Irrigation's real impact on global water and food security

R code

Arnald Puy

Contents

1 Retrieve all corpus **6**
1.1 Abstract corpus 6
1.2 Policy corpus 6
1.3 Full text corpus 7

2 Split full text corpus for analysis **8**

3 Network analysis **9**
3.1 Network metrics 13
3.2 Network plots 15

4 Analysis of paths **22**

5 Session information **31**

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

sensobol::load_packages(c("openxlsx", "data.table", "tidyverse", "bibliometrix",
                          "igraph", "ggraph", "cowplot", "tidygraph", "benchmarkme",
                          "parallel", "wesanderson", "scales"))

# 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.margin = margin(0.5, 0.1, 0.1, 0.1),
          legend.box.margin = margin(0.2,-4,-7,-7),
          plot.margin = margin(3, 4, 0, 4),
          legend.text = element_text(size = 8),
          axis.title = element_text(size = 10),
          legend.key.width = unit(0.4, "cm"),
          legend.key.height = unit(0.4, "cm"),
          legend.title = element_text(size = 9))
}
```

```r
# CREATION OF VECTORS WITH NAMES ###################################################

database <- c("wos", "scopus", "dimensions")
topic <- c("water", "food")

# Create all possible combinations
combinations <- expand.grid(database = database, topic = topic)

# Combine the vectors with an underscore
file.name <- paste(combinations$database, "dt", combinations$topic, sep = "_")

# READ IN THE DATA #################################################################

# Loop to create the file names ------------------------------------------------

for (i in 1:length(file.name)) {

  database.type <- str_extract(file.name, "^(wos|scopus|dimensions)")

  if(isTRUE(database.type[i] == "wos")) {

    file.name[i] <- paste(file.name[i], "bib", sep = ".")

  } else {

    file.name[i] <- paste(file.name[i], "csv", sep = ".")
  }

}

# vector with new column names -------------------------------------------------

new_colnames <- c("doi", "authors", "year", "title", "journal", "abstract", "database")
to_lower <- c("authors", "title", "journal", "abstract")

# Loop to read in the datasets -------------------------------------------------

out <- list()

for (i in 1:length(file.name)) {

  database.type <- str_extract(file.name[i], "^(wos|scopus|dimensions)")

  if(isTRUE(database.type == "wos")) {

    out[[i]] <- convert2df(file = file.name[i],
                dbsource = "wos",
```

```
                format = "bibtex") %>%
      data.table() %>%
      .[, .(DI, AU, PY, TI, SO, AB)] %>%
      .[, database:= "wos"]

  } else if (isTRUE(database.type == "dimensions")) {

    out[[i]] <- fread(file.name[i], skip = 1) %>%
      .[, .(DOI, Authors, PubYear, Title, `Source title`, Abstract)] %>%
      .[, database:= "dimensions"]

  } else if(isTRUE(database.type == "scopus")) {

    out[[i]] <- fread(file.name[i]) %>%
      .[, .(DOI, Authors, Year, Title, `Source title`, Abstract)] %>%
      .[, database:= "scopus"]
  }

  setnames(out[[i]], colnames(out[[i]]), new_colnames) %>%
    .[, (to_lower):= lapply(.SD, tolower), .SDcols = (to_lower)] %>%
    .[, abstract:= sub("references.*", "", abstract)]

}
```

```
##
## Converting your wos collection into a bibliographic dataframe
##
##
## Warning:
## In your file, some mandatory metadata are missing. Bibliometrix functions may not work prope
##
## Please, take a look at the vignettes:
## - 'Data Importing and Converting' (https://www.bibliometrix.org/vignettes/Data-Importing-and
## - 'A brief introduction to bibliometrix' (https://www.bibliometrix.org/vignettes/Introductio
##
##
## Missing fields:  C1 CR
## Done!
##
##
## Converting your wos collection into a bibliographic dataframe
##
##
## Warning:
## In your file, some mandatory metadata are missing. Bibliometrix functions may not work prope
##
## Please, take a look at the vignettes:
```

```
## - 'Data Importing and Converting' (https://www.bibliometrix.org/vignettes/Data-Importing-and
## - 'A brief introduction to bibliometrix' (https://www.bibliometrix.org/vignettes/Introductio
##
##
## Missing fields:  C1 CR
## Done!
```

```
names(out) <- combinations$topic

# CLEAN THE DATASETS ##############################################################

# Arrange ---------------------------------------------------------------------

dt <- rbindlist(out, idcol = "topic")

tmp <- split(dt, list(dt$topic, dt$database))

cols_to_merge_by <- c("doi", "year", "title", "journal", "abstract")

dt.water <- merge(merge(tmp$water.dimensions, tmp$water.scopus, by = cols_to_merge_by,
            all = TRUE), tmp$water.wos, by = cols_to_merge_by,
      all = TRUE)

dt.food <- merge(merge(tmp$food.dimensions, tmp$food.scopus, by = cols_to_merge_by,
            all = TRUE), tmp$food.wos, by = cols_to_merge_by,
      all = TRUE)

# Filer out duplicated studies by doi -----------------------------------------

tmp.list <- list(dt.water, dt.food)
duplicated.dois <- final.dt <- list()

for (i in 1:length(tmp.list)) {

  duplicated.dois[[i]] <- duplicated(tmp.list[[i]]$doi, incomparables = NA, na.rm = TRUE)
  final.dt[[i]] <- tmp.list[[i]][!duplicated.dois[[i]]][, location.belief.system:= "abstract"]

}

names(final.dt) <- topic

# Check if there is any duplicated doi ----------------------------------------

any(duplicated(final.dt$food$doi, na.rm = TRUE, incomparables = NA))
```

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

```
# Export to xlsx --------------------------------------------------------------
```

```r
for (i in names(final.dt)) {

  write.xlsx(final.dt[[i]][, .(doi, year, title, abstract, location.belief.system)],
             paste("final.dt", names(final.dt[i]), "xlsx", sep = "."))
}
```

1 Retrieve all corpus

1.1 Abstract corpus

```r
final.dt.water.screened <- data.table(read.xlsx("final.dt.water_screened.xlsx"))
final.dt.food.screened <- data.table(read.xlsx("final.dt.food_screened.xlsx"))
screened.dt <- list(final.dt.water.screened, final.dt.food.screened)
names(screened.dt) <- c("water", "food")

lapply(screened.dt, function(x) x[, .N, screening])
```

```
## $water
##    screening     N
##       <char> <int>
## 1:         F   168
## 2:         T   163
##
## $food
##    screening     N
##       <char> <int>
## 1:         F   465
## 2:         T    39
```

```r
# Export for close-reading only the references that do include
# the belief system in the abstract ------------------------------------------

for (i in names(screened.dt)) {

  screened.dt[[i]][screening == "T"] %>%
    unique(., by = "title") %>%
    .[, .(doi, title, year)] %>%
    write.xlsx(., paste("abstract.corpus", i, "xlsx", sep = "."))

