Asia	Armenia	0.7257428
## 61:	Oceania	Australia	0.7277279
## 62:	Africa	Egypt	0.7287327
## 63:	Americas	Mexico	0.7295769
## 64:	Asia	China	0.7310074
## 65:	Asia	Iran	0.7320971
## 66:	Americas	Chile	0.7341855
## 67:	Asia	India	0.7368912
## 68:	Asia	Japan	0.7370818
## 69:	Europe	Bulgaria	0.7393857
## 70:	Asia	Burma	0.7407613
## 71:	Asia	Afghanistan	0.7408257
## 72:	Asia	Philippines	0.7431090
## 73:	Americas	Argentina	0.7460277
## 74:	Asia	Turkmenistan	0.7470244
## 75:	Asia	Vietnam	0.7474112
## 76:	Asia	Pakistan	0.7475370
## 77:	Americas	Ecuador	0.7475424
## 78:	Europe	Macedonia	0.7477843
## 79:	Asia	North Korea	0.7482417
## 80:	Americas	Peru	0.7485998
## 81:	Asia	Uzbekistan	0.7486592
## 82:	Europe	Moldova	0.7488747
## 83:	Africa	Chad	0.7498686
## 84:	Americas	Paraguay	0.7498686
## 85:	Americas	Uruguay	0.7498686
## 86:	Africa	Sudan	0.7498884
## 87:	Asia	Malaysia	0.7503642
## 88:	Europe	Portugal	0.7505188
## 89:	Asia	Thailand	0.7505896
## 90:	Americas	Bolivia	0.7506488
## 91:	Asia	Bangladesh	0.7508831
## 92:	Americas	Venezuela	0.7512531
## 93:	Asia	Sri Lanka	0.7513038
## 94:	Asia	Nepal	0.7520751
## 95:	Africa	Cameroon	0.7527058
## 96:	Asia	Kyrgyzstan	0.7546852
## 97:	Asia	Indonesia	0.7566068
## 98:	Americas	Guyana	0.7568347

## 99:	Asia	Tajikistan	0.7569981
## 100:	Asia	Georgia	0.7580011
## 101:	Asia	South Korea	0.7584425
## 102:	Asia	Cambodia	0.7615212
## 103:	Africa	Ethiopia	0.7628903
## 104:	Europe	Albania	0.7632624
## 105:	Asia	Yemen	0.7644014
## 106:	Africa	Tanzania	0.7646283
## 107:	Africa	Nigeria	0.7656979
## 108:	Americas	Haiti	0.7658826
## 109:	Asia	Oman	0.7711095
## 110:	Americas	Belize	0.7728079
## 111:	Europe	Bosnia and Herzegovina	0.7728079
## 112:	Africa	Congo	0.7728079
## 113:	Americas	French Guiana	0.7728079
## 114:	Africa	Gabon	0.7728079
## 115:	Africa	Ghana	0.7728079
## 116:	Americas	Guatemala	0.7728079
## 117:	Africa	Guinea	0.7728079
## 118:	Americas	Jamaica	0.7728079
## 119:	Africa	Lesotho	0.7728079
## 120:	Africa	Mauritania	0.7728079
## 121:	Americas	Panama	0.7728079
## 122:	Oceania	Papua New Guinea	0.7728079
## 123:	Europe	Poland	0.7728079
## 124:	Americas	Puerto Rico	0.7728079
## 125:	Africa	Togo	0.7728079
## 126:	Americas	Trinidad	0.7728079
## 127:	Africa	Uganda	0.7728079
## 128:	Africa	Western Sahara	0.7728079
## 129:	Africa	Zaire	0.7728079
## 130:	Europe	Serbia	0.7733555
## 131:	Americas	El Salvador	0.7743265
## 132:	Africa	Kenya	0.7743758
## 133:	Americas	Costa Rica	0.7745177
## 134:	Asia	Mongolia	0.7749708
## 135:	Americas	Colombia	0.7779004
## 136:	Africa	Senegal	0.7793409
## 137:	Americas	Nicaragua	0.7803721
## 138:	Africa	Madagascar	0.7806578
## 139:	Africa	Mali	0.7808877
## 140:	Africa	Angola	0.7814033
## 141:	Asia	Bhutan	0.7814033
## 142:	Africa	Burundi	0.7814033
## 143:	Africa	Central African Republic	0.7814033
## 144:	Africa	Djibouti	0.7814033
## 145:	Americas	Dominican Republic	0.7814033
## 146:	Africa	Equatorial Guinea	0.7814033

```
## 147:    Africa                      Eritrea 0.7814033
## 148:    Africa                      Gambia 0.7814033
## 149:    Africa                      Guinea-Bissau 0.7814033
## 150:  Americas                      Honduras 0.7814033
## 151:    Asia                        Laos 0.7814033
## 152:    Africa                      Liberia 0.7814033
## 153:    Africa                      Niger 0.7814033
## 154:    Europe                      Norway 0.7814033
## 155:    Africa                      Rwanda 0.7814033
## 156:    Africa                      Sierra Leone 0.7814033
## 157:    Africa                      Somalia 0.7814033
## 158:  Americas                      Suriname 0.7830037
##      Continent                      Country      range
```

```
ggplot(calc, aes(range)) +
  geom_histogram() +
  labs(x = "Range", y = "N. of countries") +
  theme_AP()
```

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



```
# COMPARE RANGES -----
ranges_empirical <- uncertainty.dt[, .(higher = max(V1), lower = min(V1)), IFT] %>%
  .[, Study:= "This study"]%>%
  .[!IFT == "Jager"]

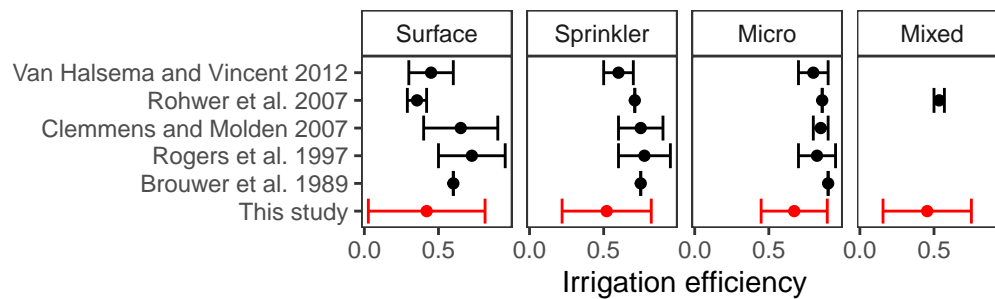
ranges_efficiencies <- fread("ranges_efficiencies.csv")

rbind(ranges_empirical, ranges_efficiencies)[, mean.value:= (higher + lower) / 2] %>%
  .[, Study:= factor(Study, levels = c("This study",
    "Brouwer et al. 1989",
    "Rogers et al. 1997",
    "Clemmens and Molden 2007",
    "Rohwer et al. 2007",
```

```

"Van Halsema and Vincent 2012"))] %>%
na.omit() %>%
ggplot(., aes(mean.value, Study, color = ifelse(Study == "This study", "red", "black"))) +
geom_point() +
scale_x_continuous(breaks = pretty_breaks(n = 3)) +
geom_errorbar(aes(xmin = lower, xmax = higher)) +
scale_color_identity() +
facet_wrap(~IFT, ncol = 4) +
labs(x = "Irrigation efficiency", y = "") +
theme_AP()

```



4.3 Overlap between irrigation efficiencies

```

# CHECK OVERLAP -----

dd <- uncertainty.dt[!Continent == "Oceania"][Study == "Rohwer et al. approach"] %>%
  split(., .$Continent, drop = TRUE)

overlap.dt <- lapply(dd, function(x) split(x, x$IFT, drop = TRUE)) %>%
  lapply(., function(x) lapply(x, function(y) y[, V1])) %>%
  lapply(., function(x) overlap(x)$OV)

overlap.dt

## $Africa
## Surface-Sprinkler      Surface-Mixed      Sprinkler-Mixed
##           0.3329115           0.5084454           0.5255184
##
## $Americas
## Surface-Mixed
##           0.5084078
##
## $Asia
## Surface-Micro Surface-Mixed      Micro-Mixed
##           0.05314604           0.42996793           0.07870342
##
## $Europe
## Surface-Sprinkler      Surface-Mixed      Sprinkler-Mixed
##           0.3372350           0.5362841           0.4917646

```

```

ff <- uncertainty.dt[!Continent == "Oceania"] %>%
  .[Country %in% intersect(rohwer[, Country], jager[, Country])] %>%
  split(., .$Country, drop = TRUE) %>%
  lapply(., function(x) split(x, x$Approach, drop = TRUE)) %>%
  lapply(., function(x) lapply(x, function(y) y[, V1])) %>%
  lapply(., function(x) overlap(x)$OV) %>%
  lapply(., data.table) %>%
  rbindlist(., idcol = "Country") %>%
  .[, Continent := countrycode(., Country,
                              origin = "country.name",
                              destination = "continent")]

ff[, .(median = median(V1)), Continent]

##      Continent      median
## 1:      Asia 0.9198489
## 2:     Europe 0.7806813
## 3:     Africa 0.9335912
## 4:  Americas 0.8630331

list_continents <- list(c("Africa", "Asia"), c("Americas", "Europe"))

# PLOT OVERLAP -----

dd <- list()
for(i in 1:length(list_continents)) {
  dd[[i]] <- ff[Continent %in% list_continents[[i]]] %>%
    ggplot(., aes(reorder(Country, V1), V1)) +
    geom_point() +
    scale_color_discrete(name = "GM") +
    labs(y = "Overlap", x = "") +
    facet_wrap(~Continent, scales = "free_y") +
    coord_flip() +
    theme_AP()
}

dd

## [[1]]

```

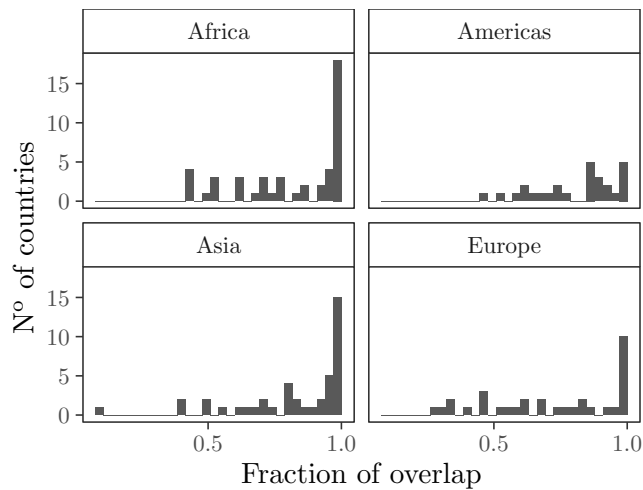


[[2]]

