Uncertainty in global irrigation water use persists after 50 years of research

R code

Arnald Puy

Contents

1 Preliminary functions

```
#   PRELIMINARY FUNCTIONS ###########################################################

sensobol::load_packages(c("openxlsx", "data.table", "tidyverse","cowplot",
                          "benchmarkme", "parallel", "wesanderson", "scales", "ncdf4",
                          "countrycode", "rworldmap", "sp", "doParallel", "here", "lme4",
                          "microbenchmark", "mgcv", "brms", "randomForest", "here",
                          "igraph", "ggraph"))

# 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),
          strip.background = element_rect(fill = "white"),
          legend.text = element_text(size = 7.3),
          axis.title = element_text(size = 10),
          legend.key.width = unit(0.4, "cm"),
          legend.key.height = unit(0.4, "cm"),
          legend.key.spacing.y = unit(0, "lines"),
          legend.box.spacing = unit(0, "pt"),
          legend.title = element_text(size = 7.3),
          axis.text.x = element_text(size = 7),
          axis.text.y = element_text(size = 7),
          axis.title.x = element_text(size = 7.3),
          axis.title.y = element_text(size = 7.3),
          plot.title = element_text(size = 8),
          strip.text.x = element_text(size = 7.4),
          strip.text.y = element_text(size = 7.4))
}

# Select color palette -----------------------------------------------------

selected.palette <- "Darjeeling1"
```

```
# SOURCE ALL R FUNCTIONS NEEDED FOR THE STUDY ######################################

# Source all .R files in the "functions" folder ------------------------------

r_functions <- list.files(path = here("functions"), pattern = "\\.R$", full.names = TRUE)
lapply(r_functions, source)
```

2 Bibliographical study

```r
# NAOMI DATASET ###################################################################

references.projected <- data.table(read.xlsx("./data/references_projection.xlsx")) %>%
  .[, focus:= "projected"]

references.current <- data.table(read.xlsx("./data/references_current.xlsx")) %>%
  .[, focus:= "current"]

references.full.dt <- rbind(references.projected, references.current) %>%
  .[, study:= paste(author, model, climate.scenario, sep = ".")]

# CLEAN THE DATASET ###############################################################

colnames_vector <- c("title", "author", "region")

# Remove leading and trailing spaces ------------------------------------------

references.full.dt[, (colnames_vector):= lapply(.SD, trimws), .SDcols = (colnames_vector)]
references.full.dt[, (colnames_vector):= lapply(.SD, str_squish), .SDcols = (colnames_vector)]

# Lowercaps -------------------------------------------------------------------

references.full.dt[, (colnames_vector):= lapply(.SD, tolower), .SDcols = (colnames_vector)]

# Remove multiple spaces ------------------------------------------------------

references.full.dt[, (colnames_vector):= lapply(.SD, function(x)
  gsub("\\s+", " ", x)), .SDcols = (colnames_vector)]

# Correct America -------------------------------------------------------------

references.full.dt[, region:= ifelse(region == "america", "americas", region)]

# Extract the publication year ------------------------------------------------

references.full.dt[, publication.date:= str_extract(author, "\\d{4}")] %>%
  .[, publication.date:= as.numeric(publication.date)]
```

```r
# FEATURES OF THE DATASET #########################################################

# Definition of target years --------------------------------------------------

target_year <- c(2000, 2010, 2050, 2070, 2100)

# Name of different studies ---------------------------------------------------
```

```r
sort(unique(references.full.dt[variable == "iww" & region == "global", title]))
```

```
##  [1] "a global water scarcity assessment under shared socio-economic pathways - part 2: wate
##  [2] "a pathway of global food supply adaptation in a world with increasingly constrained gr
##  [3] "a reservoir operation scheme for global river routing models"
##  [4] "agricultural green and blue water consumption and its influence on the global water sy
##  [5] "an integrated assessment of global and regional water demands for electricity generati
##  [6] "an integrated model for the assessment of global water resources - part 2: applicatior
##  [7] "appraisal and assessment of world water resources"
##  [8] "aquastat: fao's global information system on water and agriculture"
##  [9] "bending the curve: toward global sustainability"
## [10] "cited in world resources 1990-1991, p. 172"
## [11] "climate change impacts on irrigation water requirements: effects of mitigation, 1990-2
## [12] "climate impacts on global irrigation requirements under 19 gcms, simulated with a vege
## [13] "climate mitigation policy implications for global irrigation water demand"
## [14] "climate policy implications for agricultural water demand"
## [15] "future long-term changes in global water resources driven by socio-economic and climat
## [16] "global and regional evaluation of energy for water"
## [17] "global hydrological cycles and world water resources,"
## [18] "global impacts of conversions from natural to agricultural ecosystems on water resourc
## [19] "global irrigation characteristics and effects simulated by fully coupled land surface
## [20] "global irrigation water demand: variability and uncertainties arising from agricultura
## [21] "global modeling of irrigation water requirements"
## [22] "global modeling of withdrawal, allocation and consumptive use of surface water and gro
## [23] "global monthly sectoral water use for 2010-2100 at 0.5° resolution across alternative
## [24] "global water demand and supply projections"
## [25] "globwat - a global water balance model to assess water use in irrigated agriculture"
## [26] "high-resolution modeling of human and climate impacts on global water resources"
## [27] "how can we cope with the water resources situation by the year 2050?"
## [28] "human appropriation of renewable fresh water"
## [29] "impact of climate forcing uncertainty and human water use on global and continental wa
## [30] "implementation and evaluation of irrigation techniques in the community land model"
## [31] "incorporating anthropogenic water regulation modules into a land surface model"
## [32] "incorporation of groundwater pumping in a global land surface model with the represent
## [33] "integrated crop water management might sustainably halve the global food gap"
## [34] "isimip database"
## [35] "long-term global water projections using six socioeconomic scenarios in an intgrated a
## [36] "lpjml4 - a dynamic global vegetation model with managed land - part 2: model evaluatio
## [37] "modelling global water stress of the recent past: on the relative importance of trends
## [38] "multimodel projections and uncertainties of irrigation water demand under climate char
## [39] "pcr-globwb 2: a 5 arcmin global hydrological and water resources model"
## [40] "physical impacts of climate change on water resources"
## [41] "present-day irrigation mitigares heat extremes"
## [42] "projecting irrigation water requirements across multiple socio-economic development fu
## [43] "projection of future world water resources under sres scenarios: water withdrawal"
## [44] "quantifying global agricultural water appropriation with data derived from earth obser
## [45] "recent global cropland water consumption constrained by observations"
```

```
## [46] "reconciling irrigated food production with environmental flows for sustainable develop
## [47] "reconstructing 20th century global hydrography: a contribution to the global terrestri
## [48] "the state of the world's land and water resources for food and agriculture"
## [49] "the world's water, 2000-2001: the biennial report on freshwater resources"
## [50] "united nations world water development report 2020: water and climate change"
## [51] "water 2050. moving toward a sustainable vision fot the earth's fresh water"
## [52] "water and sustainability. global pattern and long-range problems"
## [53] "water savings potentials of irrigation systems: global simulation of processes and lin
## [54] "water sector assumptions for the shared socioeconomic pathways in an integrated modeli
## [55] "world agriculture towards 2030/2050: the 2012 revision"
## [56] "world agriculture towards 2030/2055"
## [57] "world water demand and supply, 1990 to 2025: scenarios and issues"
## [58] "world water in 2025 - global modeling and scenario analysis for the world commission o
## [59] "world water resources and their future"
```

```
# Number of data points ------------------------------------------------------

nrow(references.full.dt[variable == "iww" & region == "global"])
```

```
## [1] 1394
```

```
# Number of different studies per variable -----------------------------------

references.full.dt[region == "global", unique(title), variable] %>%
  .[, .N, variable]
```

```
##    variable     N
##      <char> <int>
## 1:      iww    60
## 2:      tww    20
## 3:      iwc    20
## 4:      twc     4
## 5:      iwr     2
```

```
# Number of data points for each target year ---------------------------------

references.full.dt[variable == "iww" & region == "global" &
                     estimation.year %in% target_year, .N, estimation.year]
```

```
##    estimation.year     N
##              <num> <int>
## 1:            2000    66
## 2:            2070   124
## 3:            2100   121
## 4:            2010   110
## 5:            2050   125
```

```
# Number of unique studies estimating for each target year -------------------

references.full.dt[variable == "iww" & region == "global" &
                     estimation.year %in% target_year, unique(title), estimation.year] %>%
```

```
  .[, .N, estimation.year]
```

```
##    estimation.year     N
##               <num> <int>
## 1:            2000    22
## 2:            2070     5
## 3:            2100     5
## 4:            2010    10
## 5:            2050    13
```

```
# Number of data points for every targeted year -----------------------------

references.full.dt[variable == "iww" & region == "global", .N, estimation.year] %>%
  .[order(estimation.year)]
```

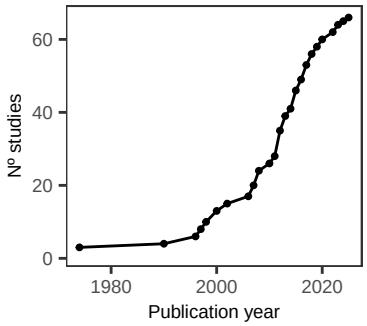
```
##     estimation.year     N
##               <num> <int>
##  1:            1900     3
##  2:            1910     2
##  3:            1920     2
##  4:            1930     2
##  5:            1940     4
##  6:            1950     4
##  7:            1960     6
##  8:            1970     5
##  9:            1975    22
## 10:            1980    29
## 11:            1983     1
## 12:            1985    33
## 13:            1988     1
## 14:            1990    28
## 15:            1993     2
## 16:            1994     3
## 17:            1995    40
## 18:            1996     2
## 19:            2000    66
## 20:            2002     1
## 21:            2003     1
## 22:            2004     1
## 23:            2005    34
## 24:            2006     1
## 25:            2007     1
## 26:            2008     1
## 27:            2010   110
## 28:            2015     9
## 29:            2020    96
## 30:            2021     1
## 31:            2025    14
```

```
## 32:              2030    87
## 33:              2035     7
## 34:              2040    98
## 35:              2050   125
## 36:              2055     6
## 37:              2060    87
## 38:              2065     7
## 39:              2070   124
## 40:              2075     6
## 41:              2080   103
## 42:              2090    84
## 43:              2095    14
## 44:              2100   121
##     estimation.year     N
```

```
# Cumulative sum of published studies ----------------------------------------

cumulative.iww <- references.full.dt[, .(title, publication.date, variable)] %>%
  .[variable == "iww"] %>%
  .[!duplicated(.)] %>%
  setorder(., publication.date) %>%
  .[, .N, publication.date] %>%
  .[, cumulative_sum := cumsum(N)] %>%
  ggplot(., aes(publication.date, cumulative_sum)) +
  geom_line() +
  scale_x_continuous(breaks = breaks_pretty(n = 3)) +
  geom_point(size = 0.7) +
  theme_AP() +
  labs(x = "Publication year", y = "Nº studies")

cumulative.iww
```