}
```

1.2 Policy corpus

```r
# LOAD IN DIMENSIONS DATASETS (POLICY TEXT) ###################################

# Function to load and preprocess data ----------------------------------------
```

```r
load_and_preprocess_data <- function(file_path, topic) {
  fread(file_path, skip = 1)[, topic := topic]
}

colnames.full.text <- c("doi", "year", "title", "journal", "topic")
keywords <- c("water", "irrigat")

# Load data ------------------------------------------------------------------

dt.policy.water <- load_and_preprocess_data("dimensions_dt_policy.csv", "water")
dt.policy.food <- load_and_preprocess_data("dimensions_dt_policy_food.csv", "food")

dimensions.full.text.policy <- rbind(dt.policy.food, dt.policy.water) %>%
  .[, .(`Policy document ID`, PubYear, Title, `Publishing Organization`,
        `Sustainable Development Goals`, `Source Linkout`, topic)]

dimensions.full.text.policy[, .N, topic]
```

```
##     topic     N
##    <char> <int>
## 1:   food 10573
## 2:  water  3455
```

```r
# Create a logical condition for pattern matching using grepl
pattern_condition_policy <- sapply(keywords, function(keyword)
  grepl(keyword, dimensions.full.text.policy$Title, ignore.case = TRUE))

# Combine conditions with OR using rowSums
matching.rows.policy <- dimensions.full.text.policy[rowSums(pattern_condition_policy) > 0]

matching.rows.policy[, .N, topic]
```

```
##     topic     N
##    <char> <int>
## 1:   food   750
## 2:  water   450
```

```r
# Export ---------------------------------------------------------------------

for (i in c("water", "food")) {

  matching.rows.policy[topic == i] %>%
    write.xlsx(paste("policy.corpus", i, "xlsx", sep = "."))
}
```

1.3 Full text corpus

```r
# LOAD IN DIMENSIONS DATASET (FULL TEXT) ########################################

full.text.corpus.water <- fread("full.text.corpus.water.csv")
```

2 Split full text corpus for analysis

```r
# SPLIT THE DATASET INTO N FOR RESEARCH #########################################

# Function to split dataset in n chunks ---------------------------------------

split_dt_fun <- function(dt, num_parts) {

  split_dt <- list()

  # Calculate the number of rows in each part
  rows_per_part <- nrow(dt) %/% num_parts

  # Split the data.table into roughly equal parts
  for (i in 1:num_parts) {

    start_row <- (i - 1) * rows_per_part + 1
    end_row <- i * rows_per_part

    if (i == num_parts) {

      end_row <- nrow(dt)
    }
    split_dt[[i]] <- dt[start_row:end_row, ]
  }

  return(split_dt)

}

# Create the datasets for close reading ---------------------------------------

times.nanxin <- 2
times.arnald <- 1
nanxin <- paste(rep("nanxin", times.nanxin), 1:times.nanxin, sep = "")
arnald <- paste(rep("arnald", times.arnald), 1:times.arnald, sep = "")
names_surveyors <- c(arnald, nanxin, "seth", paste("student", 1:4, sep = ""))
n.surveyors <- length(names_surveyors)

survey.dt.split <- split_dt_fun(dt = full.text.corpus.water, num_parts = n.surveyors)
names(survey.dt.split) <- names_surveyors
```

```r
# Export ------------------------------------------------------------------

for (i in 1:length(survey.dt.split)) {

  write.xlsx(survey.dt.split[[i]],
             file = paste0(names(survey.dt.split)[i], ".dt", ".xlsx"))

}
```

3 Network analysis

```r
# CREATE VECTORS TO READ IN AND CLEAN THE DATASETS ##############################

tmp <- list()
names.files <- c("WORK", "NETWORK")
topics <- c("water")
corpus <- c("abstract.corpus", "policy.corpus", "full.text.corpus")
cols_of_interest <- c("title", "author", "claim", "citation")

# Paste all possible combinations of names ----------------------------------

combs <- expand.grid(corpus = corpus, topics = topics, approach = names.files)
all.files <- paste(paste(paste(combs$corpus, combs$topics, sep = "."), combs$approach, sep = "_
                   "xlsx", sep = ".")

# READ IN DATASETS AND TURN TO LOWERCAPS ########################################

tmp <- list()

for (i in 1:length(all.files)) {

  tmp[[i]] <- data.table(read.xlsx(all.files[i]))

  if (!str_detect(all.files[i], "NETWORK")) {

    tmp[[i]][, title:= tolower(title)]
  } else {

    tmp[[i]][, (cols_of_interest):= lapply(.SD, tolower), .SDcols = (cols_of_interest)]
  }
}

names(tmp) <- all.files


# CLEAN AND MERGE DATASETS ######################################################
```

```r
dataset.networks <- all.files[str_detect(all.files, "NETWORK")]
network.dt <- tmp[dataset.networks] %>%
  rbindlist() %>%
  .[, policy:= grepl("^policy", doi)]

network.dt[, author:= ifelse(policy == TRUE, doi, author)]

# CHECK NUMBER OF FAO AQUASTAT CITES ##############################################

network.dt[citation %like% "fao aquastat"] %>%
  .[, .N, citation]
```

```
##              citation     N
##                 <char> <int>
##  1: fao aquastat 2006     1
##  2:      fao aquastat     8
##  3: fao aquastat 2010     5
##  4: fao aquastat 2020     2
##  5: fao aquastat 2011     2
##  6: fao aquastat 2012     3
##  7: fao aquastat 2021     2
##  8: fao aquastat 2017     1
##  9: fao aquastat 2015     2
## 10: fao aquastat 2019     3
## 11: fao aquastat 2016     4
## 12: fao aquastat 2014     1
## 13: fao aquastat 2023     1
## 14: fao aquastat 2018     1
## 15: fao aquastat 2004     1
```

```r
# WRITE LOOKUP TABLE TO CHECK ALREADY RETRIEVED STUDIES ########################

lookup.dt <- network.dt[, .(doi, title, author)] %>%
  .[order(title)] %>%
  unique(.)

nrow(lookup.dt)
```

```
## [1] 518
```

```r
write.xlsx(lookup.dt, "lookup.dt.xlsx")

# Remove the year from mentions to FAO Aquastat --------------------------------

pattern <- "\\b(?:19|20)\\d{2}\\b"  # Matches years between 1900 and 2099

for (col in c("citation", "author")) {
  matches <- grepl("^fao aquastat\\s+\\d+$", network.dt[[col]], ignore.case = TRUE)
```

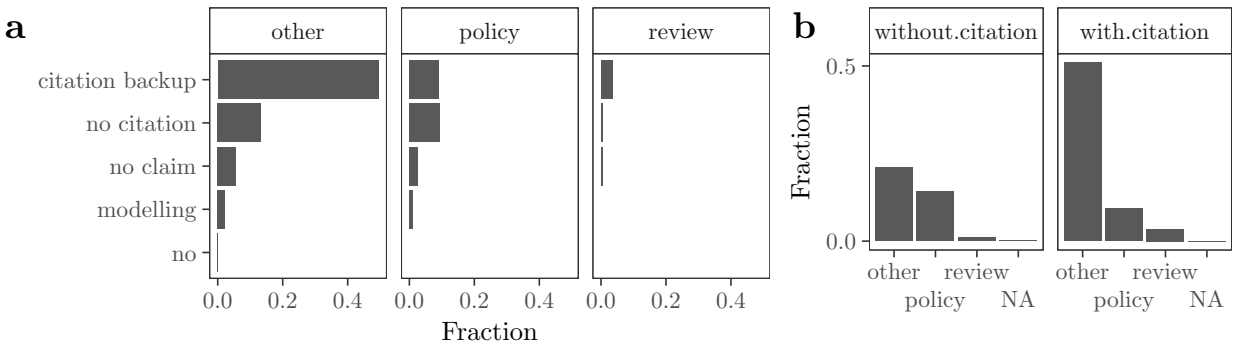
```
  network.dt[matches, (col) := gsub("\\d+", "", network.dt[[col]][matches], perl = TRUE)]
  network.dt[, (col) := trimws(network.dt[[col]])]
}