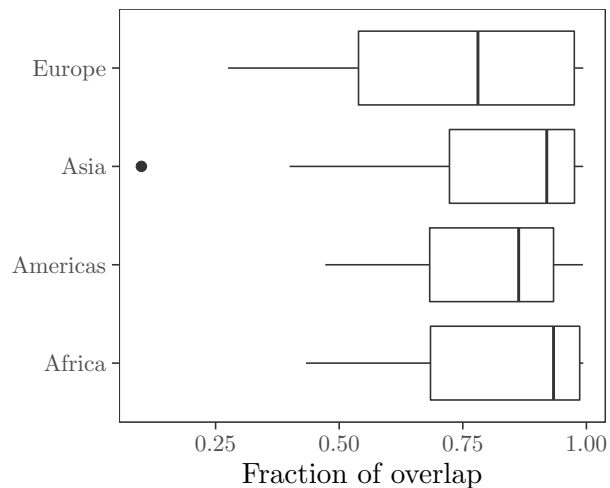


PLOT OVERLAP AS HISTOGRAMS AND BOXPLOTS

```
ggplot(ff, aes(V1)) +
  geom_histogram() +
  facet_wrap(~Continent) +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP() +
  labs(x = "Fraction of overlap", y = "Nº of countries")
```



```
ggplot(ff, aes(Continent, V1)) +
  geom_boxplot() +
  coord_flip() +
  theme_AP() +
  labs(y = "Fraction of overlap", x = "")
```



CHECK CORRESPONDENCE BETWEEN SHARES OF IFT AND PREDOMINANT TECHNOLOGY -----

Retrieve countries where overlap is <0.3

```
merge(jager, rohwer, by = c("Country")) %>%
  .[Country %in% ff[V1 < 0.3][, Country]] %>%
  .[, .(Country, Surface_fraction, Sprinkler_fraction, Drip_fraction, IFT)]
```

```
##      Country Surface_fraction Sprinkler_fraction Drip_fraction   IFT
## 1:      Oman           0.793           0.113         0.094 Micro
## 2:  Slovenia           0.000           0.693         0.307 Mixed
## 3:      Spain           0.297           0.226         0.478 Mixed
```

PLOT UNCERTAINTY -----

```
gg <- list()
```

```

for (i in 1:length(list_continents)) {
  gg[[i]] <- ggplot(uncertainty.dt[Continent %in% list_continents[[i]]],
    aes(x = V1, y = fct_reorder(Country, V1), fill = Study)) +
    geom_density_ridges(scale = 2, alpha = 0.3) +
    labs(x = "Irrigation efficiency", y = "") +
    facet_wrap(~Continent, scales = "free") +
    scale_x_continuous(breaks = pretty_breaks(n = 3),
      limits = c(0, 1)) +
    scale_fill_manual(values = wes_palette("Chevalier1")) +
    theme_AP() +
    theme(legend.position = "top") +
    guides(fill = guide_legend(nrow = 2, byrow = TRUE))
}

```

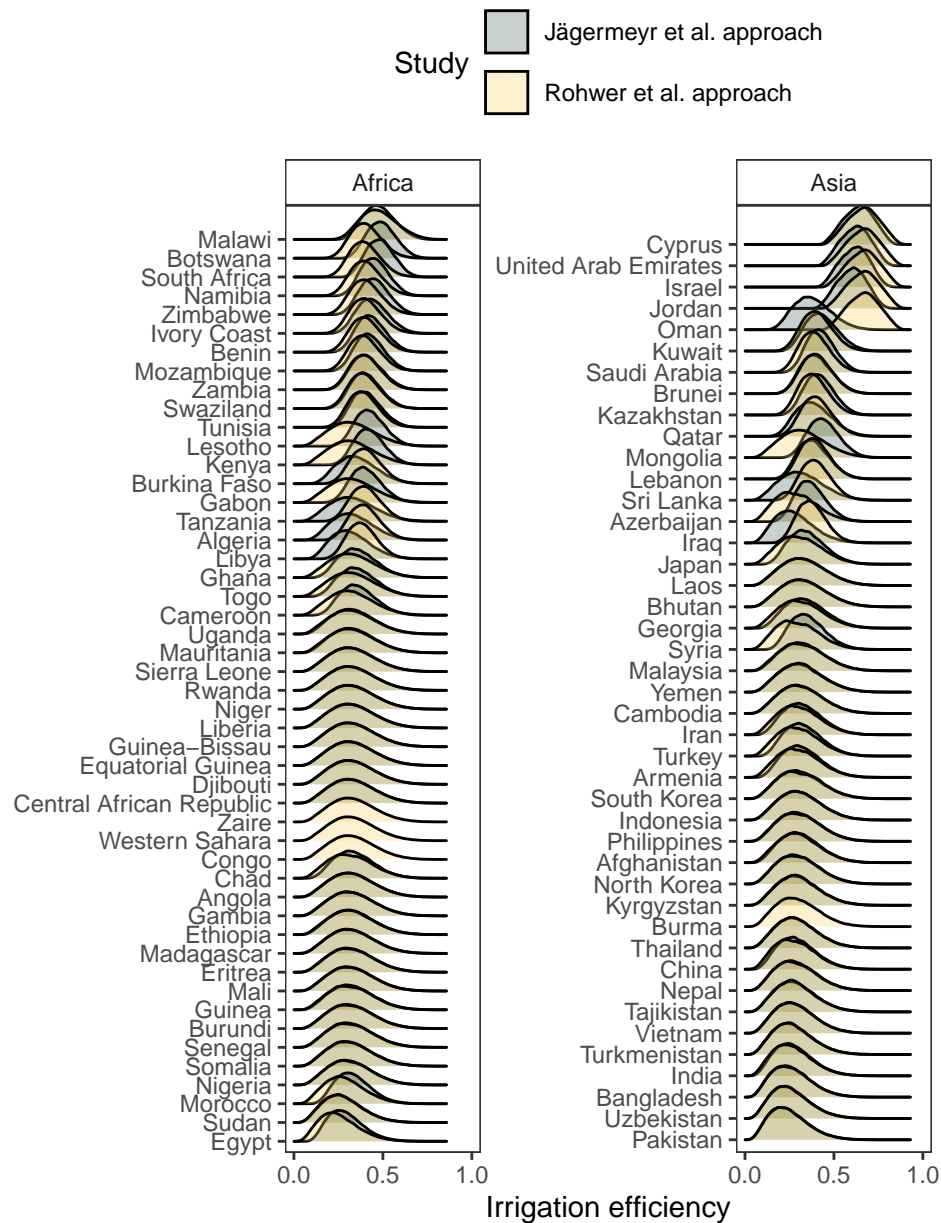
MERGE PLOTS -----

```
gg
```

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

```
## Picking joint bandwidth of 0.0121
```

```
## Picking joint bandwidth of 0.0118
```

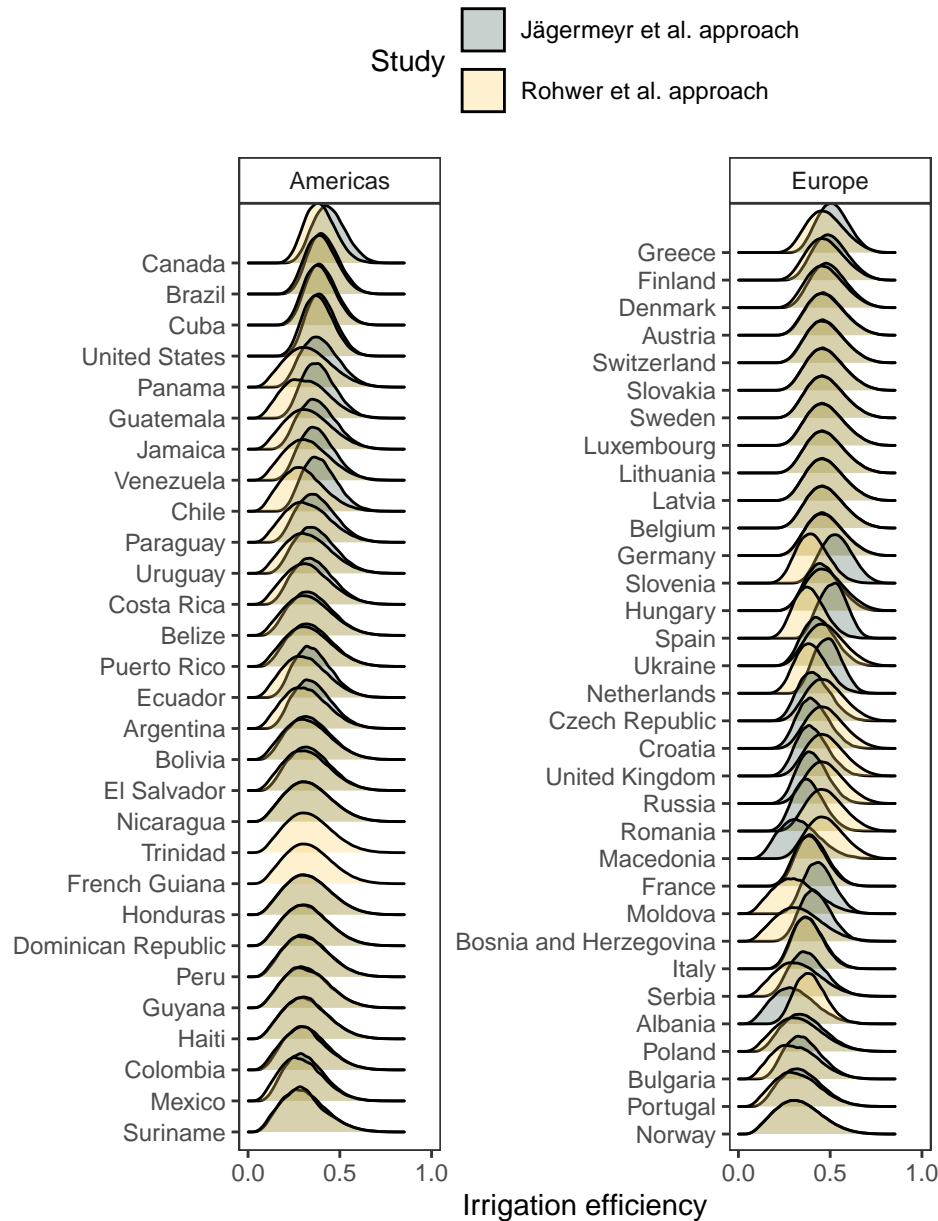


```
##
```

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

```
## Picking joint bandwidth of 0.0123
```

```
## Picking joint bandwidth of 0.0109
```



PLOT UNCERTAINTY IN EACH IRRIGATION TECHNOLOGY -----

```
gg <- list()

for(i in 1:length(list_continents)) {
  gg[[i]] <- uncertainty.dt[Approach == "Rohwer et al. 2007"][Continent %in% list_continents[i]]
  ggplot(., aes(x = V1, y = fct_reorder(Country, V1), fill = IFT)) +
    geom_density_ridges(scale = 2, alpha = 0.3) +
    labs(x = "Irrigation efficiency", y = "") +
    facet_wrap(~Continent, scales = "free") +
    scale_x_continuous(breaks = pretty_breaks(n = 3),
                      limits = c(0, 1)) +
    theme_AP() +