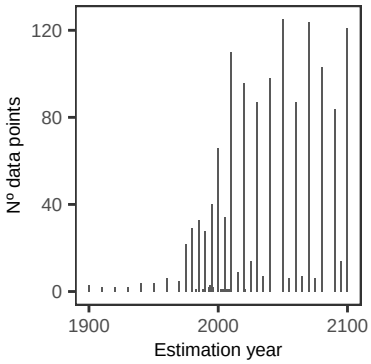
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

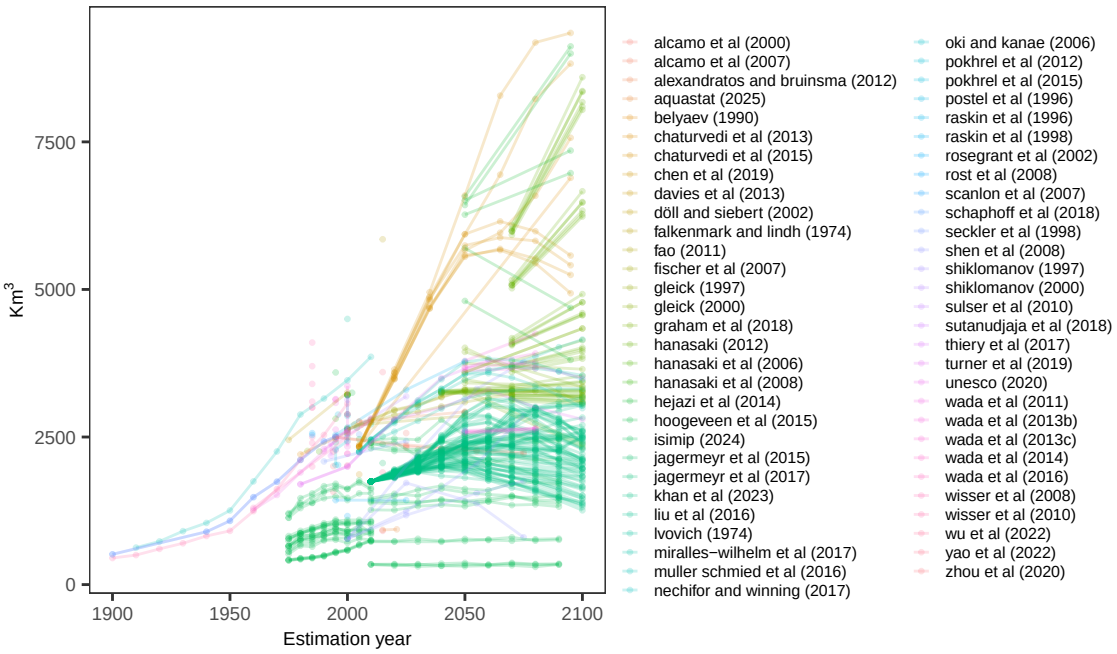



```
# DISTRIBUTION OF DATA POINTS THROUGH YEARS @#####################################
```

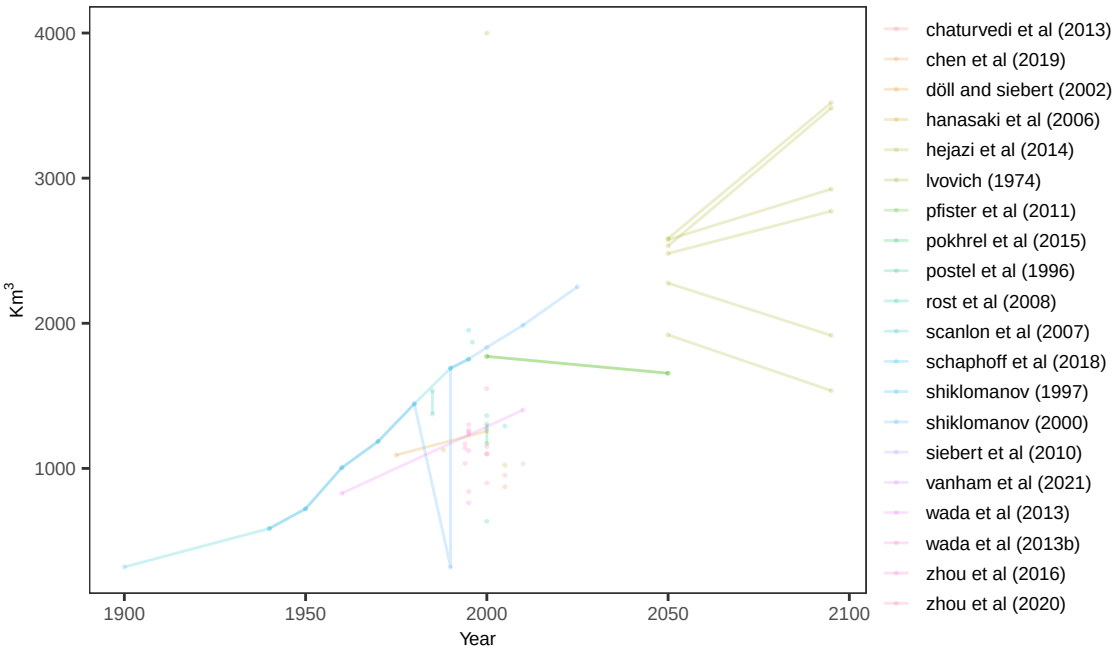
```
plot.bar <- references.full.dt[variable == "iww" & region == "global", .N, estimation.year] %>%
  ggplot(., aes(estimation.year, N)) +
  geom_bar(stat = "identity") +
  scale_x_continuous(breaks = breaks_pretty(n = 3)) +
  labs(x = "Estimation year", y = "Nº data points") +
  theme_AP()

plot.bar
```



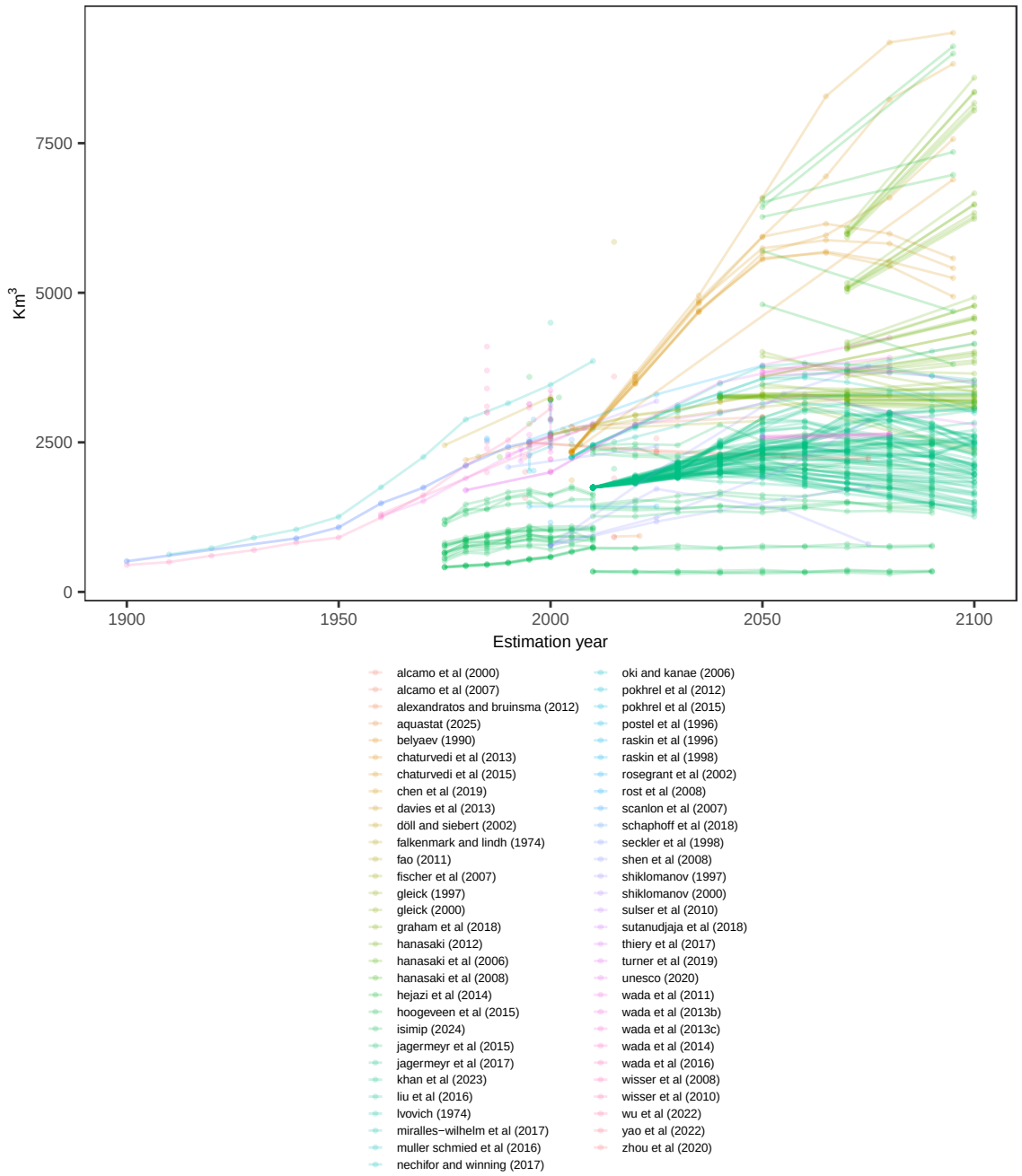

```
# PLOT ALL ESTIMATIONS ##########################################################

def.alpha <- 0.2

plot.iww <- references.full.dt[variable == "iww" & region == "global"] %>%
  .[, .(author, study, estimation.year, value)] %>%
  na.omit() %>%
  ggplot(., aes(estimation.year, value, color = author, group = study)) +
  geom_point(alpha = def.alpha, size = 0.5) +
  labs(x = "Estimation year", y = bquote("Km"^3)) +
  scale_color_discrete(name = "") +
  geom_line(alpha = def.alpha) +
  theme_AP() +
  guides(color = guide_legend(ncol = 2)) +
  theme(legend.text = element_text(size = 5.5),
        legend.key.width = unit(0.25, "cm"),
          legend.key.height = unit(0.25, "cm"))

plot.iww
```




```
references.full.dt[variable == "iwc" & region == "global"] %>%
  .[, .(author, study, estimation.year, value)] %>%
  na.omit() %>%
  ggplot(., aes(estimation.year, value, color = author, group = study)) +
  geom_point(alpha = def.alpha, size = 0.2) +
  labs(x = "Year", y = bquote("Km"^3)) +
  scale_color_discrete(name = "") +
  geom_line(alpha = def.alpha) +
  theme_AP()
```