# Rename columns ----------------------------------------------------------

setnames(network.dt, c("author", "citation"), c("from", "to"))

# Create copy and remove duplicated ---------------------------------------

network.dt.claim <- copy(network.dt)
network.dt.claim <- unique(network.dt.claim,
                           by = c("from", "to", "document.type", "nature.claim"))

fwrite(network.dt.claim, "network.dt.claim.csv")

# Convert all to lower caps -----------------------------------------------

network.dt <- network.dt[, .(from, to, document.type, nature.claim)]
cols_to_change <- colnames(network.dt)
network.dt[, (cols_to_change):= lapply(.SD, trimws), .SDcols = (cols_to_change)]
```

```
# PLOT DESCRIPTIVE STATISTICS ##########################################################

total.rows <- nrow(network.dt)

# Check proportion of studies by nature of claim --------------------------

network.dt.claim[, .N, nature.claim] %>%
  .[, total:= total.rows] %>%
  .[, fraction:= N / total] %>%
  print()
```

```
##       nature.claim     N total    fraction
##             <char> <int> <int>       <num>
## 1: citation backup   398   642 0.619937695
## 2:       modelling    21   642 0.032710280
## 3:     no citation   147   642 0.228971963
## 4:            <NA>    16   642 0.024922118
## 5:        no claim    55   642 0.085669782
## 6:              no     1   642 0.001557632
```

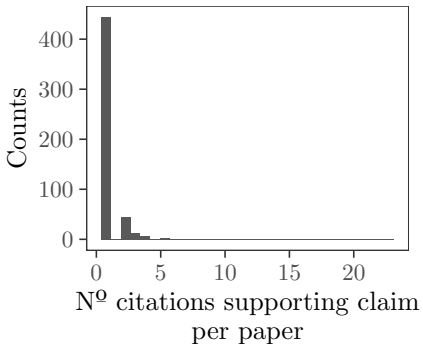
```
# Count document type by nature of claim ----------------------------------

a <- network.dt[, .N, .(nature.claim, document.type)] %>%
  .[, total.rows:= total.rows] %>%
  .[, proportion:= N / total.rows] %>%
  na.omit() %>%
```

```
  ggplot(., aes(reorder(nature.claim, proportion), proportion)) +
  coord_flip() +
  geom_bar(stat = "identity") +
  facet_wrap(~document.type) +
  scale_y_continuous(breaks = breaks_pretty(n = 2)) +
  labs(x = "", y = "Fraction") +
  theme_AP()

# Count how many documents make the claim and cite / do not cite,
# by document.type ------------------------------------------------------------

b <- network.dt[, .(without.citation = sum(is.na(to)),
                    with.citation = .N - sum(is.na(to))), document.type] %>%
  melt(., measure.vars = c("without.citation", "with.citation")) %>%
  .[, total.rows:= total.rows] %>%
  .[, proportion:= value / total.rows] %>%
  ggplot(., aes(document.type, proportion)) +
  geom_bar(stat = "identity") +
  scale_y_continuous(breaks = breaks_pretty(n = 2)) +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  labs(x = "", y = "Fraction") +
  facet_wrap(~variable) +
  theme_AP()

# merge -----------------------------------------------------------------------

plot_grid(a, b, ncol = 2, rel_widths = c(0.63, 0.37), labels = "auto")
```




```
# PLOT DISTRIBUTION OF CITATION SUPPORTING THE CLAIM ############################

network.dt[, .N, from] %>%
  .[order(-N)] %>%
  ggplot(., aes(N)) +
  geom_histogram() +
  theme_AP() +
  labs(x = "Nº citations supporting claim \n per paper", y = "Counts")
```

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



3.1 Network metrics

```
# CALCULATE NETWORK METRICS ###############################################################

# only complete cases ----------------------------------------------------------

network.dt.complete <- network.dt[complete.cases(network.dt$to), ]

# Transform to graph -----------------------------------------------------------

citation_graph <- graph_from_data_frame(d = network.dt.complete, directed = TRUE)

# Calculate network metrics ----------------------------------------------------

edge_density(citation_graph)
```

```
## [1] 0.0022697
```

```
# Modularity:
# - c.1: Strong community structure, where nodes within groups are highly connected.
# - c. -1: Opposite of community structure, where nodes between groups are more connected.
# - c. 0: Indicates absence of community structure or anti-community structure in the network.
wtc <- cluster_walktrap(citation_graph)
modularity(wtc)
```

```
## [1] 0.8251864
```

```
network_metrics <- data.table(node = V(citation_graph)$name,

                              # Degree of a node: The number of connections or
                              # edges linked to that node.
                              # It represents how well-connected or central a
                              # node is within the graph.
                              degree = degree(citation_graph, mode = "in"),

                              degree.out = degree(citation_graph, mode = "out"),

                              # Betweenness centrality of a node: Measures the
```

```
                                # extent to which a node lies on the shortest
                                # paths between all pairs of other nodes in the graph.
                                # Nodes with high betweenness centrality act as
                                # bridges or intermediaries, facilitating
                                # communication and information flow between other nodes.
                                betweenness = betweenness(citation_graph),

                                # Closeness centrality of a node: Measures how
                                # close a node is to all other nodes in the graph,
                                # taking into account the length of the shortest paths.
                                # Nodes with high closeness centrality are able to
                                # efficiently communicate or interact with other
                                # nodes in the graph.
                                closeness = closeness(citation_graph),
                                pagerank = page_rank(citation_graph)$vector
)

# Define the max number of rows
max.number <- 3

degree.nodes <- network_metrics[order(-degree)][1:max.number]
degree.nodes.out <- network_metrics[order(-degree.out)][1:max.number]
betweenness.nodes <- network_metrics[order(-betweenness)][1:max.number]
pagerank.nodes <- network_metrics[order(-closeness)][1:max.number]

degree.nodes
```

```
##                 node degree degree.out betweenness closeness   pagerank
##               <char>  <num>      <num>       <num>     <num>      <num>
## 1:      fao aquastat     37          0         0.0       NaN 0.07388163
## 2:          fao 2011     10          1        10.5 1.0000000 0.01188210
## 3: molden et al 2007      9          1        20.0 0.3333333 0.01164597
```

```
degree.nodes.out
```

```
##               node degree degree.out betweenness  closeness    pagerank
##             <char>  <num>      <num>       <num>      <num>       <num>
## 1:       wada 2015      0         23           0 0.02702703 0.001301589
## 2: wada et al 2014      1          9          10 0.09090909 0.001522859
## 3: wada et al 2016      0          8           0 0.06250000 0.001301589
```