```

```

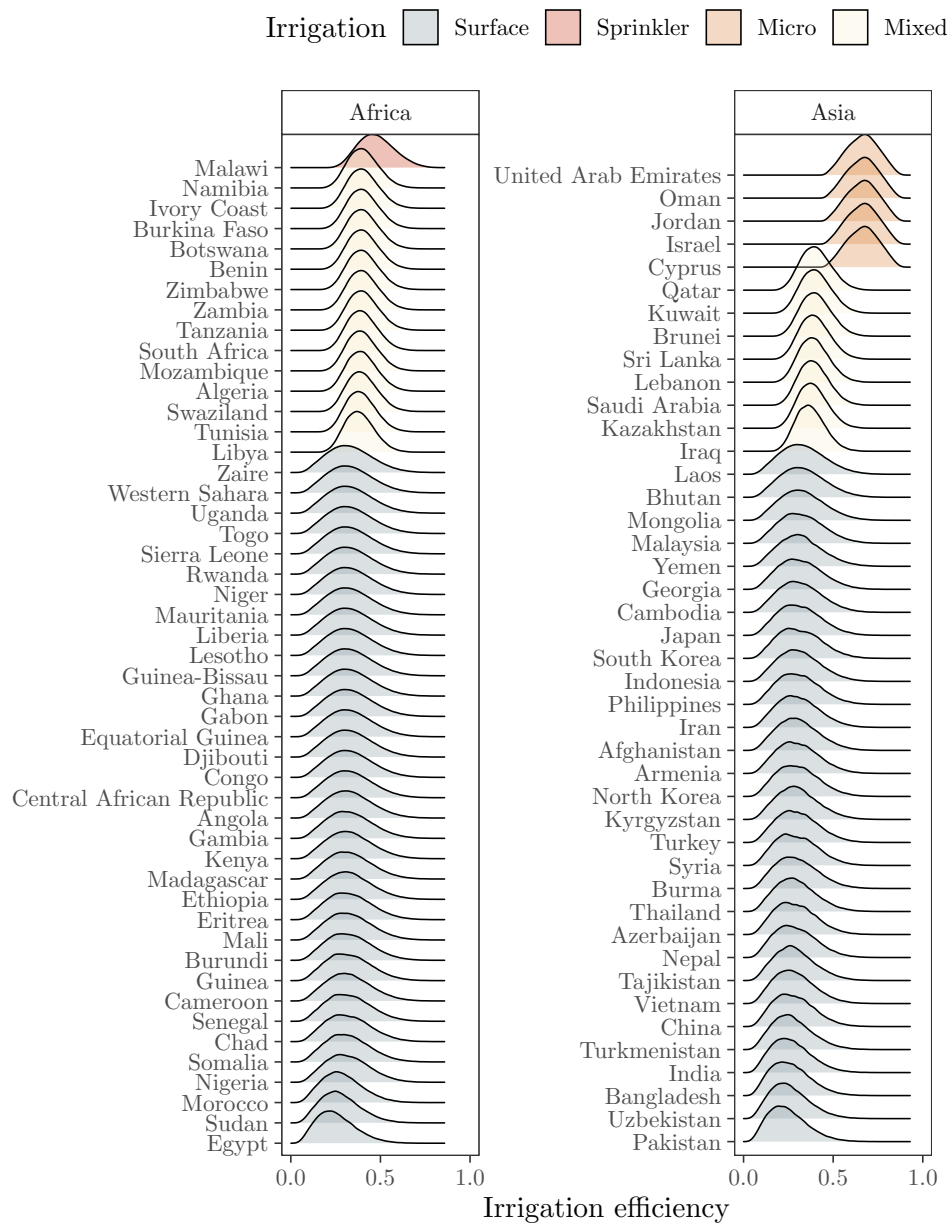
    theme(legend.position = "top")
  if(i == 1) {
    gg[[i]] <- gg[[i]] +
      scale_fill_manual(values = c("#899DA4", "#C93312", "#DC863B", "#FAEFD1"),
                        labels = c("Surface", "Sprinkler", "Micro", "Mixed"),
                        name = "Irrigation")
  } else if(i == 2) {
    gg[[i]] <- gg[[i]] +
      scale_fill_manual(values = c("#899DA4", "#C93312", "#FAEFD1"),
                        labels = c("Surface", "Sprinkler", "Mixed"),
                        name = "Irrigation")
  }
}
gg

```

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

```
## Picking joint bandwidth of 0.0123
```

```
## Picking joint bandwidth of 0.0119
```

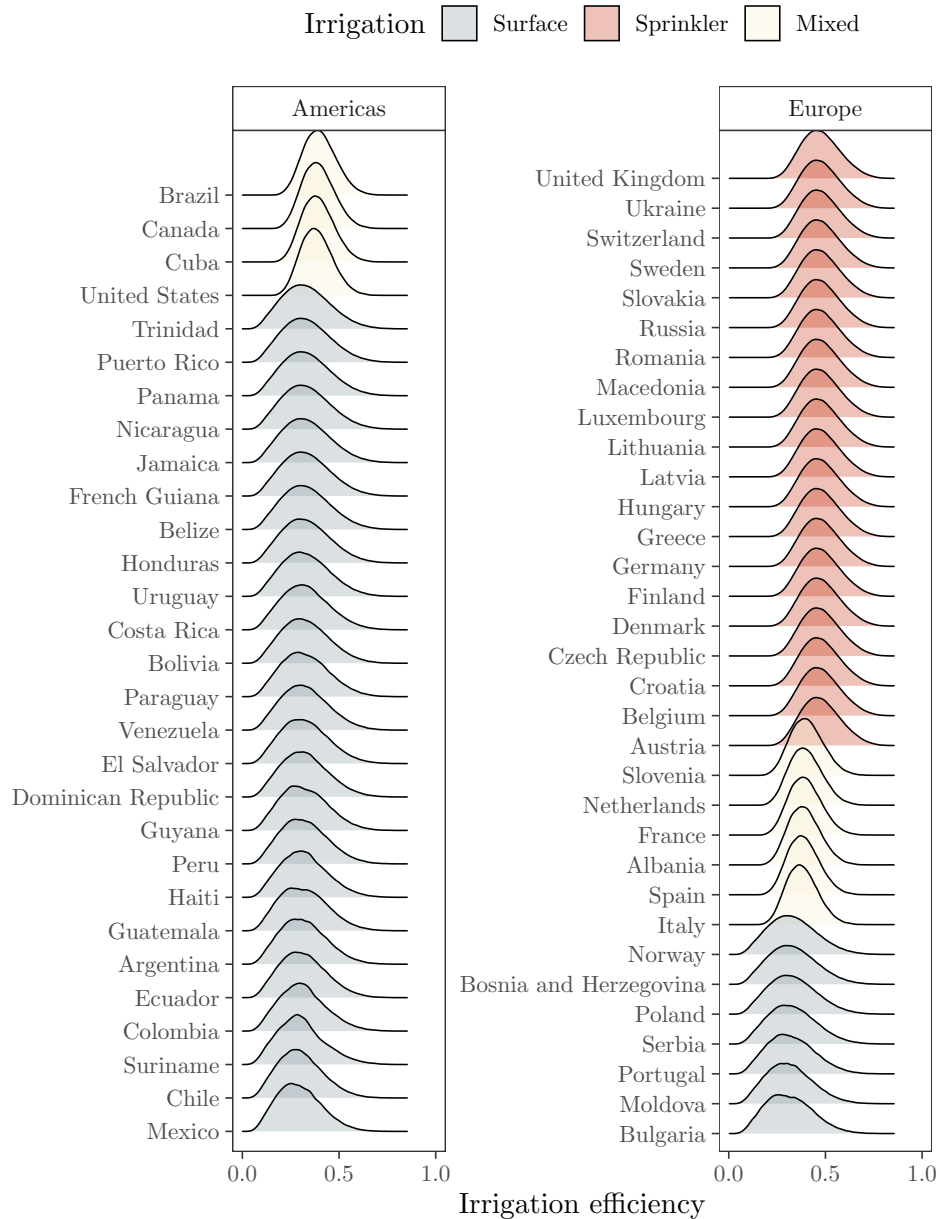


##

[[2]]

Picking joint bandwidth of 0.0129

Picking joint bandwidth of 0.0113



```
# PLOT ROHWER ET AL.'S IRRIGATION EFFICIENCY VALUES -----

rohwer[, Continent:= countrycode(rohwer[, Country], origin = "country.name",
                                destination = "continent")]

## Warning in countrycode_convert(sourcevar = sourcevar, origin = origin, destination = dest,
dd <- list()
for (i in 1:length(list_continents)) {
  dd[[i]] <- ggplot(rohwer[Continent %in% list_continents[[i]]],
                    aes(x = Ep, y = fct_reorder(Country, Ep), color = IFT)) +
    geom_point() +
    labs(x = "Irrigation efficiency", y = "") +
    scale_x_continuous(breaks = pretty_breaks(n = 3),
```



```

    limits = c(0, 1)) +
  facet_wrap(~Continent, scales = "free") +
  scale_color_discrete(name = "Irrigation") +
  theme_AP()
}

```

```
dd
```

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

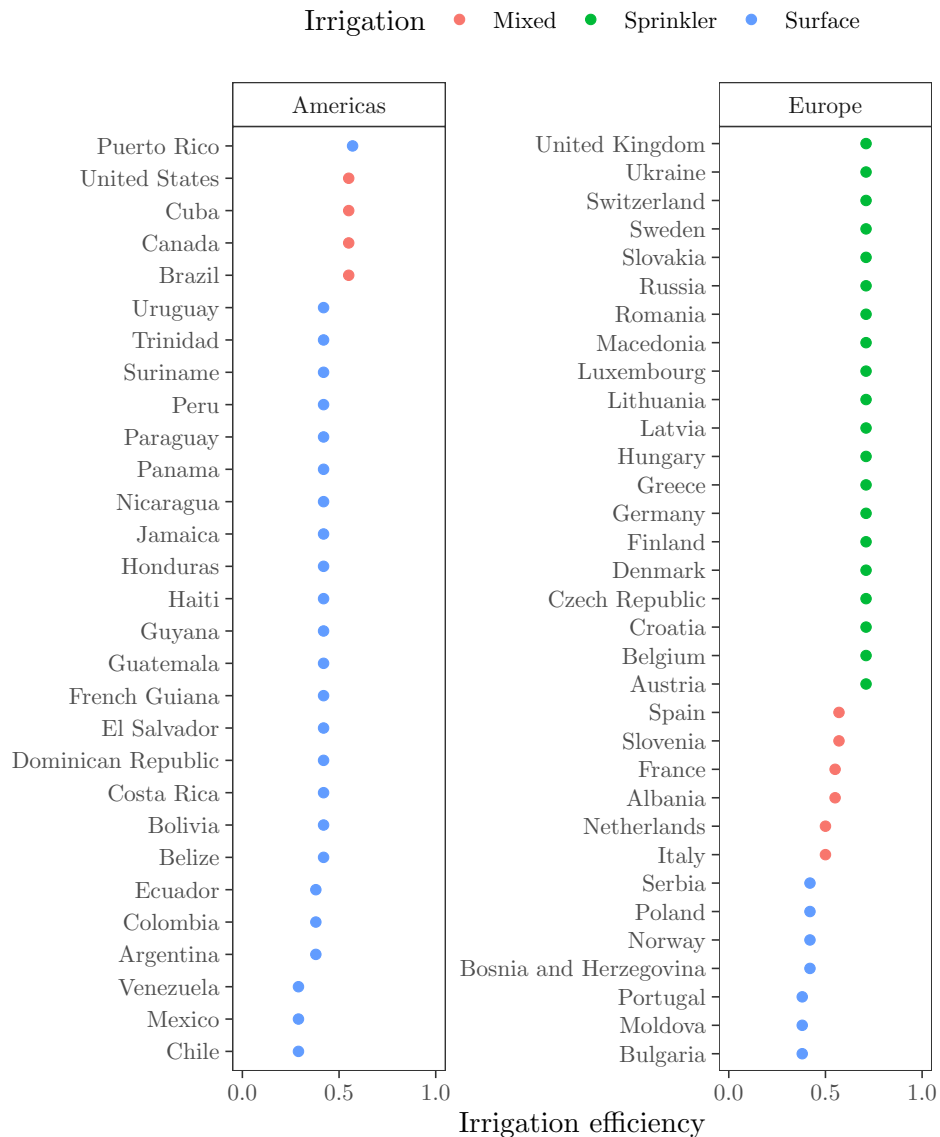
```
## Warning: Removed 1 rows containing missing values (geom_point).
```