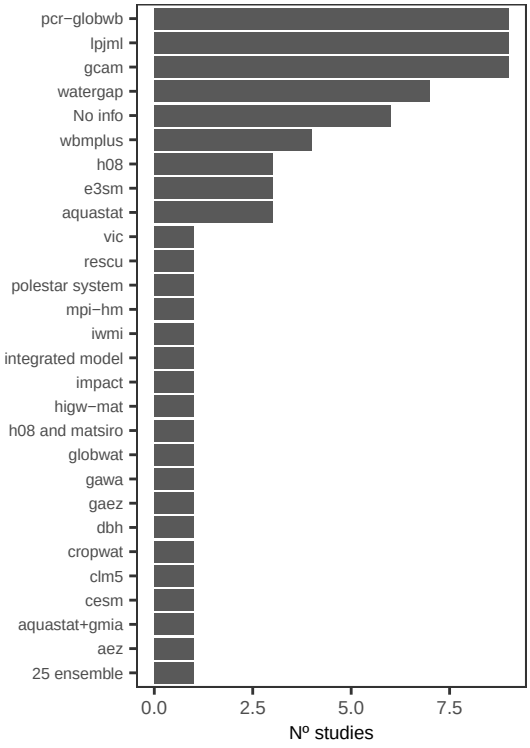


```
plot.iww +
  theme(legend.position = "bottom",
        legend.text = element_text(size = 4.8))
```



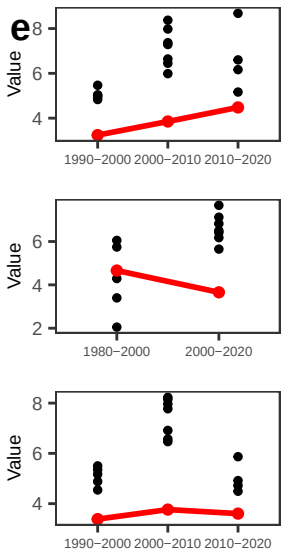

```
# PLOT NUMBER OF UNIQUE STUDIES PER MODEL ######################################

plot.models <- references.full.dt[variable == "iww" & region == "global"] %>%
  .[, .(title, doi, model)] %>%
  .[, model:= tolower(model)] %>%
  .[, unique(doi), model] %>%
  .[, model := gsub("(?i)watergap\\s*\\d*\\.?\\d*", "watergap", model, perl = TRUE)] %>%
```

```
  .[, .N, model]  %>%
  .[, model:= ifelse(is.na(model), "No info", model)] %>%
  ggplot(., aes(reorder(model, N), N)) +
  geom_bar(stat = "identity") +
  labs(x = "", y = "Nº studies") +
  coord_flip() +
  theme_AP() +
  theme(axis.text.y = element_text(size = 5.5))

plot.models
```



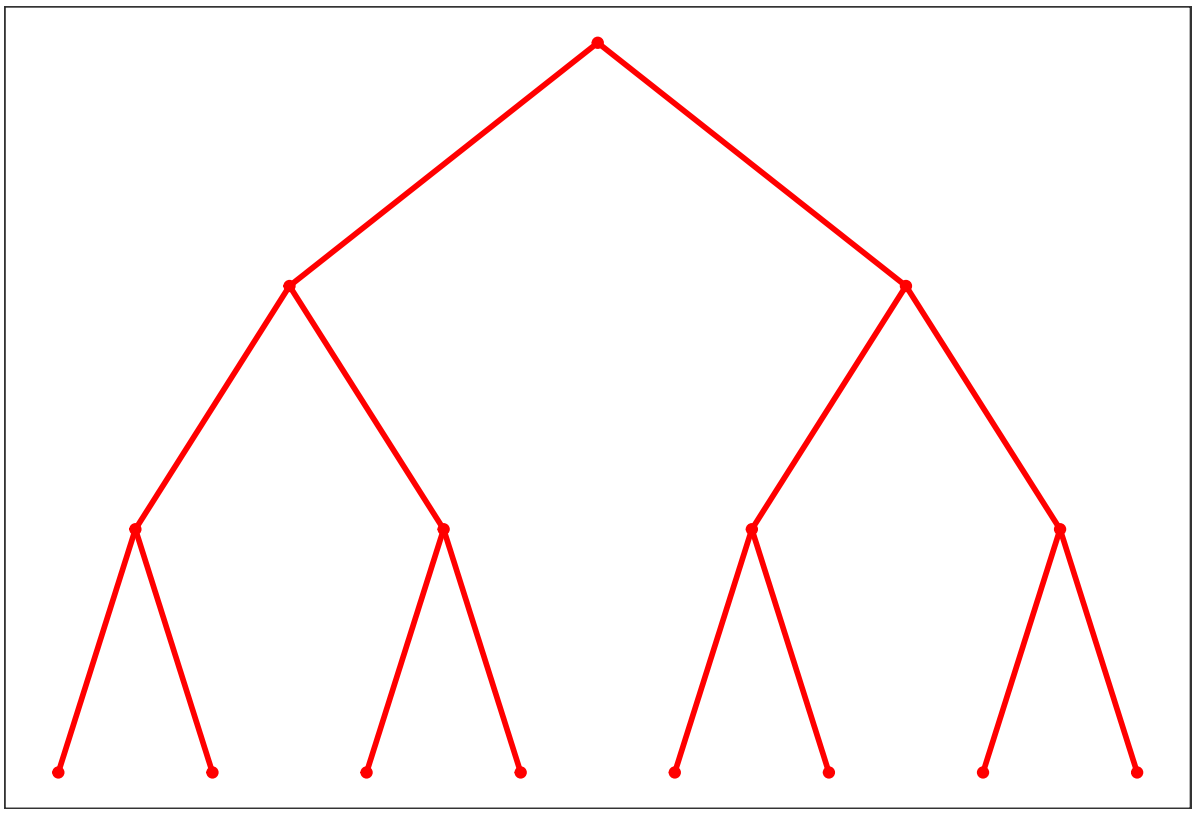

```
# PLOT EXAMPLES TO ILLUSTRATE APPROACH ##########################################

# Set seed for reproducibility ------------------------------------------------

set.seed(123)

# Create datasets for different SD trends -------------------------------------

data_increasing <- data.frame(
  period = rep(c("1990-2000", "2000-2010", "2010-2020"), times = c(5, 7, 4)),
  value = c(rnorm(5, mean = 5, sd = 0.3),  # Low SD
            rnorm(7, mean = 7, sd = 0.8),  # Medium SD
            rnorm(4, mean = 6, sd = 1.5))  # High SD
)
```

```r
data_decreasing <- data.frame(
  period = rep(c("1980-2000", "2000-2020"), times = c(5, 7)),
  value = c(rnorm(5, mean = 5, sd = 1.5),  # High SD
            rnorm(7, mean = 7, sd = 0.8))  # Medium
)

data_invertedV <- data.frame(
  period = rep(c("1990-2000", "2000-2010", "2010-2020"), times = c(5, 7, 4)),
  value = c(rnorm(5, mean = 5, sd = 0.4),  # Low SD
            rnorm(7, mean = 7, sd = 1.4),  # High SD (peak in the middle)
            rnorm(4, mean = 5, sd = 0.4))  # Low SD again
)

# Function to compute SD and create a ggplot ----------------------------------

create_plot <- function(data, title) {
  sd_values <- data %>%
    group_by(period) %>%
    summarize(sd_value = sd(value) + 3)

  ggplot(data, aes(x = period, y = value)) +
    geom_point(size = 1) +
    geom_point(data = sd_values, aes(x = period, y = sd_value), color = "red", size = 1.5) +
    geom_line(data = sd_values, aes(x = period, y = sd_value, group = 1), color = "red", linew
    theme_AP() +
    theme(axis.text.x = element_text(size = 5.35),
          plot.margin = unit(c(0.1, 0.1, 0, 0.1), "cm")) +
    scale_y_continuous(breaks = breaks_pretty(n = 3)) +
    labs(x = "", y = "Value")
}

# Generate the three plots ----------------------------------------------------

p1 <- create_plot(data_increasing)
p2 <- create_plot(data_decreasing)
p3 <- create_plot(data_invertedV)

# Merge using plot_grid -------------------------------------------------------

plot.examples.trends.data <- plot_grid(p1, p2, p3, ncol = 1, labels = c("e", "", ""))
plot.examples.trends.data
```