```
betweenness.nodes
```

```
##                   node degree degree.out betweenness  closeness    pagerank
##                 <char>  <num>      <num>       <num>      <num>       <num>
## 1: boretti and rosa 2019      2          4    22.00000 0.05555556 0.003514290
## 2:     molden et al 2007      9          1    20.00000 0.33333333 0.011645966
## 3:    siebert et al 2010      6          3    16.33333 0.33333333 0.005498505
```

```
pagerank.nodes
```

```
##                     node degree degree.out betweenness closeness    pagerank
##                   <char>  <num>      <num>       <num>     <num>       <num>
## 1: sharma and irmak 2012      0          1           0         1 0.001301589
## 2:       world bank 2007      3          1          10         1 0.009815555
## 3:   brajovic et al 2015      0          1           0         1 0.001301589
```

3.2 Network plots

```
# ADD FEATURES TO NODES ############################################################

# Retrieve a vector with the node names ----------------------------------------

graph <- tidygraph::as_tbl_graph(network.dt.complete, directed = TRUE)
vec.names <- graph %>%
  activate(nodes) %>%
  pull() %>%
  data.table(name = .)

# Merge with info from the network.dt ------------------------------------------

vec.nature.claim <- merge(merge(vec.names, unique(network.dt[, .(from, nature.claim)]),
                                by.x = "name", by.y = "from", all.x = TRUE),
                          unique(network.dt[, .(from, document.type)]),
                          by.x = "name", by.y = "from", all.x = TRUE)

# Merge with the correct order -------------------------------------------------

order_indices <- match(vec.names$name, vec.nature.claim$name)
final.vec.nature.claim <- vec.nature.claim[order_indices, ] %>%
  .[, nature.claim]
final.vec.document.type <- vec.nature.claim[order_indices, ] %>%
  .[, document.type]

# Attach to the graph ----------------------------------------------------------

graph <- graph %>%
  activate(nodes) %>%
  mutate(nature.claim = final.vec.nature.claim,
         document.type = final.vec.document.type,
         degree = network_metrics$degree,
         degree.out = network_metrics$degree.out,
         betweenness = network_metrics$betweenness,
         pagerank = network_metrics$pagerank)
```

```r
# NUMBER OF NODES ###################################################################

V(graph)
```

```
## + 425/425 vertices, named, from ce3cf5c:
##   [1] sharma and irmak 2012
##   [2] doreau et al 2012
##   [3] world water assessment programme 2009
##   [4] world bank 2007
##   [5] brajovic et al 2015
##   [6] rivers et al 2015
##   [7] kijne 2005
##   [8] hafeez and khalid awan 2022
##   [9] dunkelman et al 2017
##  [10] nordin et al 2013
## + ... omitted several vertices
```

```r
# NUMBER OF EDGES ###################################################################

ecount(graph)
```

```
## [1] 409
```

```r
# PROPORTION OF ALL PATHS THAT PASS THROUGH FIVE HIGHEST BETWEENNESS NODES ######

bc <- betweenness(graph)
nodes_of_interest <- sort(bc, decreasing = TRUE)[1:5]
total_paths <- choose(vcount(graph), 2)  # Total number of paths
total_paths
```

```
## [1] 90100
```

```r
sum(nodes_of_interest) / total_paths
```

```
## [1] 0.0009387717
```

```r
# PROPORTION OF LINKS CONNECTED TO THE 5 NODES WITH HIGHEST DEGREE #############

dg <- degree(graph)
nodes_of_interest_degree <- sort(dg, decreasing = TRUE)[1:5]
total_edges <- ecount(graph)  # Total number of edges
sum(nodes_of_interest_degree) / total_edges
```

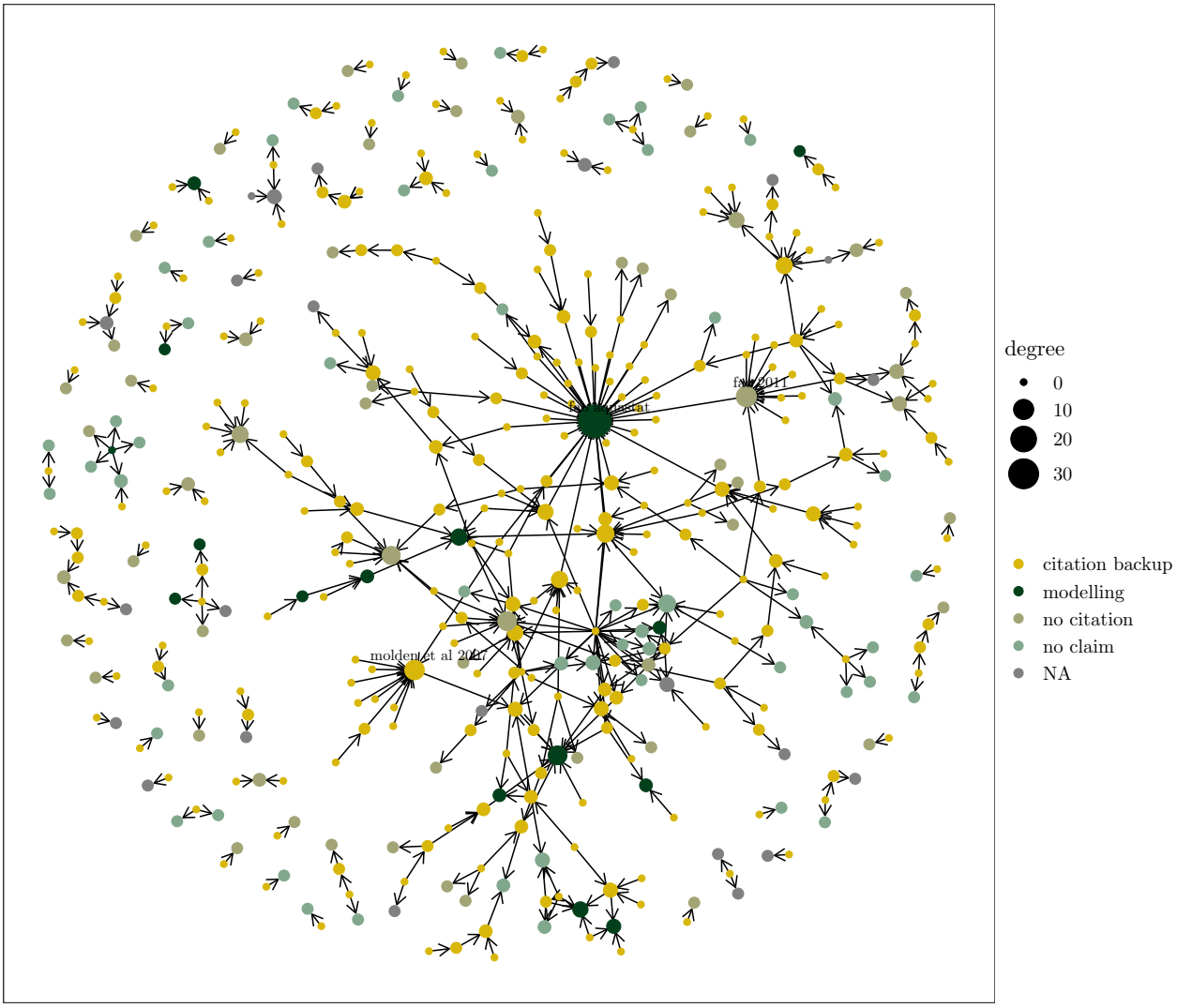
```
## [1] 0.2224939
```