Irrigation ● Micro ● Mixed ● Sprinkler ● Surface



```
##
```

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



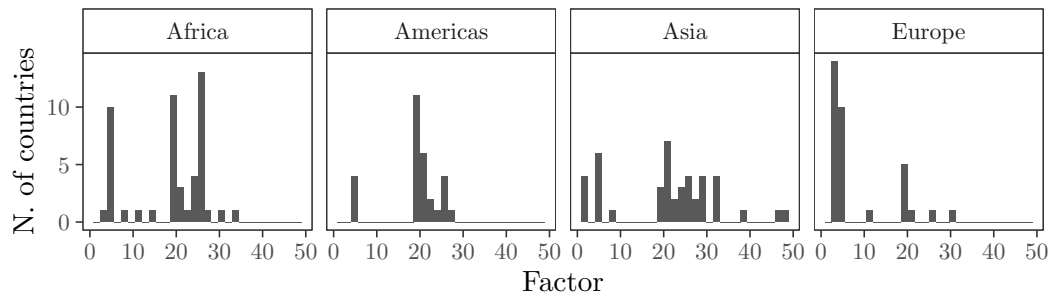
```
# CALCULATE THE UNCERTAINTY IN THE RANGES -----

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

factor_unc <- uncertainty.dt[, .(min = min(V1), max = max(V1)), .(Continent, Country)] %>%
  .[Continent %in% selection_continents] %>%
  .[, factor:= max / min]

ggplot(factor_unc, aes(factor)) +
  geom_histogram() +
  facet_wrap(~Continent, ncol = 4) +
  labs(x = "Factor", y = "N. of countries") +
  theme_AP()

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



Number of countries whose irrigation water withdrawals fluctuate a factor of x
due to uncertainty in irrigation efficiency

```
factor_unc %>%
  .[, factor:= floor(max / min)] %>%
  .[, .(number.countries = .N), factor] %>%
  .[order(factor)] %>%
  print()
```

```
##      factor number.countries
## 1:      2              4
## 2:      3             15
## 3:      4             28
## 4:      5              2
## 5:      7              1
## 6:      8              1
## 7:     11              2
## 8:     13              1
## 9:     19             29
##10:     20             16
##11:     21              4
##12:     22              3
##13:     23              3
##14:     24              5
##15:     25             20
##16:     26              3
##17:     27              3
##18:     28              3
##19:     29              1
##20:     30              1
##21:     31              3
##22:     32              2
##23:     33              1
##24:     39              1
##25:     47              1
##26:     48              1
##      factor number.countries
```

4.4 Retrieve data from ISIMIP

```
# FUNCTIONS TO EXTRACT DATA FROM .NC FILES -----

coords2country = function(points) {
  countriesSP <- rworldmap::getMap(resolution = 'low')
  pointsSP = sp::SpatialPoints(points, proj4string=CRS(proj4string(countriesSP)))
  indices = sp::over(pointsSP, countriesSP)
  indices$ADMIN
}

# 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"]
  setcolorder(out, c("Country", "Continent", "Code", "Water.Withdrawn"))
})
return(out)
}

# READ IN NC FILES -----

# Define settings
vecs <- 1:((2010 - 1970) * 12)
vec <- split(vecs, ceiling(seq_along(vecs) / 12))
names(vec) <- 1971:2010
selected.years <- "2010"
dname <- "pirrww"

files <- list("h08_wfdei_nobc_hist_varsoc_co2_pirrww_global_monthly_1971_2010.nc",
              "pcr-globwb_wfdei_nobc_hist_varsoc_co2_pirrww_global_monthly_1971_2010.nc",
              "lpjml_wfdei_nobc_hist_varsoc_co2_pirrww_global_monthly_1971_2010.nc",
              "watergap2_wfdei_nobc_hist_varsoc_co2_pirrww_global_monthly_1971_2010.nc")

names.isimip <- c("H08", "PCR-GLOBWB", "LPJmL", "WaterGap")

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

# EXTRACT CORRECTIVE COEFFICIENTS FOR IRRIGATION EFFICIENCY FOR LPJML -----

ncin <- nc_open("irrigation_project_efficiencies.nc")
lon <- ncvar_get(ncin, "lon")
lat <- ncvar_get(ncin, "lat")
tmp_array <- ncvar_get(ncin)
lonlat <- as.matrix(expand.grid(lon, lat))
da <- na.omit(cbind(lonlat, as.vector(tmp_array))) %>%
  data.frame() %>%
  na.omit()
Country <- coords2country(da[1:nrow(da), 1:2])
lpjml_efficiencies <- cbind(Country, da) %>%
  na.omit() %>%
  data.table() %>%
  .[, .(Ep = mean(V3)), Country]

# ARRANGE NC FILES -----

names(isimip.dt) <- names.isimip

isimip.dt <- lapply(isimip.dt, function(x) rbindlist(x)) %>%

```

```

rbindlist(., idcol = "Model") %>%
na.omit() %>%
# To correct for duplicate country in Cyprus
.[, .(Water.Withdrawn = mean(Water.Withdrawn)), .(Model, Country, Continent, Code)]

lpjml_harmonized <- merge(isimip.dt[Model == "LPJmL"], lpjml_efficiencys, all.x = TRUE) %>%
.[, Water.Withdrawn:= Water.Withdrawn * Ep] %>%
.[, Ep:= NULL]

isimip.dt <- rbind(isimip.dt[!Model == "LPJmL"], lpjml_harmonized)

fwrite(isimip.dt, "isimip.dt")

# MERGE UNCERTAINTY IN EP WITH ISIMIP DATA -----

efficiency.dt <- copy(uncertainty.dt) %>%
  setnames(., "V1", "Ep")

ghm.dt <- dcast(isimip.dt, Country + Continent + Code ~ Model, value.var = "Water.Withdrawn")
full.dt <- merge(efficiency.dt, ghm.dt, by = c("Country", "Continent"), all.x = TRUE) %>%
  .[, (names.isimip):= lapply(.SD, function(x) x / Ep), .SDcols = names.isimip]
tmp.dt <- melt(full.dt, measure.vars = names.isimip, variable.name = "Model",
  value.name = "IWW_corrected")
ghm.large <- melt(ghm.dt, measure.vars = names.isimip, variable.name = "Model",
  value.name = "IWW")
gm.uncertainty <- tmp.dt[, .(min = min(IWW_corrected), max = max(IWW_corrected)),
  .(Country, Continent, Model)]
gm.dt <- merge(ghm.large, gm.uncertainty)

```

4.5 Retrieve data from ISIMIP (climate change in 2050)

```

# READ IN FILES ON CLIMATE CHANGE UNCERTAINTY (2050) -----

files <- list(
  "watergap2_miroc5_ewembi_rcp85_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "watergap2_miroc5_ewembi_rcp60_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "watergap2_miroc5_ewembi_rcp45_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "watergap2_miroc5_ewembi_rcp26_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp85_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp60_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "lpjml_miroc5_ewembi_rcp26_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "pcr-globwb_miroc5_ewembi_rcp60_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "pcr-globwb_miroc5_ewembi_rcp26_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp85_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp60_2005soc_co2_pirrww_global_monthly_2006_2099.nc",
  "h08_miroc5_ewembi_rcp26_2005soc_co2_pirrww_global_monthly_2006_2099.nc"
)

```

```

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

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

# ARRANGE DATASETS -----

ghms <- c(rep("WaterGap", times = 4),
          rep("LPJmL", times = 3),
          rep("PCR-GLOBWB", times = 2),
          rep("H08", times = 3))

climate_scenario <- c(85, 60, 45, 26, 85, 60, 26, 60, 26, 85, 60, 26)
names.isimip <- paste(ghms, climate_scenario, sep = "/")

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

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

# Arrange data
isimip.climate.dt <- lapply(isimip.climate, function(x) rbindlist(x, idcol = "Year")) %>%
  rbindlist(., idcol = "Model") %>%
  .[!Continent == "Oceania"] %>%
  separate(., "Model", c("Model", "Climate scenario"), "/") %>%
  na.omit() %>%
  .[Year == 2050]

# Export
fwrite(isimip.climate.dt, "isimip.climate.dt.csv")

# PLOT RANGES OF STRUCTURAL UNCERTAINTY AND RANGES OF
# STRUCTURAL UNCERTAINTY + UNCERTAINTY IN IRRIGATION EFFICIENCY +
# UNCERTAINTY IN CLIMATE CHANGE -----

countries_list <- c("Egypt", "Sudan", "South Africa", "Morocco", "Madagascar",
                    "United States", "Mexico", "Brazil", "Chile", "Peru",

```

```

      "India", "China", "Pakistan", "Iran", "Indonesia",
      "Italy", "Spain", "France", "Ukraine", "Romania")

range.gm <- gm.dt %>%
  .[, .(min = min(IWW, na.rm = TRUE), max = max(IWW, na.rm = TRUE)),
    .(Country, Continent)] %>%
  .[, Approach:= "GM"]

range.study <- gm.dt %>%
  .[, .(min = min(min, na.rm = TRUE), max = max(max, na.rm = TRUE)),
    .(Country, Continent)] %>%
  .[, Approach:= "GM + uncertainty in irrigation efficiency"]

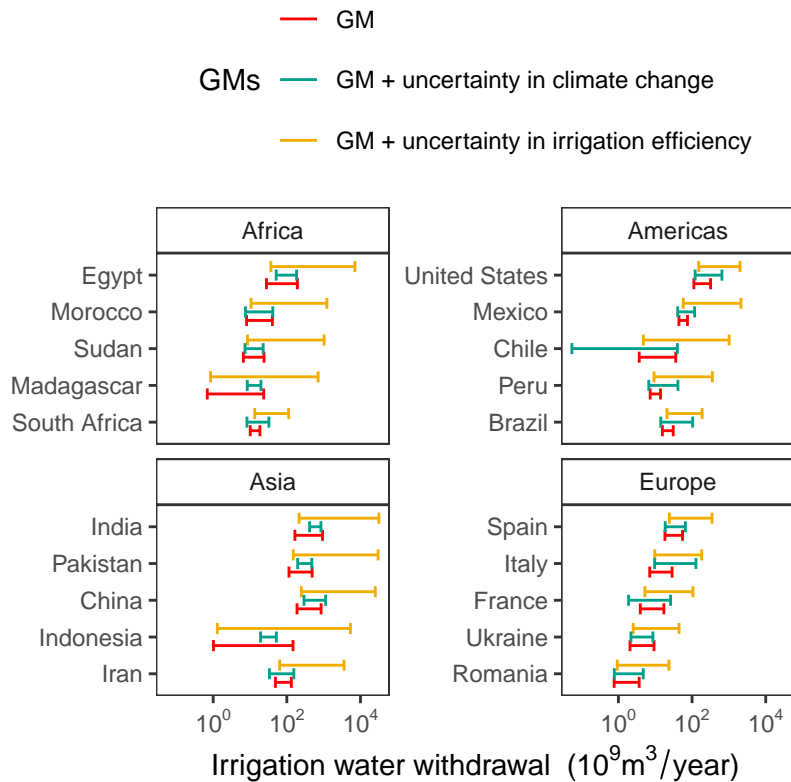
range.climate <- isimip.climate.dt %>%
  .[, .(min = min(Water.Withdrawn), max = max(Water.Withdrawn)),
    .(Country, Continent)] %>%
  .[, Approach:= "GM + uncertainty in climate change"]

all.uncertainties <- rbind(range.gm, range.study, range.climate) %>%
  .[, mean:= (min + max) / 2]