2.1 The garden of forking paths

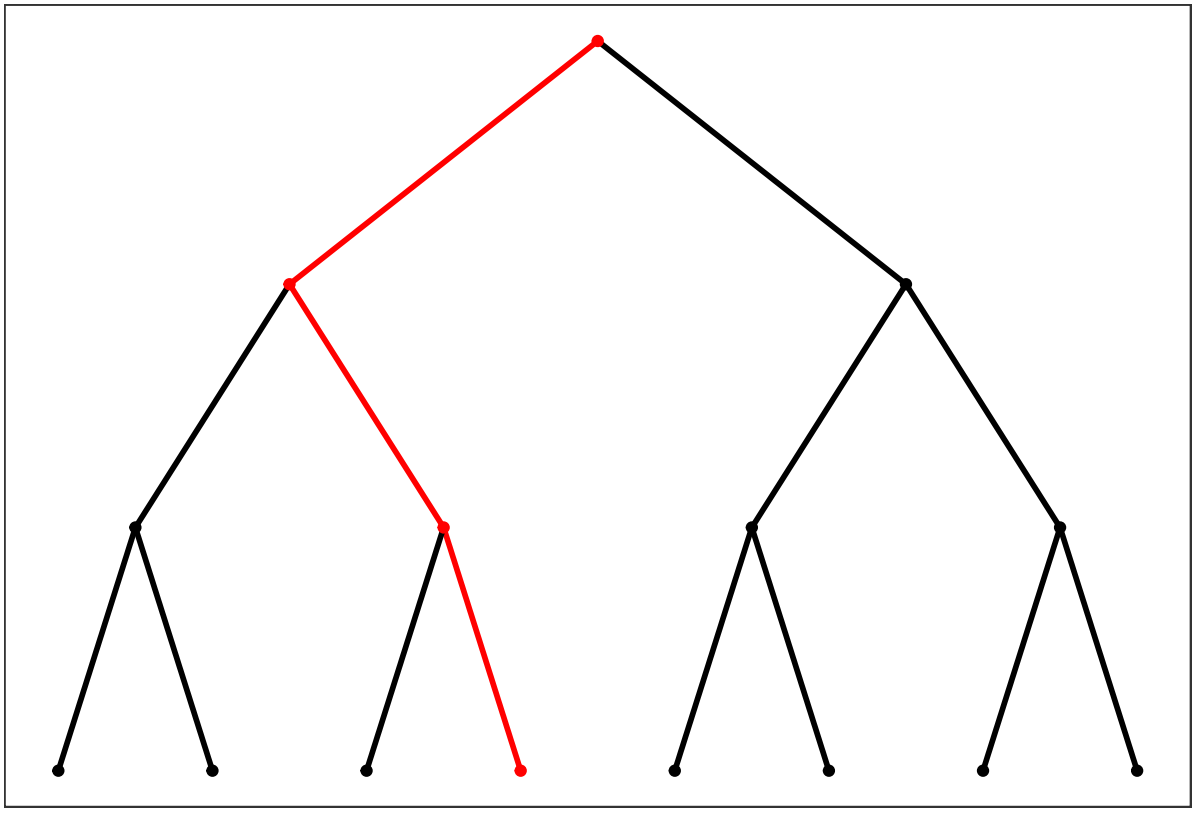
```
# GRAPHICAL REPRESENTATION OF THE GARDEN OF FORKING PATHS ######################

# Define size of nodes ------------------------------------------------------

size.nodes <- 1.5

# Create a balanced binary tree with height 3 -------------------------------

tree <- make_tree(15, children = 2, mode = "out")

# Create a tree plot with all edges highlighted in red ----------------------

all.paths <- ggraph(tree, layout = "dendrogram") +
  geom_edge_link(color = "red", width = 1) +
  geom_node_point(size = size.nodes, color = "red") +
  theme_AP() +
  labs(x = "", y = "") +
  theme(legend.position = "none",
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank())

all.paths
```



```r
# Create a tree plot with only one analytical path highlighted -----------------

# Define the path to highlight (from root to a specific node) ------------------

highlight_nodes <- c(1, 2, 5, 11)  # Path: 1 → 2 → 5 → 11

highlight_edges <- apply(cbind(head(highlight_nodes, -1),
                               tail(highlight_nodes, -1)), 1, function(x)
                                 paste(x, collapse = "-"))

# Assign default colors (black) to all edges and nodes -------------------------

E(tree)$edge_color <- "black"
V(tree)$node_color <- "black"

# Extract edges from the tree and match with highlight_edges -------------------

edge_list <- apply(get.edgelist(tree), 1, function(x) paste(x, collapse = "-"))
```

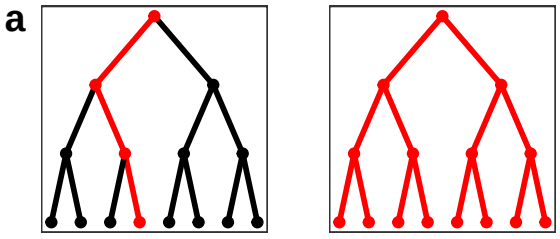
```
## Warning: `get.edgelist()` was deprecated in igraph 2.0.0.
## i Please use `as_edgelist()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
E(tree)$edge_color[edge_list %in% highlight_edges] <- "red"

# Highlight the selected nodes in red -___-------------------------------------

V(tree)$node_color[highlight_nodes] <- "red"

# Plot the tree with explicitly defined colors for both edges and nodes --------

one.path <- ggraph(tree, layout = "dendrogram") +
  geom_edge_link(aes(edge_color = edge_color), width = 1) +  # Correct edge colors
  geom_node_point(aes(color = node_color), size = size.nodes) +  # Correct node colors
  scale_edge_color_manual(values = c("black" = "black", "red" = "red")) +  # Fix for edges
  scale_color_manual(values = c("black" = "black", "red" = "red")) +  # Fix for nodes
  theme_AP() +
  labs(x = "", y = "") +
  theme(legend.position = "none",
        axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank())

one.path
```



```
# MERGE FORKING PATHS ############################################################
```

```r
plot_grid(one.path, all.paths, ncol = 2, labels = c("a", ""))
```




```r
# DEFINE THE UNCERTAINTY SPACE ########################################################

# Target year ---------------------------------------------------------------

## Defined above

# Target year interval ------------------------------------------------------

target_year_interval <- c("yes", "no")

# Interval publication ------------------------------------------------------

interval <- c(10, 15, 20)

# Metrics of study ----------------------------------------------------------

metrics <- c("cv", "range", "sd", "var", "entropy", "iqr")

# Inclusion criteria --------------------------------------------------------

exclude_before_1990 <- c("yes", "no")

# Rolling windows -----------------------------------------------------------

rolling_window_factor <- c(1, 0.5)

# Define the forking paths --------------------------------------------------

forking_paths <- expand.grid(target_year = target_year,
                             target_year_interval = target_year_interval,
                             interval = interval,
                             exclude_before_1990 = exclude_before_1990,
                             rolling_window_factor = rolling_window_factor,
                             metric = c(metrics, paste(metrics, "_normalized", sep = ""))) %>%
  data.table()

# Number of simulations -----------------------------------------------------
```

```r
nrow(forking_paths)
```

```
## [1] 1440
```

```r
# RUN MODEL ##################################################################

# Select only simulations at the global level of iww ---------------------------

dt <- references.full.dt[variable == "iww" & region == "global"]

# Run simulations --------------------------------------------------------------

trend <- list()

for (i in 1:nrow(forking_paths)) {

  trend[[i]] <- forking_paths_fun(dt = dt,
                                  target_year = forking_paths[[i, "target_year"]],
                                  target_year_interval = forking_paths[[i, "target_year_interva
                                  interval = forking_paths[[i, "interval"]],
                                  rolling_window_factor = forking_paths[[i, "rolling_window_fa
                                  exclude_before_1990 = forking_paths[[i, "exclude_before_1990
                                  metric = forking_paths[[i, "metric"]])
}
```

```r
# ARRANGE DATA ###############################################################

output.dt <- lapply(trend, function(x) x[["results"]]) %>%
  do.call(rbind, .) %>%
  data.table() %>%
  setnames(., "V1", "trend")

final.dt <- cbind(forking_paths, output.dt)

# Export simulations -----------------------------------------------------------

fwrite(final.dt, "forking.paths.dataset.csv")

# Print the fraction of simulations in each classification ---------------------

final.dt %>%
  .[, .(total = .N), trend] %>%
  .[, fraction:= total / nrow(output.dt)] %>%
  print()
```

```
##           trend total   fraction
##          <char> <int>      <num>
## 1:       Random   621 0.43125000
## 2:    Ascending   488 0.33888889
```

```
## 3:   Descending   271 0.18819444
## 4: single point    60 0.04166667
```

```
# Now remove all simulations that produced just one single point --------------

final.dt <- final.dt[!trend == "single point"]

# Simulations that did not lead to a reduction in uncertainty -----------------

final.dt %>%
  .[, .(total = .N), trend] %>%
  .[, fraction:= total / nrow(output.dt)] %>%
  .[!trend == "Descending"] %>%
  .[, sum(fraction)]
```

```
## [1] 0.7701389
```

```
# PLOTS FORKING PATHS EXAMPLES ###################################################

plots.dt <- lapply(trend, function(x) x[["plot"]])

random.plots <- c(1, 986, 345)
decreasing.plots <- c(1093, 556, 4)
increasing.plots <- c(10, 602, 770)

out.random <- out.decreasing <- out.increasing <- list()

for (i in 1:length(random.plots)) {

  out.random[[i]] <- plot_plots_forking_paths_fun(random.plots[i])
  out.decreasing[[i]] <- plot_plots_forking_paths_fun(decreasing.plots[i])
  out.increasing[[i]] <- plot_plots_forking_paths_fun(increasing.plots[i])
}

pt.random <- plot_grid(out.random[[1]] + geom_smooth() + labs(x = "", y = "+ Uncertainty"),
                       out.random[[2]] + geom_smooth() + labs(x = "", y = ""),
                       out.random[[3]] + geom_smooth() + labs(x = "", y = ""),
                       ncol = 3)
```

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

```
pt.decreasing <- plot_grid(out.decreasing[[1]] + geom_smooth() + labs(x = "", y = "+ Uncertain
                           out.decreasing[[2]] + geom_smooth() + labs(x = "", y = ""),
                           out.decreasing[[3]] + geom_smooth(method = "lm", se = F) + labs(x =
                           ncol = 3)
```