```r
# PLOT NETWORK ######################################################################

seed <- 123

# by nature of claim ------------------------------------------------------
```

```r
set.seed(seed)

# Label the nodes with highest degree ----------------------------------------

ggraph(graph, layout = "igraph", algorithm = "nicely") +
  geom_edge_link(arrow = arrow(length = unit(1.8, 'mm')),
                 end_cap = circle(1, "mm")) +
  geom_node_point(aes(color = nature.claim, size = degree)) +
  geom_node_text(aes(label = ifelse(degree >= min(degree.nodes$degree), name, NA)),
                 repel = TRUE, size = 2.2) +
  labs(x = "", y = "") +
  scale_color_manual(name = "",
                     values = wes_palette(name = "Cavalcanti1", 5)) +
  theme_AP() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "right")
```

```
## Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` in the `default_aes` field and elsewhere instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

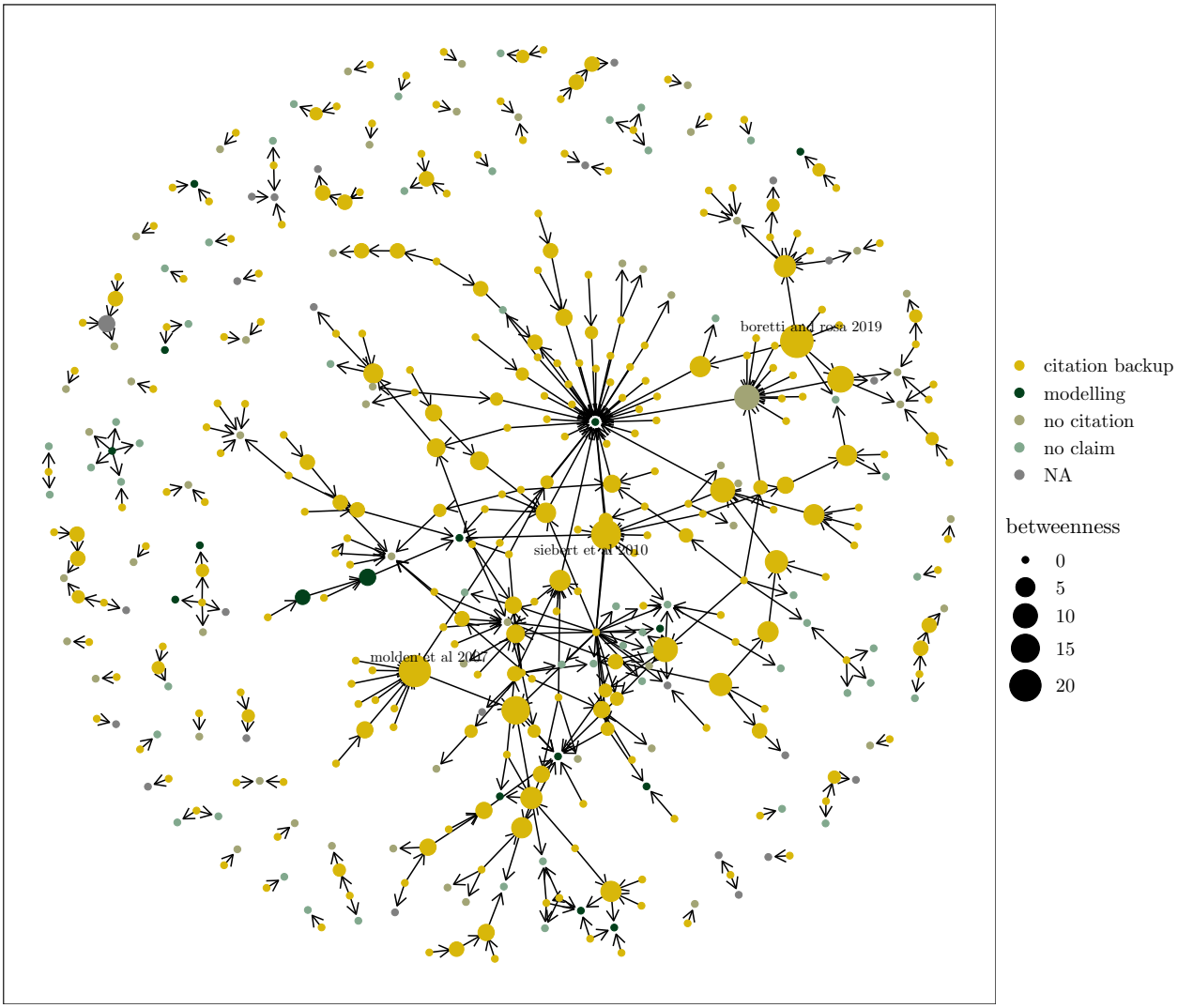
```
## Warning: Removed 422 rows containing missing values (`geom_text_repel()`).
```




```
set.seed(seed)

# Label the nodes with highest betweenness ----------------------------------

ggraph(graph, layout = "igraph", algorithm = "nicely") +
  geom_edge_link(arrow = arrow(length = unit(1.8, 'mm')),
                 end_cap = circle(1, "mm")) +
  geom_node_point(aes(color = nature.claim, size = betweenness)) +
  geom_node_text(aes(label = ifelse(betweenness >= min(betweenness.nodes$betweenness), name, NA
                 repel = TRUE, size = 2.2) +
  labs(x = "", y = "") +
  scale_color_manual(name = "",
                     values = wes_palette(name = "Cavalcanti1", 5)) +
  theme_AP() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_blank(),
```

```
        axis.ticks.y = element_blank(),
        legend.position = "right")
```