# Substitute 0 by NA -----
all.uncertainties[all.uncertainties == 0] <- NA

all.uncertainties %>%
  .[Country %in% countries_list] %>%
  ggplot(., aes(reorder(Country, mean), mean, color = Approach)) +
  geom_errorbar(aes(ymin = min,
                    ymax = max),
                position = position_dodge(0.7)) +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_color_manual(name = "GMs", values = wes_palette("Darjeeling1")) +
  labs(y = expression(paste("Irrigation water withdrawal ", " ", "(", 10^9, m^3/year, " ", ")")),
       x = "") +
  facet_wrap(~Continent, scales = "free_y") +
  coord_flip() +
  theme_AP() +
  guides(color = guide_legend(nrow = 3, byrow = TRUE))

```

```
# EXPORT -----
fwrite(all.uncertainties, "all.uncertainties.csv")

# PLOT RANGES OF STRUCTURAL UNCERTAINTY AND RANGES OF
# STRUCTURAL UNCERTAINTY + UNCERTAINTY IN IRRIGATION EFFICIENCY (COMPLETE) -----

vec1 <- all.uncertainties[Approach == "GM", Country]
vec2 <- all.uncertainties[Approach == "GM + uncertainty in climate change", Country]
vec3 <- all.uncertainties[Approach == "GM + uncertainty in irrigation efficiency", Country]
common_countries <- Reduce(intersect, list(vec1, vec2, vec3))

dd <- list()
for (i in 1:length(list_continents)) {
  dd[[i]] <- all.uncertainties %>%
    .[Country %in% common_countries] %>%
    na.omit() %>%
    .[Continent %in% list_continents[[i]]] %>%
    ggplot(., aes(reorder(Country, mean), mean, color = Approach)) +
    geom_errorbar(aes(ymin = min,
                      ymax = max),
                  position = position_dodge(0.7)) +
    scale_y_log10(breaks = trans_breaks("log10", function(x) 10 ^ (2 * x)),
                  labels = trans_format("log10", math_format(10 ^ .x))) +
    scale_color_manual(name = "GM", values = wes_palette("Darjeeling1")) +
    labs(y = expression(paste("Irrigation water withdrawal ", " ", "(", 10^9, " m^3/year, " ", " ")),
         x = "") +
}
```

```

facet_wrap(~Continent, scales = "free_y") +
coord_flip() +
theme_AP() +
guides(color = guide_legend(nrow = 3, byrow = TRUE))
}

```

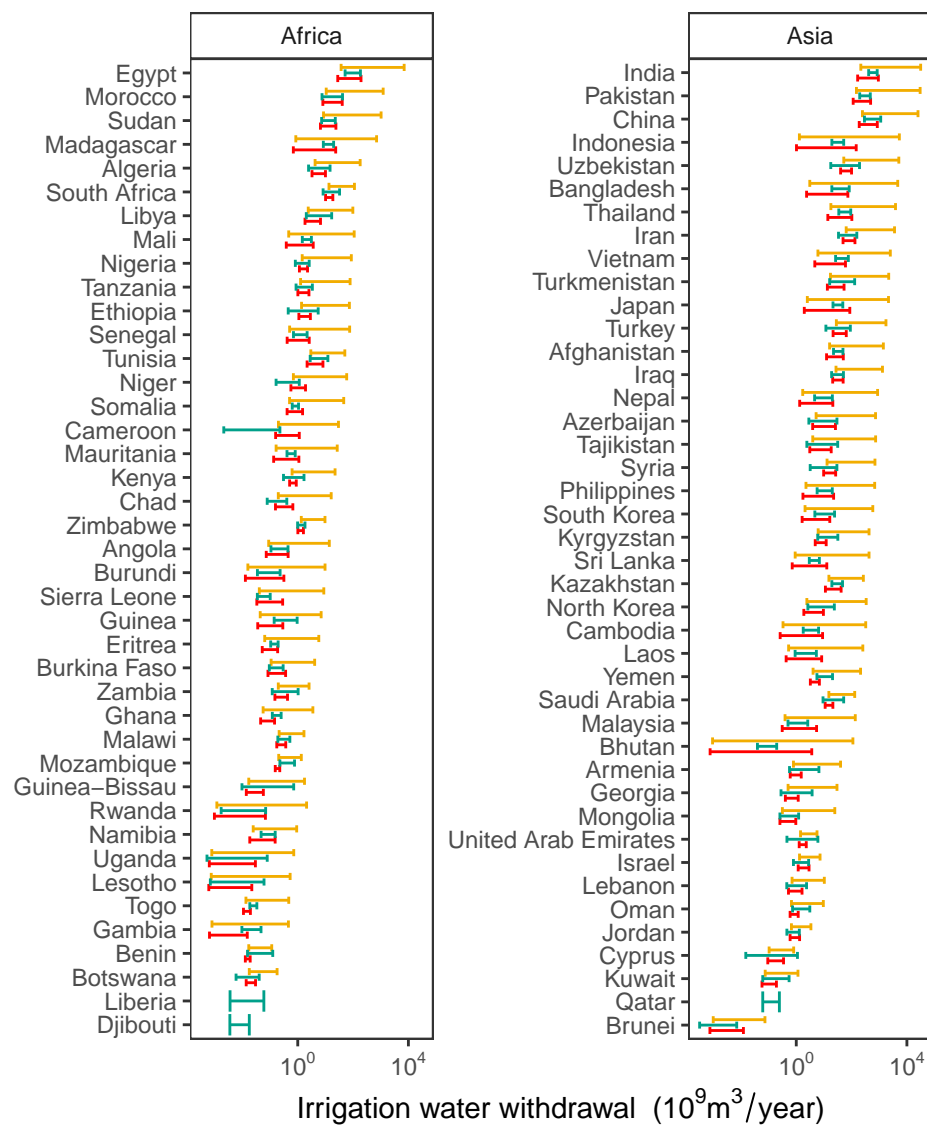
```
dd
```

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

— GM

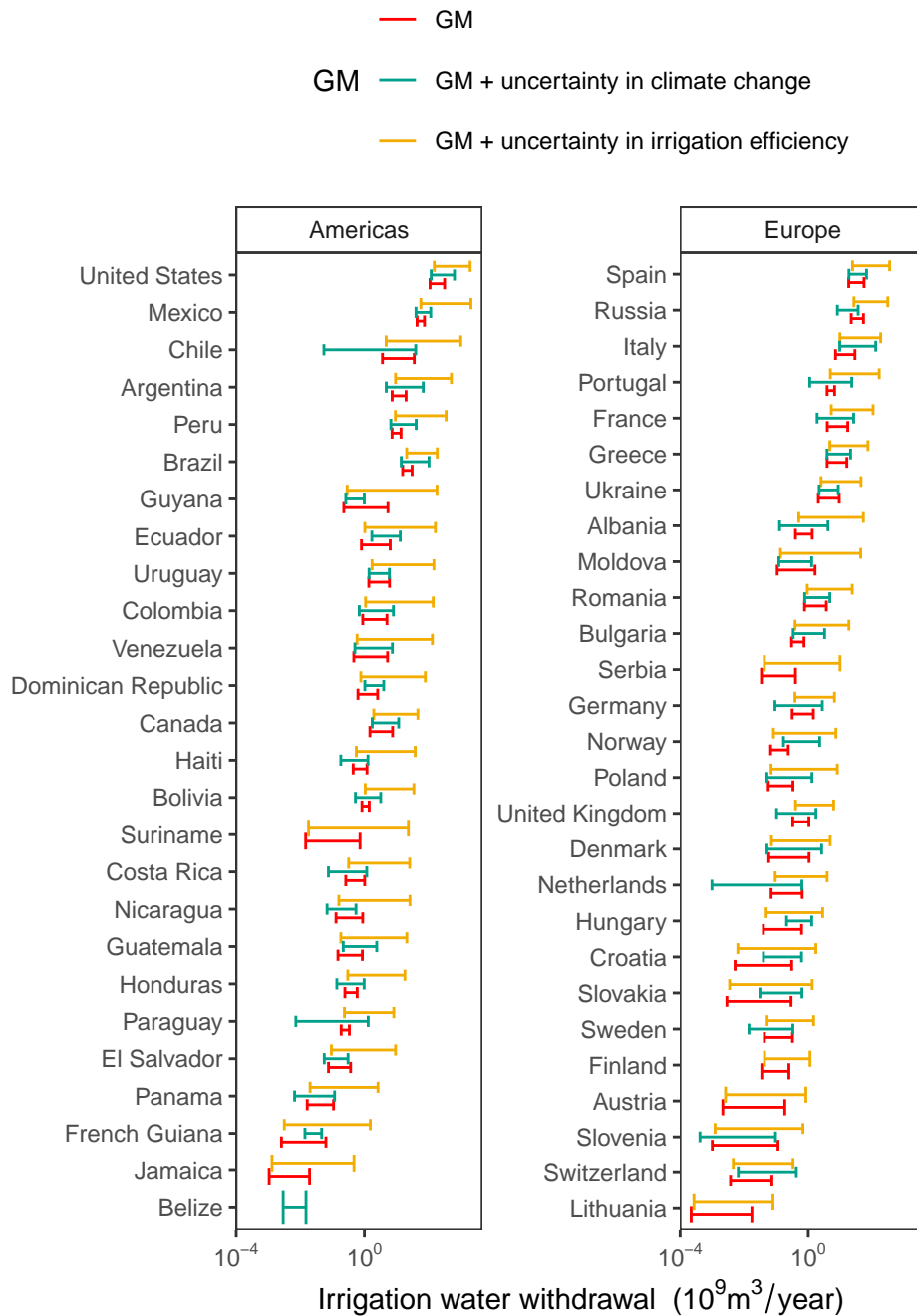
GM — GM + uncertainty in climate change

— GM + uncertainty in irrigation efficiency



```
##
```

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



```
# COMPARE RANGES -----
```

```
all.uncertainties <- all.uncertainties[, range:= max - min]
```

```
dd <- melt(all.uncertainties, measure.vars = c("min", "max")) %>%
  dcast(., Country + Continent + variable ~ Approach, value.var = "value") %>%
  .[variable == "max"] %>%
  na.omit() %>%
  # Calculate differences in orders of magnitude
```

```

[, order.magnitude:= log10(`GM + uncertainty in irrigation efficiency`) -
  log10(`GM + uncertainty in climate change`)] %>%
.[order(-order.magnitude)]

```