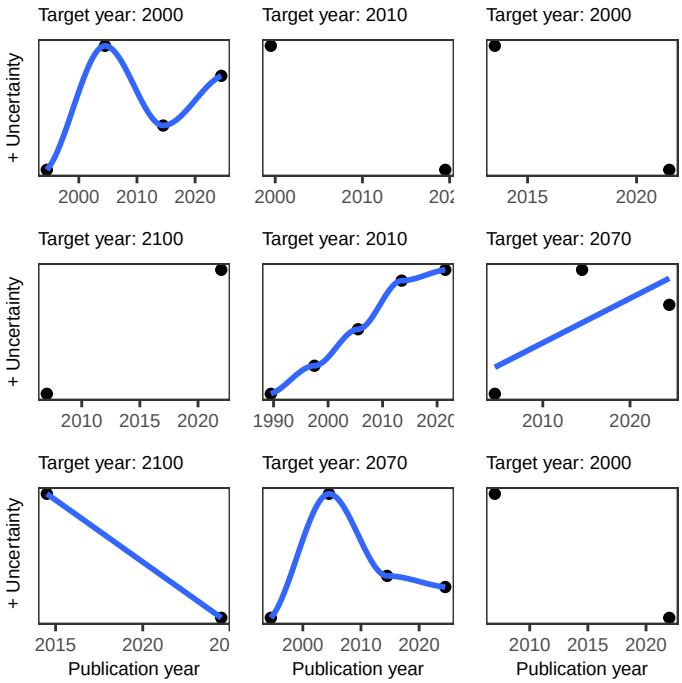
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

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

```
pt.increasing <- plot_grid(out.increasing[[1]] + geom_smooth(method = "lm", se = F),
                           out.increasing[[2]] + geom_smooth() + labs(x = "Publication year", y
                           out.increasing[[3]] + geom_smooth() + labs(x = "Publication year", y
                           ncol = 3)
```

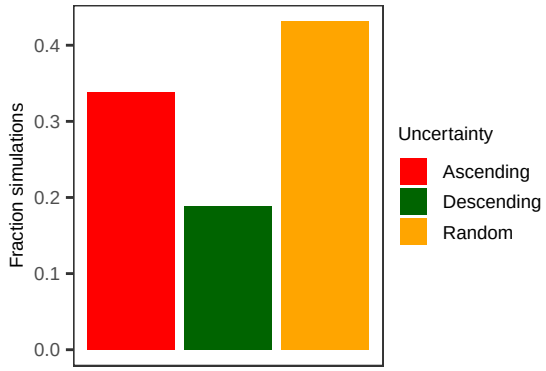
```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
plot.examples.trends <- plot_grid(pt.random, pt.decreasing, pt.increasing, ncol = 1)
plot.examples.trends
```




```
# PLOT RESULTS ###################################################################

selected_colors <- c("Ascending" = "red", "Descending" = "darkgreen", "Random" = "orange")

plot.fraction <- final.dt[, .(total = .N), trend] %>%
  .[, fraction:= total / nrow(output.dt)] %>%
  ggplot(., aes(trend, fraction, fill = trend)) +
  geom_bar(stat = "identity") +
  labs(x = "", y = "Fraction simulations") +
  scale_fill_manual(values = selected_colors, name = "Uncertainty") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  theme_AP() +
  theme(axis.ticks.x = element_blank(),
        axis.text.x = element_blank(),
        legend.position = "right")
```

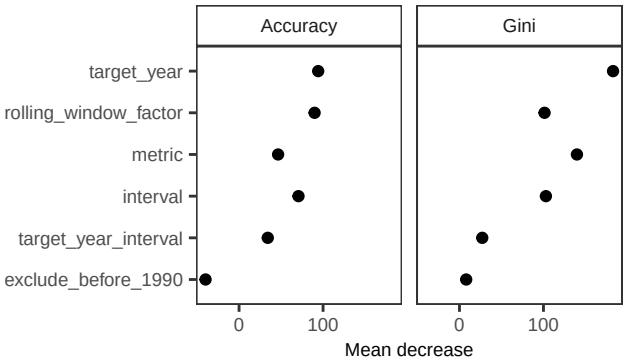
```
plot.fraction
```




```
# RANDOM FOREST ##################################################################

# Convert categorical variables to factors -----------------------------------

df <- data.frame(final.dt)
df$exclude_before_1990 <- as.factor(final.dt$exclude_before_1990)
df$metric <- as.factor(final.dt$metric)
df$trend <- as.factor(df$trend)
df$target_year_interval <- as.factor(df$target_year_interval)

# Train the model ------------------------------------------------------------

rf_model <- randomForest(trend ~ target_year + target_year_interval + interval +
                           exclude_before_1990 + rolling_window_factor + metric,
                         data = df, importance = TRUE)

# View variable importance ---------------------------------------------------

dt_rf_model <- data.frame(importance(rf_model))
dt_rf_model
```

```
##                       Ascending Descending    Random MeanDecreaseAccuracy
## target_year            58.48813   90.63844  71.27962             94.20959
## target_year_interval   27.35199   11.65554  26.97276             34.18253
## interval               40.86415   49.00795  62.65107             70.74657
## exclude_before_1990   -31.03482  -18.85533 -23.29184            -39.86391
## rolling_window_factor  55.53992   34.72830  85.64432             89.92829
## metric                 45.16112   28.87487  22.13455             46.47483
##                       MeanDecreaseGini
## target_year                  182.83710
## target_year_interval          27.14430
## interval                     102.90392
## exclude_before_1990            8.01515
## rolling_window_factor        101.29024
## metric                       139.86534
```

```
# Plot ------------------------------------------------------------------------

plot.rf <- dt_rf_model %>%
  rownames_to_column(., var = "factors") %>%
  data.table() %>%
  setnames(., c("MeanDecreaseAccuracy", "MeanDecreaseGini"),
           c("Accuracy", "Gini")) %>%
  melt(., measure.vars = c("Accuracy", "Gini")) %>%
  ggplot(., aes(reorder(factors, value), value)) +
  geom_point() +
  coord_flip() +
  facet_wrap(~variable) +
  scale_y_continuous(breaks = breaks_pretty(n = 3)) +
  labs(x = "", y = "Mean decrease") +
  theme_AP()

plot.rf
```

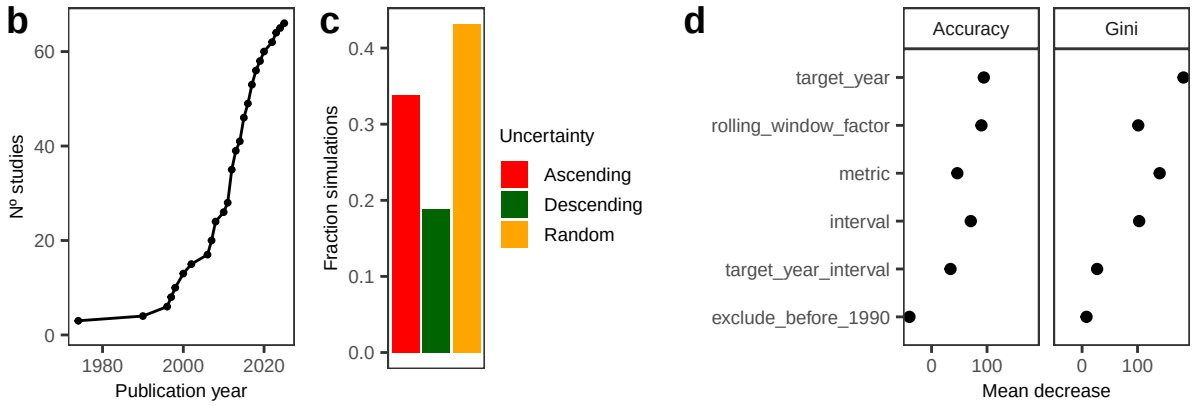



```
bottom <- plot_grid(cumulative.iww, plot.fraction, plot.rf, ncol = 3, labels = c("b", "c", "d"
          rel_widths = c(0.26, 0.3, 0.44))
```

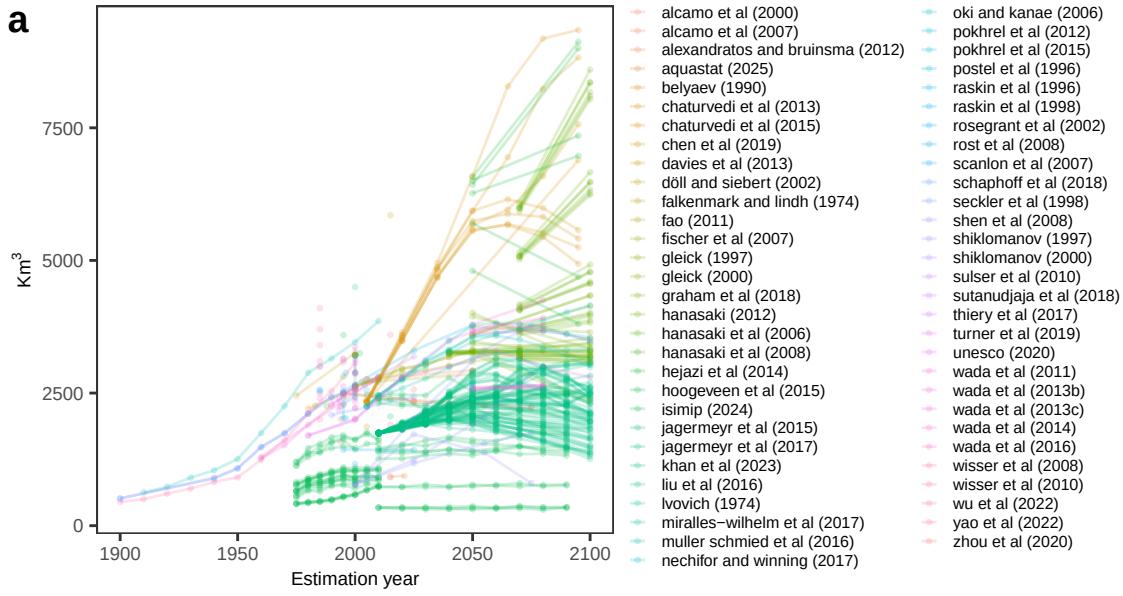
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