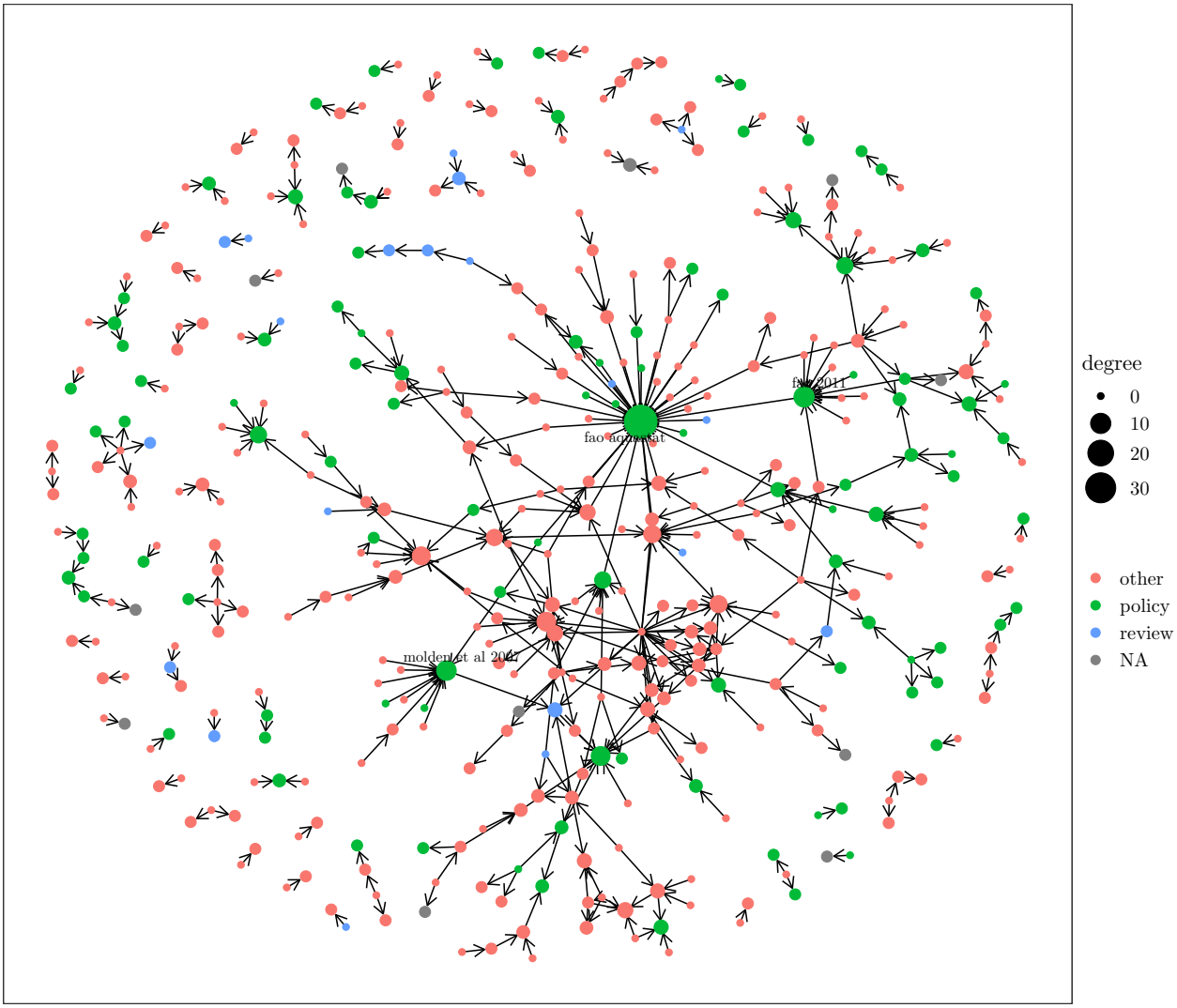
```
## Warning: Removed 422 rows containing missing values (`geom_text_repel()`).
```




```
# by document.type-----------------------------------------------------------

set.seed(seed)

ggraph(graph, layout = "igraph", algorithm = "nicely") +
  geom_edge_link(arrow = arrow(length = unit(1.8, 'mm')),
                 end_cap = circle(1, "mm")) +
  geom_node_point(aes(color = document.type, size = degree)) +
  geom_node_text(aes(label = ifelse(degree >= min(degree.nodes$degree), name, NA)),
                 repel = TRUE, size = 2.2) +
  labs(x = "", y = "") +
  scale_color_discrete(name = "") +
```

```
  theme_AP() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "right")
```

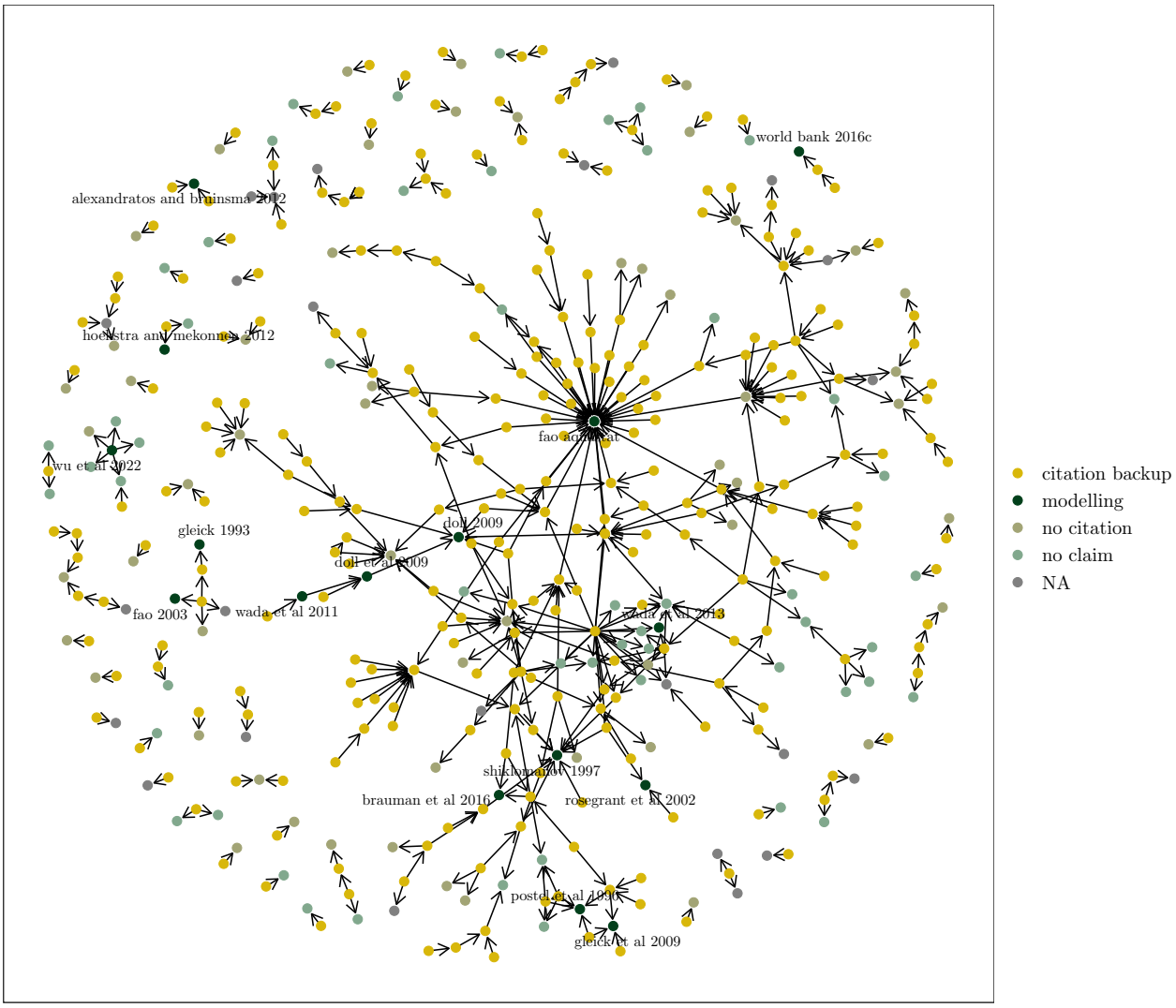
```
## Warning: Removed 422 rows containing missing values (`geom_text_repel()`).
```




```
# Label nodes that are modelling exercises -----------------------------------

set.seed(seed)

ggraph(graph, layout = "igraph", algorithm = "nicely") +
  geom_edge_link(arrow = arrow(length = unit(1.8, 'mm')),
                 end_cap = circle(1, "mm")) +
  geom_node_point(aes(color = nature.claim)) +
```

```
  geom_node_text(aes(label = ifelse(nature.claim == "modelling", name, NA)),
                 repel = TRUE, size = 2.2) +
  labs(x = "", y = "") +
  scale_color_manual(name = "",
                     values = wes_palette(name = "Cavalcanti1", 5)) +
  theme_AP() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "right")
```

```
## Warning: Removed 409 rows containing missing values (`geom_text_repel()`).
```