```

ggplot(dd, aes(order.magnitude)) +
  geom_histogram() +
  facet_wrap(~Continent) +
  labs(x = "Order of magnitude", y = "N° of countries") +
  theme_AP()

```

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

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

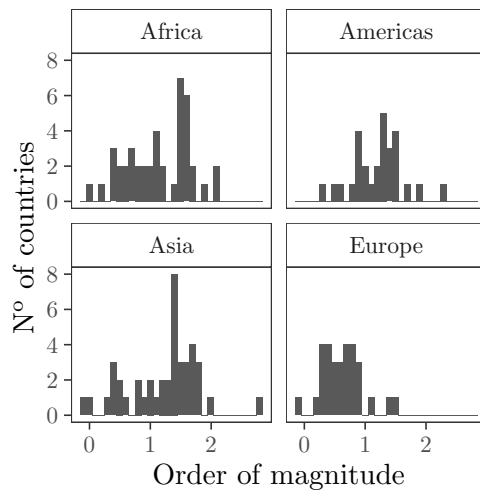
```

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.

```

```
## Warning in (function (texString, cex = 1, face = 1, engine =
```

```
##getOption("tikzDefaultEngine"), : Attempting to calculate the width of a Unicode
## string using the pdftex engine. This may fail! See the Unicode section of ?
## tikzDevice for more information.
```



5 Sensitivity analysis

```
# SAMPLE MATRIX DISTRIBUTIONS -----

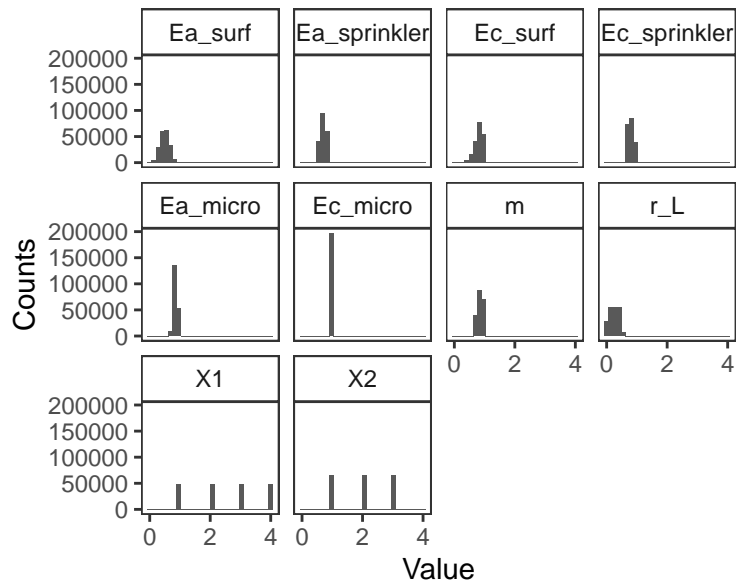
# Define labels
label_facets <- c("Ea_surf" = "$E_{a_{su}}$",
                  "Ec_surf" = "$E_{c_{su}}$",
                  "Ea_sprinkler" = "$E_{a_{sp}}$",
                  "Ec_sprinkler" = "$E_{c_{sp}}$",
                  "Ea_micro" = "$E_{a_{mi}}$",
                  "Ec_micro" = "$E_{c_{mi}}$",
                  "Proportion_large" = "$f_L$",
                  "m" = "$m$",
                  "r_L" = "$r_L$")

mat <- data.table(full_sample_matrix(IFT = "Jager", Country = "Spain")$matrix)
mat <- mat[, Proportion_large := NULL]

## Warning in `[.data.table`(mat, , `:=`(Proportion_large, NULL)): Column
## 'Proportion_large' does not exist to remove

melt(mat, measure.vars = colnames(mat)) %>%
  ggplot(., aes(value)) +
  geom_histogram() +
  labs(x = "Value", y = "Counts") +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  facet_wrap(~variable) +
  theme_AP()

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



EXTRACT SOBOLO' INDICES

```
ind <- lapply(y$`Rohwer et al. 2007`, function(x) x[["indices"]])$results)
names(ind) <- rohwer$Country
ind <- rbindlist(ind, idcol = "Country")

ind[, Continent:= countrycode(ind[, Country], origin = "country.name",
                              destination = "continent")]
```

```
## Warning in countrycode_convert(sourcevar = sourcevar, origin = origin, destination = dest,  
tmp.ift <- split(rohwer, rohwer$IFT)
```

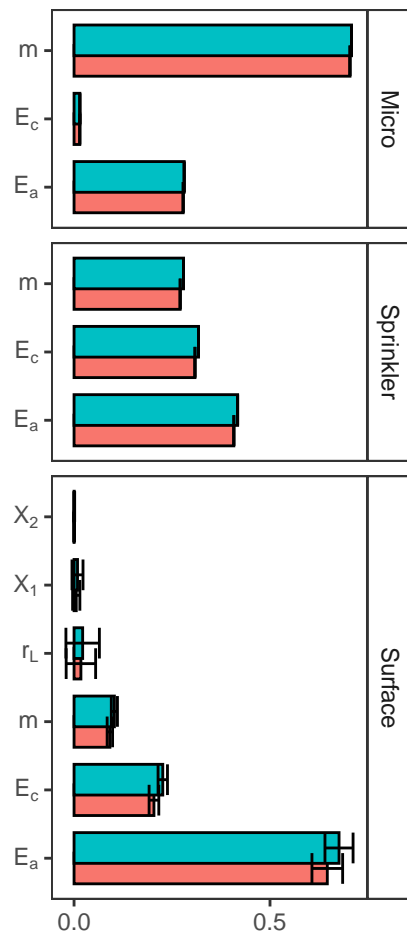
```
out <- list()
for(i in names(tmp.ift)) {
  out[[i]] <- ind[Country %in% tmp.ift[[i]][, Country]]
}
```

```
# PLOT SOBOLO' INDICES
```

```
dt.indices <- rbindlist(out, idcol = "IFT") %>%
  .[!IFT == "Mixed"] %>%
  .[, .(mean = mean(original), sd = sd(original)), .(sensitivity, parameters, IFT)] %>%
  .[, parameters:= ifelse(parameters == "Ea_surf", "E[a]",
    ifelse(parameters == "Ec_surf", "E[c]",
      ifelse(parameters == "Ea_sprinkler", "E[a]",
        ifelse(parameters == "Ec_sprinkler", "E[c]",
          ifelse(parameters == "Ea_micro", "E[a]",
            ifelse(parameters == "Ec_micro", "E[c]",
              ifelse(parameters == "X1", "X[1]",
                ifelse(parameters == "X2",
                  ifelse(parameters ==
```

```
rohwer.indices <- ggplot(dt.indices, aes(parameters, mean, fill = sensitivity), color = black)
  geom_bar(stat = "identity", position = position_dodge(0.6), color = "black") +
  geom_errorbar(aes(ymin = mean - sd, ymax = mean + sd), position = position_dodge(0.6)) +
  scale_x_discrete(labels = parse_format()) +
  scale_y_continuous(breaks = pretty_breaks(n = 2)) +
  scale_fill_discrete(name = "Sensitivity", labels = expression(S[i], T[i])) +
  labs(x = "", y = "") +
  coord_flip() +
  facet_grid(IFT~., space = "free_y", scale = "free_y") +
  theme_AP() +
  theme(legend.position = "none")
```

rohwer.indices



EXTRACT SOBEL' INDICES FOR JAGER -----

```
jager.tmp <- lapply(y[["Jägermeyr et al. 2015"]], function(x) x$indices$results)
names(jager.tmp) <- new.rohwer$Country
```

```
jager.ind <- rbindlist(jager.tmp, idcol = "Country") %>%
  .[, Continent:= countrycode(., Country),
```

```

        origin = "country.name",
        destination = "continent")] %>%
.[, parameters:= ifelse(parameters == "Ea_surf", "E[a[s]]",
        ifelse(parameters == "Ec_surf", "E[c[s]]",
        ifelse(parameters == "Ea_sprinkler", "E[a[p]]",
        ifelse(parameters == "Ec_sprinkler", "E[c[p]]",
        ifelse(parameters == "Ea_micro", "E[a[m]]",
        ifelse(parameters == "Ec_micro", "E[c[m]]",
        ifelse(parameters == "r_L", "r[L]",
        ifelse(parameters == "X1",
        ifelse(parameters ==

## Warning in countrycode_convert(sourcevar = sourcevar, origin = origin, destination = dest,
Continent_vector <- c("Africa", "Americas", "Asia", "Europe")

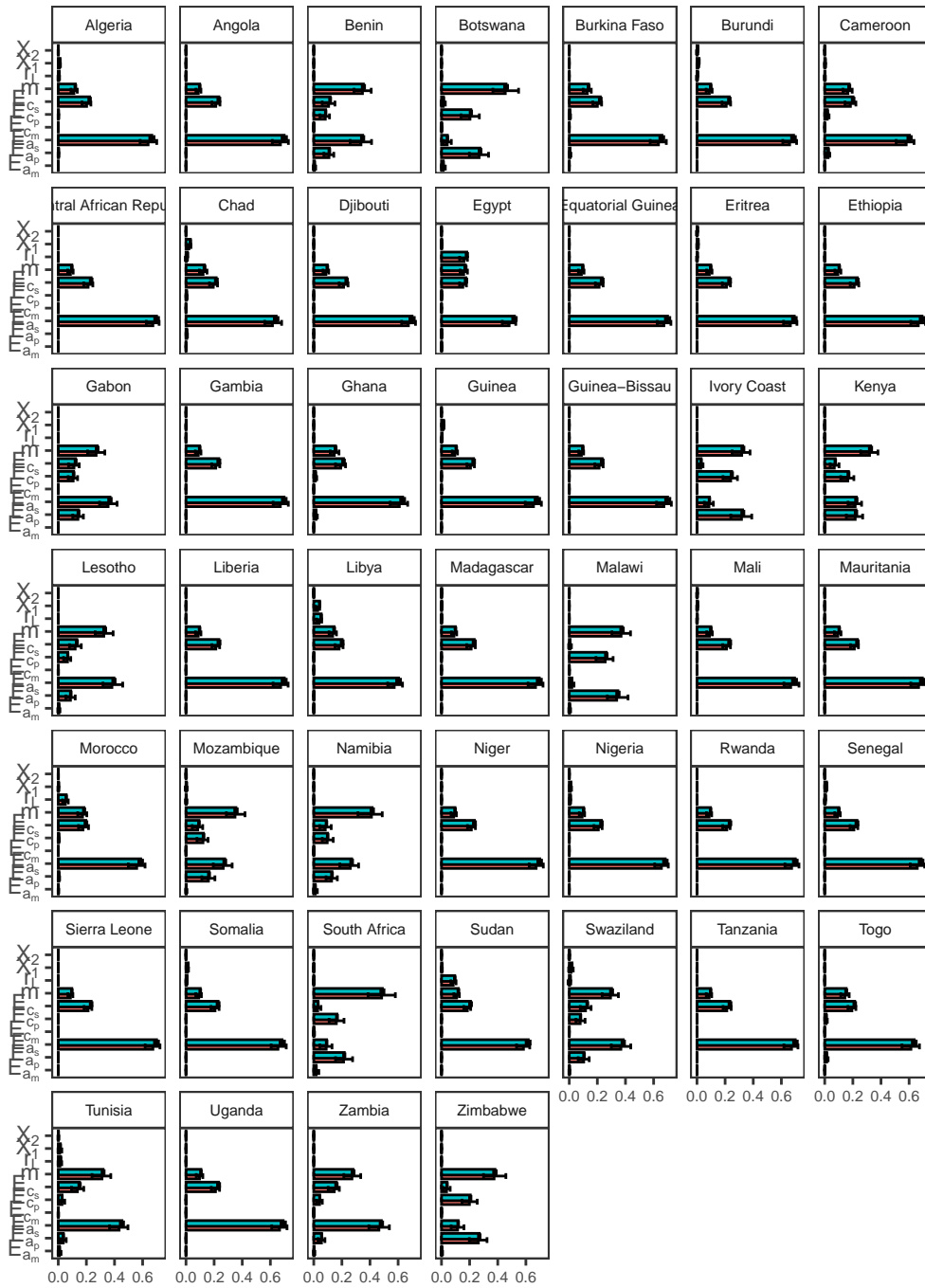
lapply(Continent_vector, function(x)
  ggplot(jager.ind[Continent == x], aes(parameters, original, fill = sensitivity), color = bla
    geom_bar(stat = "identity", position = position_dodge(0.6), color = "black") +
    geom_errorbar(aes(ymin = low.ci, ymax = high.ci),
      position = position_dodge(0.6)) +
    scale_fill_discrete(name = "Sensitivity", labels = c("Si", "Ti")) +
    labs(x = "", y = "Sobol' indices") +
    scale_x_discrete(labels = ggplot2::parse_safe) +
    coord_flip() +
    scale_y_continuous(breaks = pretty_breaks(n = 3)) +
    facet_wrap(~Country) +
    theme_AP() +
    theme(strip.text.x = element_text(size = 6),
      axis.text.x = element_text(size = 6)) +
    ggtitle(x)
  )

## [[1]]