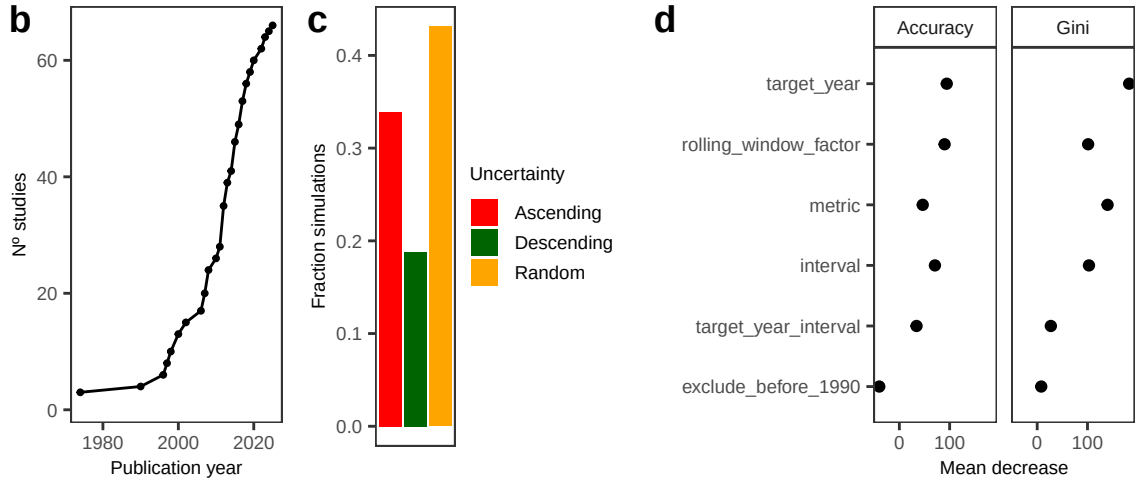
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
bottom
```

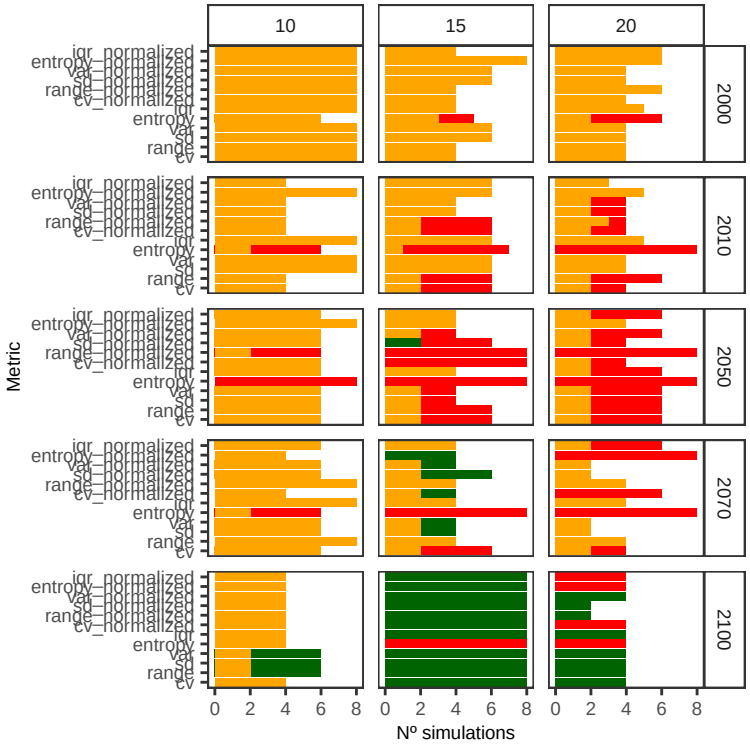



```
#
final.faceted.plot <- plot_grid(plot.iww, bottom, ncol = 1, labels = c("a", ""),
                                rel_heights = c(0.55, 0.45))

final.faceted.plot
```





```
# RESULTS FACETED BY INTERVAL AND TARGET YEAR, X AXIS METRICS ###################

plot.faceted.metrics <- final.dt %>%
  ggplot(., aes(x = factor(metric), fill = trend)) +
  geom_bar(position ="identity") +
  facet_grid(target_year ~ interval, scales = "free_y") +
  scale_fill_manual(values = selected_colors, name = "Uncertainty") +
  theme_AP() +
  labs(x = "Metric", y = "Nº simulations") +
  theme(legend.position = "none") +
  coord_flip()