4 Analysis of paths

```
# COUNT THE NUMBER OF NODES WITH PATHS ULTIMATELY LEADING TO NODES
# THAT DO NOT MAKE THE CITATION ##################################################

# Function: loop through each node that do not make the claim to find all nodes
# connected to it ----------------------------------------------------------------

nodes_to_no_claim_node_fun <- function(g, terminal_nodes) {

  if (!is.igraph(g)) {
    g <- as.igraph(g)
  }

  all_predecessors <- vector("list", length(terminal_nodes))

  for (i in seq_along(terminal_nodes)) {

    terminal_node <- terminal_nodes[i]
    predecessors <- subcomponent(g, terminal_node, mode = "in")
    all_predecessors[[i]] <- predecessors
  }

  unique_predecessors <- unique(names(unlist(all_predecessors)))

  return(unique_predecessors)

}
```

```
# CALCULATE

# Extract name of all nodes ------------------------------------------------

all_nodes <- graph %>%
  activate(nodes) %>%
  pull(name)

# Extract name of nodes that do not make the claim -------------------------

no.claim_nodes <- graph %>%
  activate(nodes) %>%
  filter(degree.out == 0 & nature.claim == "no claim") %>%
  pull(., "name")

# Extract name of nodes that do not make the claim and those that make
# the claim but do not cite anybody ----------------------------------------

no.claim.and.no.citation.nodes <- graph %>%
```

```
  activate(nodes) %>%
  filter(degree.out == 0 & nature.claim == "no claim" | nature.claim == "no citation" ) %>%
  pull(., "name")

# Run the function ------------------------------------------------------------

out <- lapply(list(no.claim_nodes, no.claim.and.no.citation.nodes), function(x)
  sort(nodes_to_no_claim_node_fun(graph, terminal_nodes = x)))

names(out) <- c("path ending in no claim", "path ending in no claim or no citation")
out
```

```
## $`path ending in no claim`
##   [1] "abbot et al 2019"              "acosta et al 2016"
##   [3] "alcamo et al 2007"             "antia 2022"
##   [5] "badrul masud et al 2019"       "barreto and amaral 2018"
##   [7] "biemans et al 2011"            "bondeau et al 2007"
##   [9] "boretti and rosa 2019"         "braun et al 2022"
##  [11] "calzadilla et al 2010"         "carmona et al 2017"
##  [13] "carvalho 2019"                 "chai et al 2016"
##  [15] "chirone et al 2022"            "coelho et al 2012"
##  [17] "cristache et al 2018"          "d'odorico et al 2019"
##  [19] "doll et al 2014"               "droppers et al 2020"
##  [21] "eckert and kovalevska 2021"    "elmoneim badr et al 2021"
##  [23] "epri 2002"                     "faiz alam et al 2023"
##  [25] "falkenmark 2013"               "falkenmark et al 1997"
##  [27] "fao 2020"                      "friha et al 2022"
##  [29] "gan et al 2013"                "gerten et al 2007"
##  [31] "giordano 2007"                 "gleick and palaniappan 2010"
##  [33] "gleick et al 2011"             "gleick et al 2018"
##  [35] "gorjian et al 2020"            "gorjian et al 2022"
##  [37] "grigas et al 2023"             "gumidyala et al 2020"
##  [39] "hanasaki et al 2008"           "hanasaki et al 2008b"
##  [41] "hoekstra 2003"                 "hofste et al 2019"
##  [43] "huang et al 2023"              "iwmi 2000"
##  [45] "jaramillo and destouni 2015"   "johnson et al 2001"
##  [47] "jury and vaux jr 2005"         "kaba gurmessa and assefa 2023"
##  [49] "kabir et al 2023"              "karimi et al 2019"
##  [51] "kaur saggi and jain 2022"      "kiani et al 2023"
##  [53] "kilemo 2022"                   "kumar dubey et al 2021"
##  [55] "kumar ravi et al 2023"         "laluet et al 2024"
##  [57] "lamastra et al 2014"           "liu and yang 2010"
##  [59] "liu et al 2016"                "marston et al 2018"
##  [61] "mcdermid et al 2023"           "meghan salmon et al 2015"
##  [63] "mekonnen and hoekstra 2012"    "mekonnen et al 2015"
##  [65] "mohanty et al 2018"            "moldovan et al 2022"
##  [67] "nahar sumiya and khatun 2016"  "oladosu et al 2019"
```

```
##  [69] "oladosu et al 2022"           "opio et al 2011"
##  [71] "othmani et al 2021"           "ozdogan et al 2010"
##  [73] "payero et al 2006"            "pellegrini et al 2016"
##  [75] "perry et al 2017"             "policy.1255933"
##  [77] "policy.1435979"               "policy.1781691"
##  [79] "policy.1874989"               "qin et al 2019"
##  [81] "ran et al 2016"               "redhu and jain 2023"
##  [83] "ren et al 2018"               "rockstrom et al 2007"
##  [85] "rodriguez et al 2022"         "sadoff et al 2020"
##  [87] "sahmat et al 2022"            "scanlon et al 2017"
##  [89] "scanlon et al 2023"           "sepaskhah and ahmadi 2010"
##  [91] "shiklomanov 2000"             "shtull-trauring et al 2016"
##  [93] "siebert et al 2005"           "siebert et al 2010"
##  [95] "siebert et al 2015"           "singh et al 2024"
##  [97] "tabunshikov et al 2021"       "turner 2008"
##  [99] "unesco 2001"                  "united nations 1998"
## [101] "united nations 2003"          "united nations 2021"
## [103] "united nations 2022"          "velez sanchez et al 2023"
## [105] "vorosmarty et al 2000"        "vorosmarty et al 2010"
## [107] "wada 2015"                    "wada et al 2014"
## [109] "wada et al 2016"              "wajima 2018"
## [111] "walter et al 2017"            "wbcsd  2009"
## [113] "wisser et al 2010"            "wmo 1997"
## [115] "world bank 2001"              "world bank 2017"
## [117] "worldometers 2019"            "wri 2000"
## [119] "wu et al 2022"                "xu et al 2020"
## [121] "yilmazkuday et al 2021"       "yin et al 2022"
## [123] "young et al 2019"             "zhuo et al 2022"
##
## $`path ending in no claim or no citation`
##   [1] "abbot et al 2019"
##   [2] "abdullah 2006"
##   [3] "abou shady et al 2023"
##   [4] "abou zaki et al 2018"
##   [5] "acosta et al 2016"
##   [6] "adama et al 2020"
##   [7] "adhikari et al 2021"
##   [8] "alan rotz 2020"
##   [9] "alcamo et al 2007"
##  [10] "alvarez et al 2004"
##  [11] "anderson et al 2017"
##  [12] "angaleeswari et al 2021"
##  [13] "antia 2022"
##  [14] "arboleda et al 2022"
##  [15] "babel and wahid 2008"
##  [16] "bac-dang et al 2019"
##  [17] "bach et al 2017"
##  [18] "badrul masud et al 2019"
```