```


Africa

Sensitivity ■ Si ■ Ti

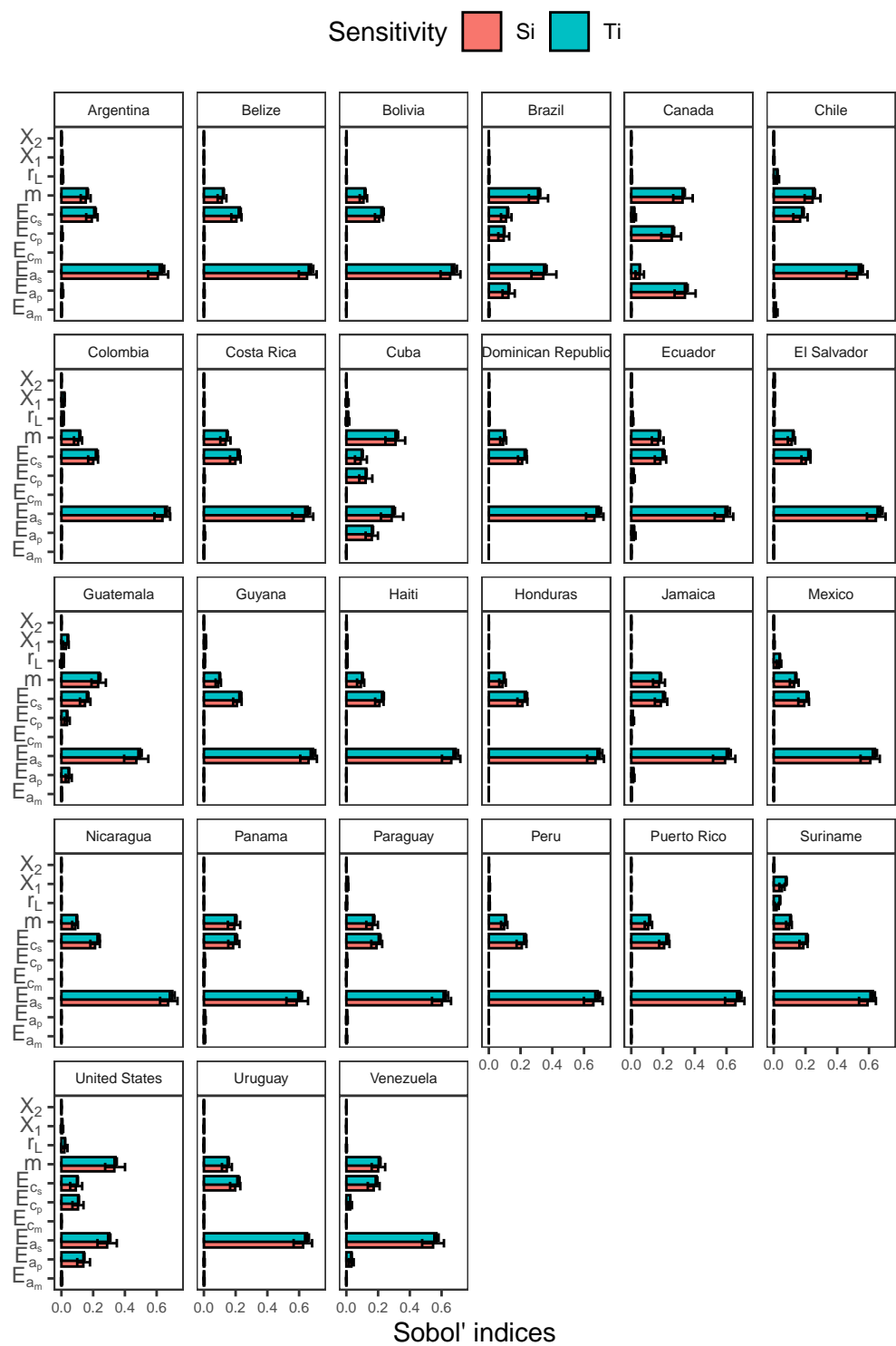


Sobol' indices

##

[[2]]

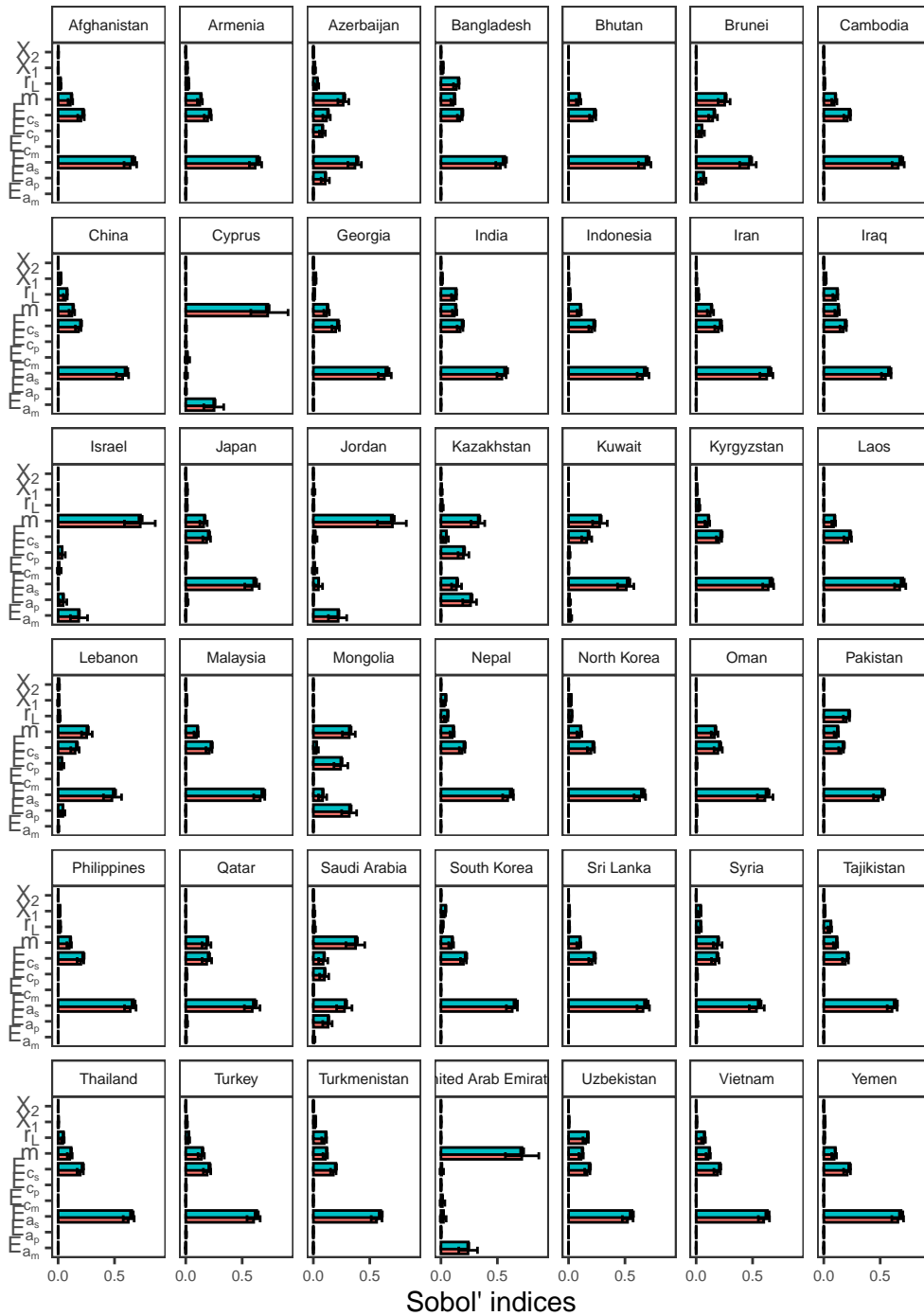
Americas



[[3]]