plot.faceted.metrics
```

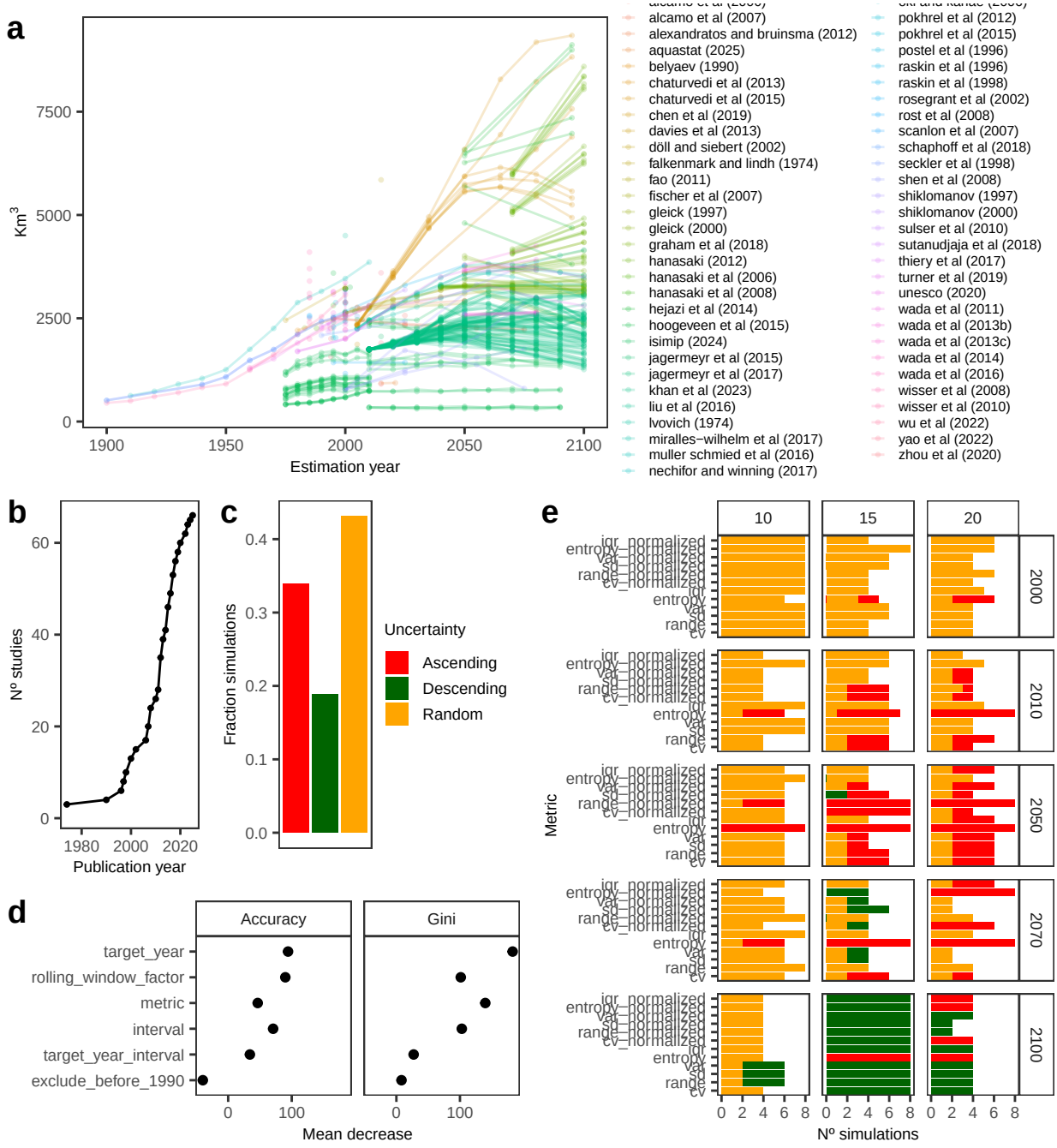



```
bottom <- plot_grid(cumulative.iww, plot.fraction, ncol = 2, rel_widths = c(0.4, 0.6),
                    labels = c("b", "c"))
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
left <- plot_grid(bottom, plot.rf, ncol = 1, labels = c("", "d"), rel_heights = c(0.6, 0.4))
bottom2 <- plot_grid(left, plot.faceted.metrics, ncol = 2, labels = c("", "e"))
plot_grid(plot.iww, bottom2, rel_heights = c(0.42, 0.58), ncol = 1, labels = c("a", ""))
```



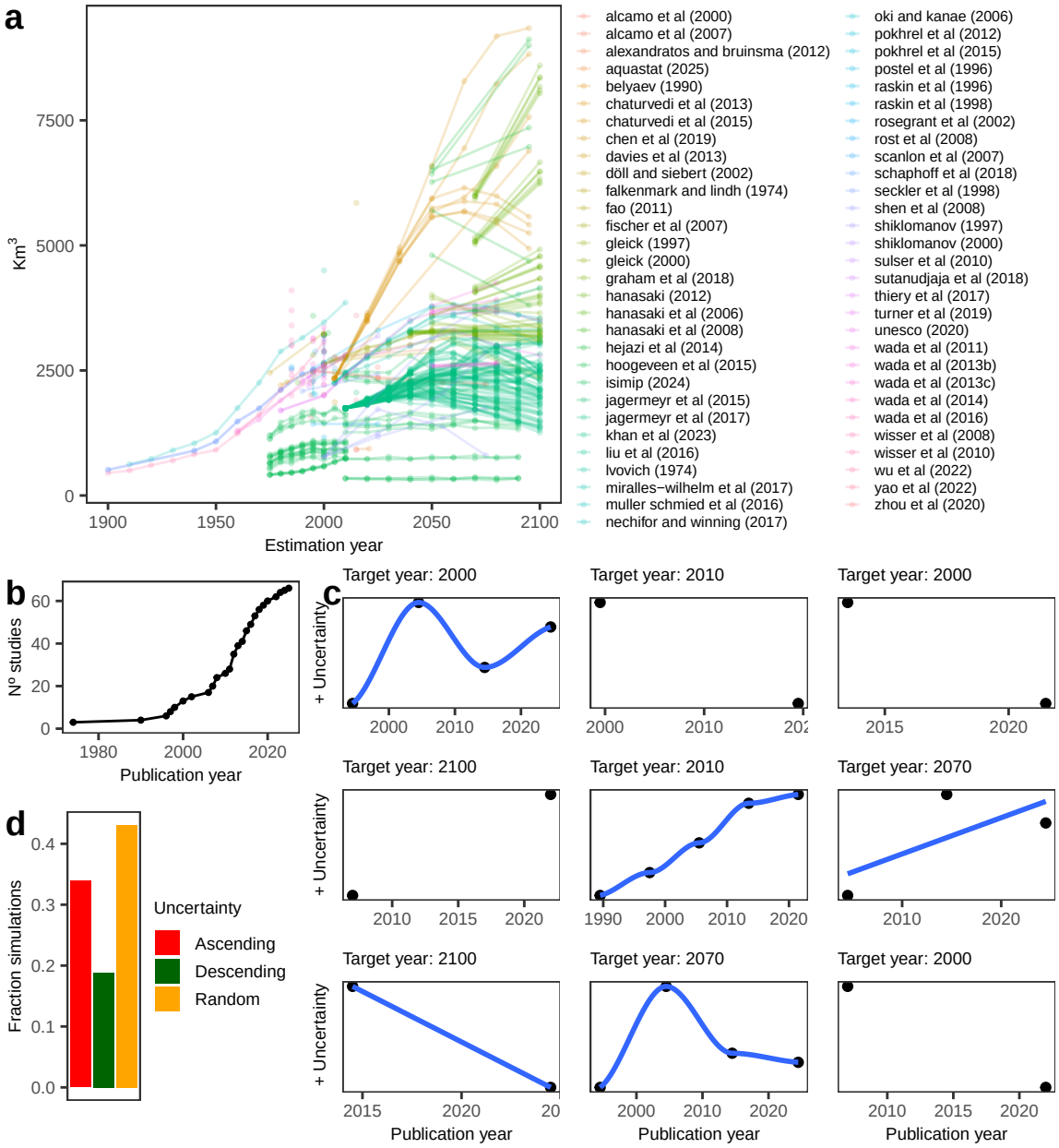

```
left <- plot_grid(cumulative.iww, plot.fraction, ncol = 1, rel_heights = c(0.4, 0.6),
                  labels = c("b", "d"))
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
bottom <- plot_grid(left, plot.examples.trends, ncol = 2, rel_widths = c(0.3, 0.7),
                    labels = c("", "c"))
```

```
plot_grid(plot.iww, bottom, ncol = 1, rel_heights = c(0.5, 0.5), labels = c("a", ""))
```

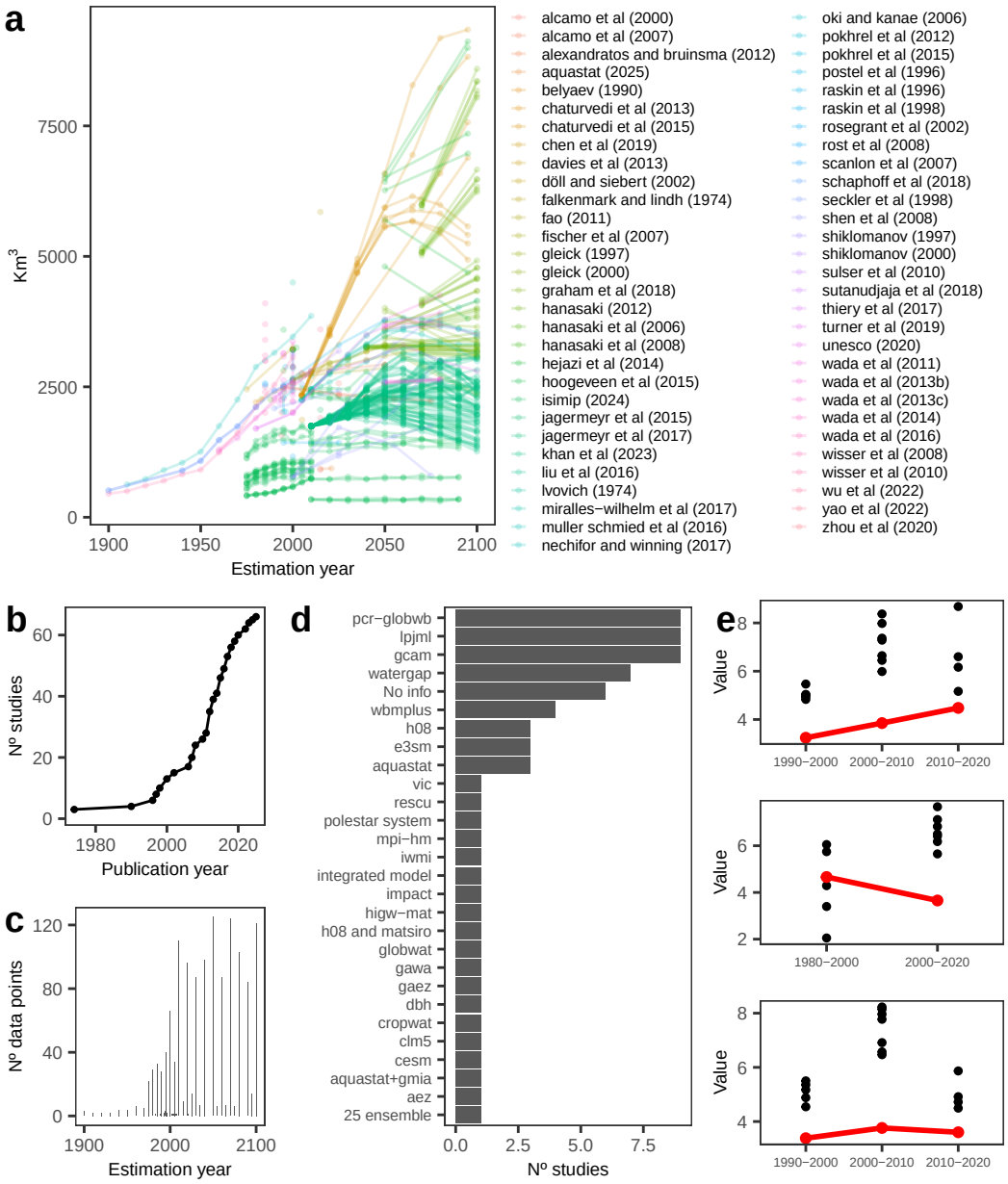



```
left <- plot_grid(cumulative.iww, plot.bar, ncol = 1, labels = c("b", "c"))
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
bottom <- plot_grid(left, plot.models, ncol = 2, labels = c("", "d"), rel_widths = c(0.4, 0.6))
bottom.right <- plot_grid(bottom, plot.examples.trends.data, ncol = 2, rel_widths = c(0.7, 0.3))
plot_grid(plot.iww, bottom.right, ncol = 1, rel_heights = c(0.5, 0.5), labels = c("a", ""))
```

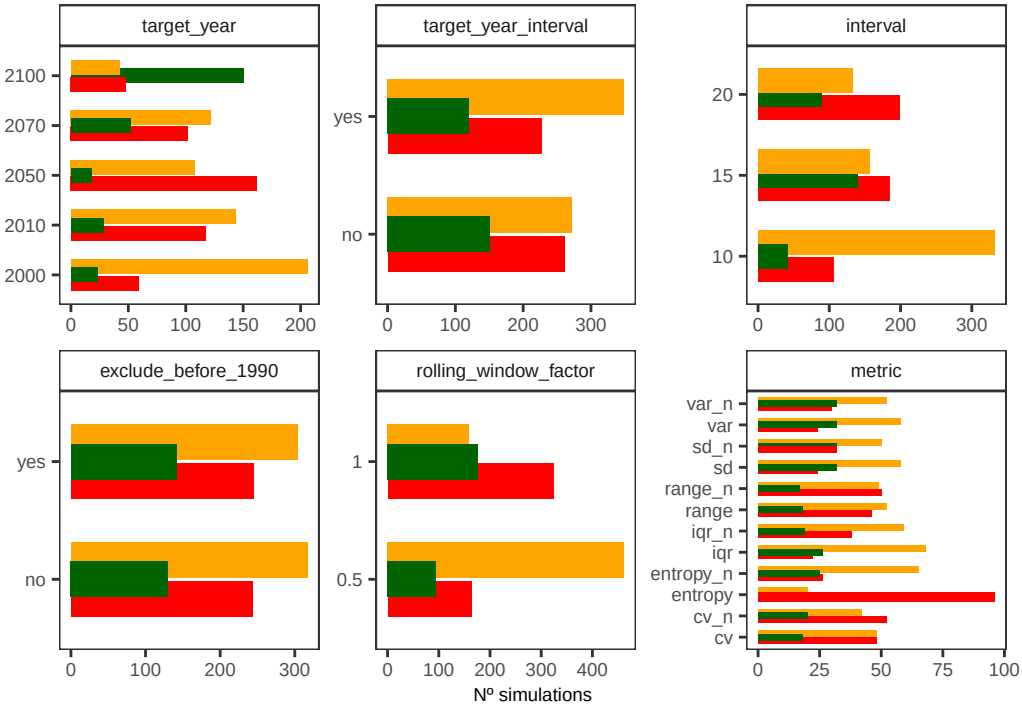