Asia

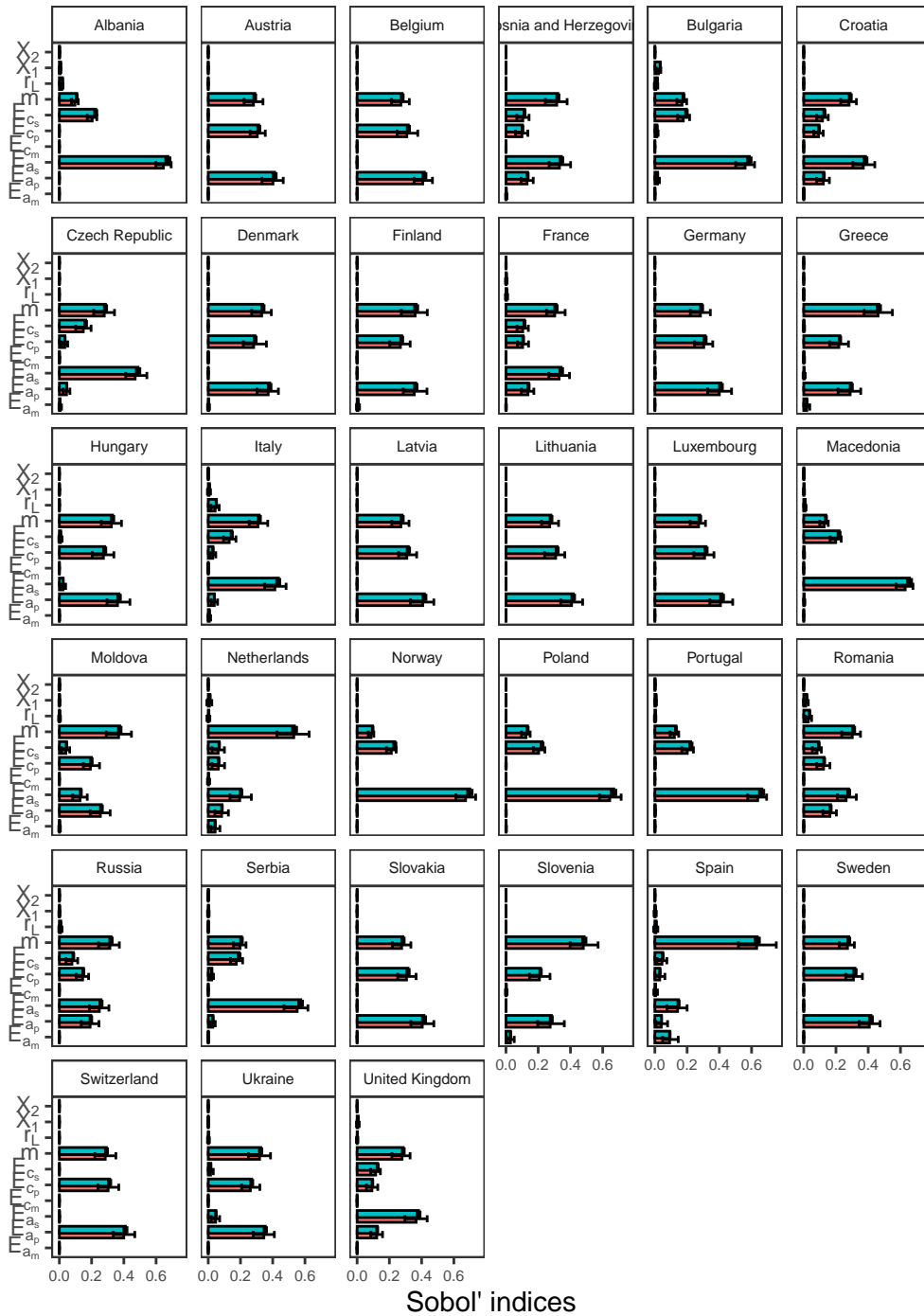
Sensitivity ■ Si ■ Ti



[[4]]

Europe

Sensitivity ■ Si ■ Ti



JAGER INDICES MERGED

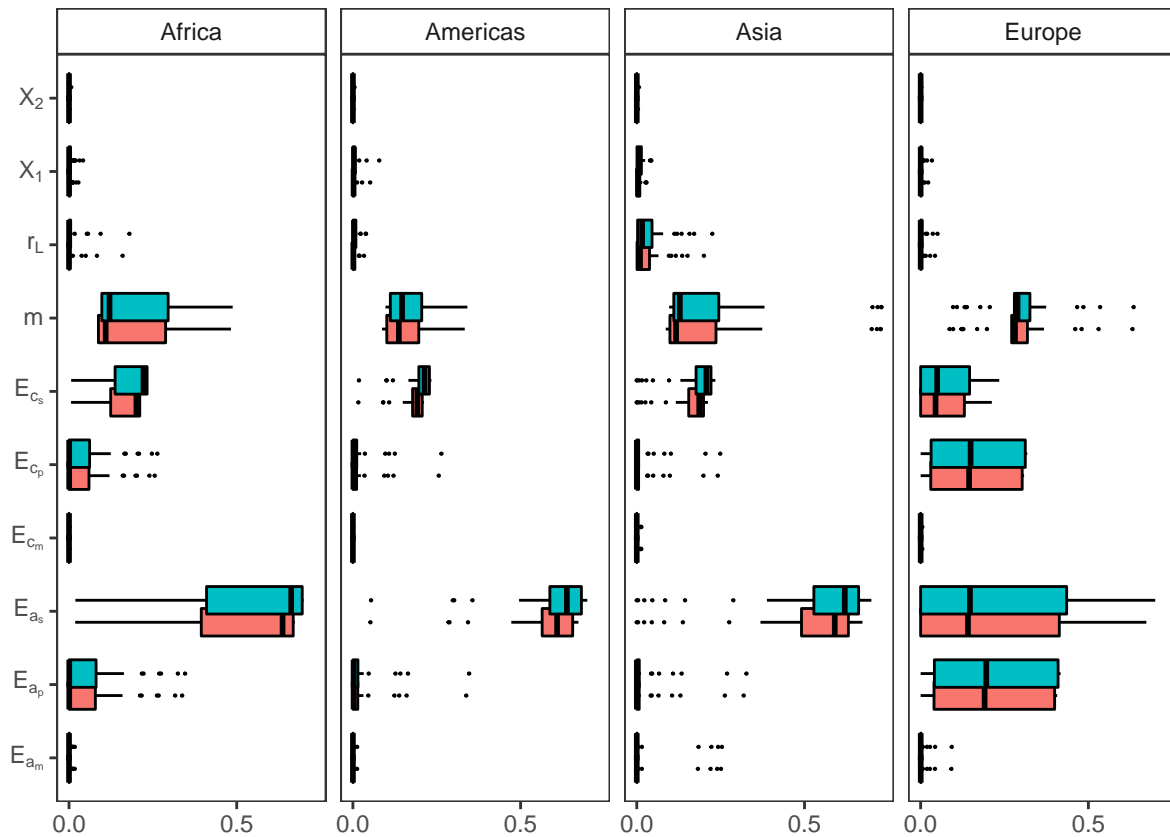
```
jager.indices <- jager.ind %>%
  na.omit() %>%
```

```

[!Continent == "Oceania"] %>%
ggplot(., aes(parameters, original, fill = sensitivity), color = black) +
geom_boxplot(position = position_dodge(0.6), color = "black",
outlier.size = 0.1) +
scale_fill_discrete(name = "Sensitivity", labels = expression(S[i], T[i])) +
scale_x_discrete(labels = parse_format()) +
scale_y_continuous(breaks = pretty_breaks(n = 2)) +
labs(x = "", y = "") +
facet_grid(~Continent) +
coord_flip() +
theme_AP() +
theme(legend.position = "none")

```

jager.indices



MERGE INDICES -----

```

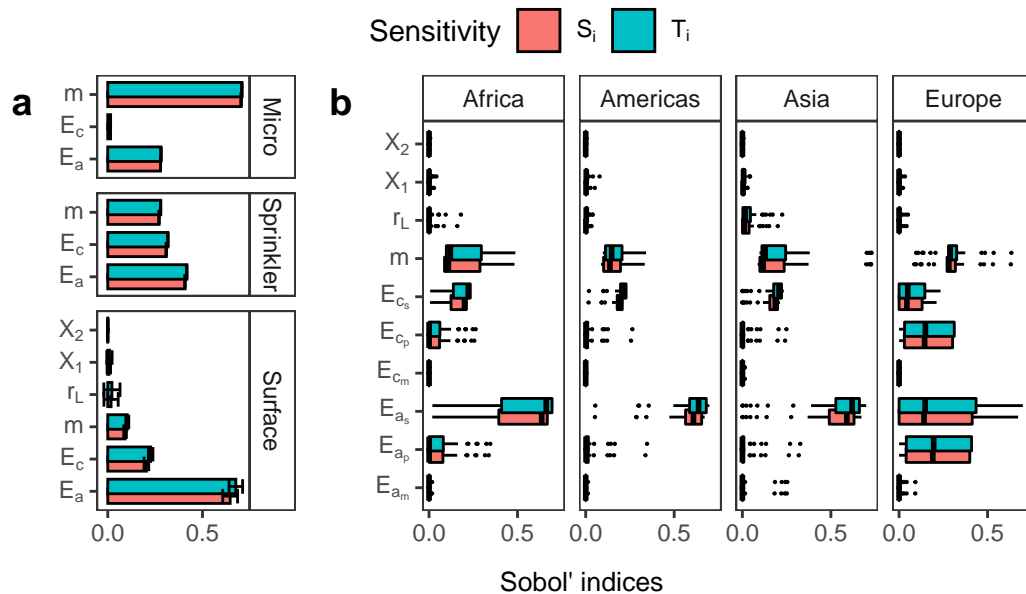
bottom <- plot_grid(rohwer.indices, jager.indices, ncol = 2,
rel_widths = c(0.3, 0.7), labels = "auto")

legend <- get_legend(rohwer.indices + theme(legend.position = "top"))

final <- plot_grid(legend, bottom, ncol = 1, rel_heights = c(0.15, 0.85))
ggdraw(add_sub(final, "Sobol' indices", vpadding = grid::unit(0,"lines"),

```

```
y = 6, x = 0.55, vjust = 5.5, size = 10))
```



SESSION INFORMATION -----

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## 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] parallel stats graphics grDevices utils datasets methods
## [8] base
##
## other attached packages:
## [1] checkpoint_1.0.0 benchmarkme_1.0.7 ncd4_1.17 rworldmap_1.3-6
## [5] sp_1.4-5 overlapping_1.6 testthat_3.0.4 scales_1.1.1
## [9] gggridges_0.5.3 countrycode_1.3.0 doParallel_1.0.16 iterators_1.0.13
## [13] foreach_1.5.1 cowplot_1.1.1 wesanderson_0.3.6 sensobol_1.0.3
## [17] forcats_0.5.1 stringr_1.4.0 dplyr_1.0.7 purrr_0.3.4
## [21] readr_2.0.1 tidyr_1.1.3 tibble_3.1.3 ggplot2_3.3.5
## [25] tidyverse_1.3.1 data.table_1.14.0
##
## loaded via a namespace (and not attached):
```

```
## [1] fs_1.5.0          lubridate_1.7.10    httr_1.4.2
## [4] tools_4.0.3        backports_1.2.1     utf8_1.2.2
## [7] R6_2.5.0           DBI_1.1.1           colorspace_2.0-2
## [10] withr_2.4.2        tidyselect_1.1.1    gridExtra_2.3
## [13] compiler_4.0.3     cli_3.0.1           rvest_1.0.1
## [16] xml2_1.3.2         labeling_0.4.2      digest_0.6.27
## [19] foreign_0.8-81     rmarkdown_2.10      benchmarkmeData_1.0.4
## [22] pkgconfig_2.0.3    htmltools_0.5.1.1   highr_0.9
## [25] dbplyr_2.1.1       maps_3.3.0          rlang_0.4.11
## [28] readxl_1.3.1       rstudioapi_0.13     farver_2.1.0
## [31] generics_0.1.0     tikzDevice_0.12.3.1 jsonlite_1.7.2
## [34] magrittr_2.0.1     dotCall64_1.0-1     Matrix_1.3-4
## [37] Rcpp_1.0.7         munsell_0.5.0       fansi_0.5.0
## [40] viridis_0.6.1      lifecycle_1.0.0     stringi_1.7.3
## [43] yaml_2.2.1         plyr_1.8.6          grid_4.0.3
## [46] maptools_1.1-1     crayon_1.4.1        lattice_0.20-44
## [49] haven_2.4.3        hms_1.1.0           knitr_1.33
## [52] pillar_1.6.2       codetools_0.2-18    reprex_2.0.1
## [55] glue_1.4.2         evaluate_0.14       modelr_0.1.8
## [58] vctrs_0.3.8        spam_2.7-0          tzdb_0.1.2
## [61] Rdpack_2.1.2       cellranger_1.1.0    gtable_0.3.0
## [64] assertthat_0.2.1   xfun_0.25           rbibutils_2.2.3
## [67] broom_0.7.9        filehash_2.4-2      viridisLite_0.4.0
## [70] fields_12.5        ellipsis_0.3.2
```

```
## 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 = FALSE))
```

```
## Num threads:
```

```
## [1] 8
```