```
# SENSITIVITY ANALYSIS PLOT BY FACET ##########################################

plot.sa.facet <- final.dt %>%
  melt(., measure.vars = c("target_year", "target_year_interval", "interval",
                           "exclude_before_1990", "rolling_window_factor", "metric")) %>%
  .[, .N, .(variable, value, trend)] %>%
  .[ , value := gsub("_normalized", "_n", value)] %>%
  ggplot(., aes(value, N, fill = trend)) +
  scale_fill_manual(values = selected_colors, name = "Uncertainty") +
  geom_bar(stat = "identity", position = position_dodge(0.5)) +
  facet_wrap(~variable, scale = "free") +
  labs(x = "", y = "Nº simulations") +
  theme_AP() +
```

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

```
## Warning in melt.data.table(., measure.vars = c("target_year",
## "target_year_interval", : 'measure.vars' [target_year, target_year_interval,
## interval, exclude_before_1990, ...] are not all of the same type. By order of
## hierarchy, the molten data value column will be of type 'character'. All
## measure variables not of type 'character' will be coerced too. Check DETAILS in
## ?melt.data.table for more on coercion.
```

```
plot.sa.facet
```

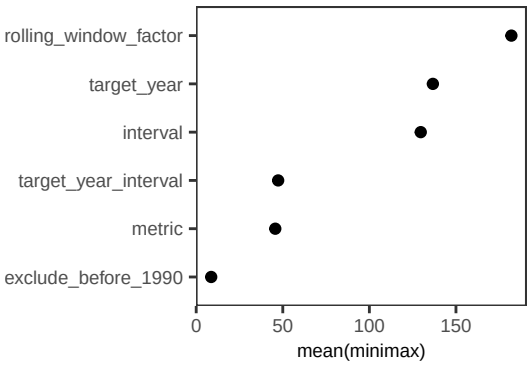



```
# PLOT AVERAGE MINIMAX AS SENSITVITY INDEX ######################################

plot.sa <- final.dt %>%
  melt(., measure.vars = c("target_year", "target_year_interval", "interval",
                           "exclude_before_1990", "rolling_window_factor", "metric")) %>%
  .[, .N, .(variable, value, trend)] %>%
  .[, .(minimax = max(N) - min(N)), .(variable, trend)] %>%
  .[, .(mean = mean(minimax),
        sd = sd(minimax)), variable] %>%
  ggplot(., aes(reorder(variable, mean), mean)) +
  geom_point() +
  labs(x = "", y = "mean(minimax)") +
  coord_flip() +
  theme_AP()
```

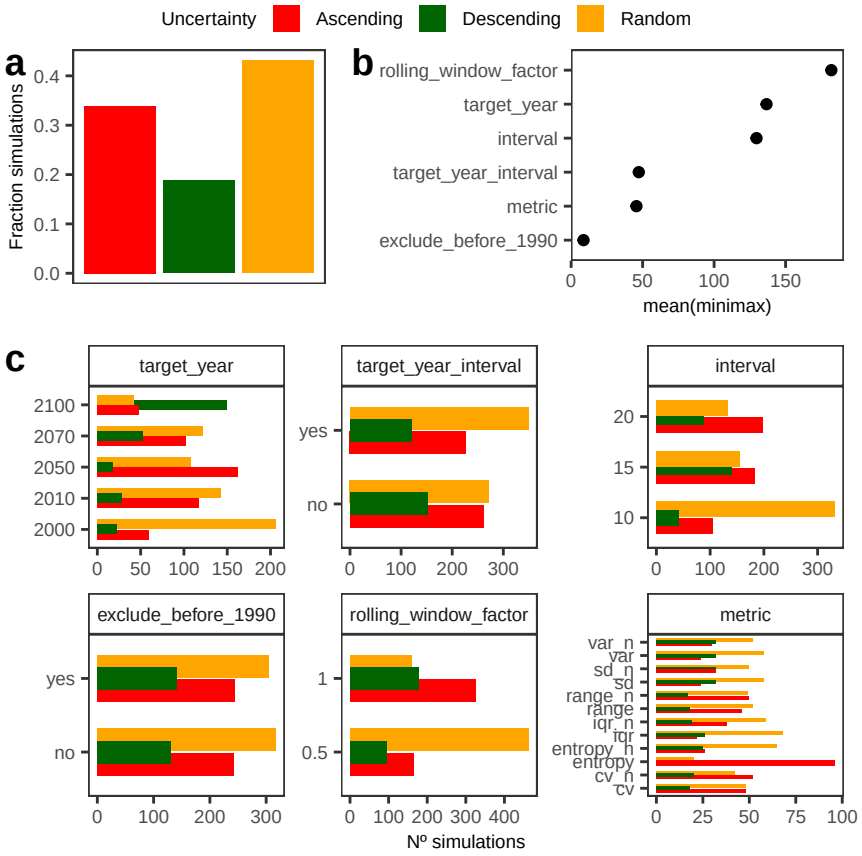
```
## Warning in melt.data.table(., measure.vars = c("target_year",
## "target_year_interval", : 'measure.vars' [target_year, target_year_interval,
```

```
## interval, exclude_before_1990, ...] are not all of the same type. By order of
## hierarchy, the molten data value column will be of type 'character'. All
## measure variables not of type 'character' will be coerced too. Check DETAILS in
## ?melt.data.table for more on coercion.
```

```
plot.sa
```




```
# MERGE SENSITIVITY ANALYSIS PLOTS ##############################################

legend <- get_legend_fun(plot.fraction + theme(legend.position = "top"))
top <- plot_grid(plot.fraction + theme(legend.position = "none"), plot.sa, ncol = 2,
                 rel_widths = c(0.4, 0.6), labels = c("a", "b"))
top.with.legend <- plot_grid(legend, top, rel_heights = c(0.1, 0.9), ncol = 1)
plot_grid(top.with.legend, plot.sa.facet, ncol = 1, rel_heights = c(0.38, 0.62),
          labels = c("", "c"))
```


Uncertainty
Ascending
Descending
Random
a
Fraction simulations
0.0
0.1
0.2
0.3
0.4
b
rolling_window_factor
target_year
interval
target_year_interval
metric
exclude_before_1990
0
50
100
150
mean(minimax)
c
target_year
2100
2070
2050
2010
2000
target_year_interval
yes
no
interval
20
15
10
exclude_before_1990
yes
no
rolling_window_factor
1
0.5
metric
var_n
var
sd_n
sd
range_n
range
iqr_n
iqr
entropy_n
entropy
cv_n
cv
0
25
50
75
100
200
300
400
Nº simulations

3 Session information

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

sessionInfo()
```

```
## R version 4.3.3 (2024-02-29)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.2.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/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
##
## time zone: Europe/London
## tzcode source: internal
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] randomForest_4.7-1.2 brms_2.22.0          Rcpp_1.0.13-1
##  [4] mgcv_1.9-1           nlme_3.1-166         microbenchmark_1.5.0
##  [7] lme4_1.1-35.5        Matrix_1.6-5         here_1.0.1
## [10] doParallel_1.0.17    iterators_1.0.14     foreach_1.5.2
## [13] rworldmap_1.3-8      sp_2.1-4             countrycode_1.6.0
## [16] ncdf4_1.23           scales_1.3.0         wesanderson_0.3.7
## [19] benchmarkme_1.0.8    cowplot_1.1.3        lubridate_1.9.3
## [22] forcats_1.0.0        stringr_1.5.1        dplyr_1.1.4
## [25] purrr_1.0.2          readr_2.1.5          tidyr_1.3.1
## [28] tibble_3.2.1         ggplot2_3.5.1        tidyverse_2.0.0
## [31] data.table_1.16.2    openxlsx_4.2.7.1
##
## loaded via a namespace (and not attached):
##  [1] Rdpack_2.6.2          rlang_1.1.4           magrittr_2.0.3
##  [4] matrixStats_1.4.1     compiler_4.3.3        loo_2.8.0
##  [7] vctrs_0.6.5           maps_3.4.2.1          crayon_1.5.3
## [10] pkgconfig_2.0.3       fastmap_1.2.0         backports_1.5.0
## [13] labeling_0.4.3        utf8_1.2.4            rmarkdown_2.29
## [16] tzdb_0.4.0            nloptr_2.1.1          tinytex_0.54
## [19] xfun_0.49             terra_1.7-78          R6_2.5.1
## [22] stringi_1.8.4         boot_1.3-31           estimability_1.5.1
## [25] knitr_1.49            fields_16.3           bayesplot_1.11.1
## [28] splines_4.3.3         timechange_0.3.0      tidyselect_1.2.1
```

```
## [31] rstudioapi_0.17.1     abind_1.4-8           yaml_2.3.10
## [34] codetools_0.2-20      lattice_0.22-6        withr_3.0.2
## [37] bridgesampling_1.1-2  benchmarkmeData_1.0.4 posterior_1.6.0
## [40] coda_0.19-4.1         evaluate_1.0.1        RcppParallel_5.1.9
## [43] zip_2.3.1             pillar_1.9.0          tensorA_0.36.2.1
## [46] checkmate_2.3.2       distributional_0.5.0  generics_0.1.3
## [49] rprojroot_2.0.4       hms_1.1.3             rstantools_2.4.0
## [52] munsell_0.5.1         minqa_1.2.8           sensobol_1.1.5
## [55] xtable_1.8-4          glue_1.8.0            emmeans_1.10.5
## [58] tools_4.3.3           mvtnorm_1.3-2         dotCall64_1.2
## [61] grid_4.3.3            rbibutils_2.3         colorspace_2.1-1
## [64] raster_3.6-30         cli_3.6.3             spam_2.11-0
## [67] fansi_1.0.6           viridisLite_0.4.2     Brobdingnag_1.2-9
## [70] gtable_0.3.6          digest_0.6.37         farver_2.1.2
## [73] htmltools_0.5.8.1     lifecycle_1.0.4       httr_1.4.7
## [76] MASS_7.3-60.0.1
```

```
## Return the machine CPU ----------------------------------------------------

cat("Machine:     "); print(get_cpu()$model_name)
```

```
## Machine:
## [1] "Apple M1 Max"
```

```
## Return number of true cores -----------------------------------------------

cat("Num cores:   "); print(detectCores(logical = FALSE))
```

```
## Num cores:
## [1] 10
```

```
## Return number of threads --------------------------------------------------

cat("Num threads: "); print(detectCores(logical = FALSE))
```

```
## Num threads:
## [1] 10
```