```
##  [19] "balyaminu 2017"
##  [20] "barker 2015"
##  [21] "barreto and amaral 2018"
##  [22] "basiri jahromi et al 2020"
##  [23] "bhaskar and jain 2018"
##  [24] "bicca rodrigues 2014"
##  [25] "biemans et al 2011"
##  [26] "biswas and tortajada 2010"
##  [27] "bondeau et al 2007"
##  [28] "bonsch et al 2016"
##  [29] "boretti and rosa 2019"
##  [30] "borin 2023"
##  [31] "boucher et al 2004"
##  [32] "bowden 2002"
##  [33] "braimoh 2013"
##  [34] "brar et al 2022"
##  [35] "braun et al 2022"
##  [36] "braune et al 2021"
##  [37] "brillo 2022"
##  [38] "brown 2009"
##  [39] "cai and rosegrant 2002"
##  [40] "caldera and breyer 2019"
##  [41] "calzadilla et al 2010"
##  [42] "carmona et al 2017"
##  [43] "carvalho 2019"
##  [44] "chai et al 2016"
##  [45] "chen et al 2018"
##  [46] "chilinda et al 2021"
##  [47] "chirone et al 2022"
##  [48] "clapp et al 2017"
##  [49] "coelho et al 2012"
##  [50] "connor 2017"
##  [51] "cristache et al 2018"
##  [52] "d'odorico et al 2019"
##  [53] "dalin et al 2012"
##  [54] "de pascale et al 2011"
##  [55] "doll 2008"
##  [56] "doll et al 2014"
##  [57] "doungmanee 2016"
##  [58] "droppers et al 2020"
##  [59] "dunkelman et al 2017"
##  [60] "eckert and kovalevska 2021"
##  [61] "elbakidze and cobourn 2014"
##  [62] "elmoneim badr et al 2021"
##  [63] "epri 2002"
##  [64] "evans and sadler 2008"
##  [65] "faiz alam et al 2023"
##  [66] "falkenmark 2013"
```

```
##  [67] "falkenmark et al 1997"
##  [68] "fao 2002"
##  [69] "fao 2002b"
##  [70] "fao 2007"
##  [71] "fao 2010"
##  [72] "fao 2011"
##  [73] "fao 2012"
##  [74] "fao 2012b"
##  [75] "fao 2017"
##  [76] "fao 2018"
##  [77] "fao 2019"
##  [78] "fao 2020"
##  [79] "fereres and soriano 2006"
##  [80] "firdayati et al 2022"
##  [81] "fitzgerald and auerbach 2016"
##  [82] "fogel and palmer 2014"
##  [83] "friha et al 2022"
##  [84] "gallardo 2015"
##  [85] "gan et al 2013"
##  [86] "gerbens-leenes and nonhebel 2004"
##  [87] "gerten et al 2007"
##  [88] "giordano 2007"
##  [89] "gleick and palaniappan 2010"
##  [90] "gleick et al 2011"
##  [91] "gleick et al 2014"
##  [92] "gleick et al 2018"
##  [93] "gorjian et al 2020"
##  [94] "gorjian et al 2022"
##  [95] "gourbesville 2008"
##  [96] "grigas et al 2023"
##  [97] "gumidyala et al 2020"
##  [98] "gurung 2016"
##  [99] "haddeland et al 2013"
## [100] "hanasaki et al 2008"
## [101] "hanasaki et al 2008b"
## [102] "hannah 2017"
## [103] "he et al 2023"
## [104] "hegazi et al 2023"
## [105] "hoekstra 2003"
## [106] "hofste et al 2019"
## [107] "hofwegen and svendsen 2000"
## [108] "huang et al 2023"
## [109] "hussein bapir and wasman hamad 2023"
## [110] "iaastd 2009"
## [111] "ingrao et al 2023"
## [112] "ipcc 2007"
## [113] "iwmi 2000"
## [114] "jagermeyr et al 2017"
```

```
## [115] "jaramillo and destouni 2015"
## [116] "jat et al 2016"
## [117] "jehan et al 2022"
## [118] "johnson et al 2001"
## [119] "jury and vaux jr 2005"
## [120] "kaba gurmessa and assefa 2023"
## [121] "kabir et al 2023"
## [122] "kapahi et al 2022"
## [123] "karimi et al 2019"
## [124] "kaur saggi and jain 2022"
## [125] "khosravifar et al 2020"
## [126] "kiani et al 2023"
## [127] "kilemo 2022"
## [128] "kiran kumara et al 2020"
## [129] "kocian and incrocci 2020"
## [130] "kumar dubey et al 2021"
## [131] "kumar ravi et al 2023"
## [132] "kundzewicz et al. 2007"
## [133] "laluet et al 2024"
## [134] "lamastra et al 2014"
## [135] "lang 2014"
## [136] "legesse lebre et al 2021"
## [137] "liu and yang 2010"
## [138] "liu et al 2016"
## [139] "lynch et al 2023"
## [140] "maldonado junior et al 2019"
## [141] "marston et al 2018"
## [142] "mashnik et al 2017"
## [143] "mcdermid et al 2023"
## [144] "meghan salmon et al 2015"
## [145] "mekonnen and hoekstra 2012"
## [146] "mekonnen et al 2015"
## [147] "mettetal 2019"
## [148] "millenium ecosystem assessment 2005"
## [149] "millenium project 2004"
## [150] "mohanty et al 2018"
## [151] "moldovan et al 2022"
## [152] "molle 2002"
## [153] "nahar sumiya and khatun 2016"
## [154] "newell and taylor 2017"
## [155] "nordin et al 2013"
## [156] "norton-brandao et al 2013"
## [157] "o'connell and billingsley 2020"
## [158] "odeku 2020"
## [159] "oecd 2010"
## [160] "oecd 2017"
## [161] "ohyama et al 2023"
## [162] "oladosu et al 2019"
```

```
## [163] "oladosu et al 2022"
## [164] "opio et al 2011"
## [165] "ostberg et al 2018"
## [166] "othmani et al 2021"
## [167] "ozdogan et al 2010"
## [168] "parameshwari 2017"
## [169] "pastor et al 2019"
## [170] "pauzuolien et al 2022"
## [171] "payero et al 2006"
## [172] "pedrero et al 2010"
## [173] "pellegrini et al 2016"
## [174] "perry et al 2017"
## [175] "pfister and bayer 2013"
## [176] "pokhrel et al 2012"
## [177] "pokhrel et al 2016"
## [178] "policy.1094742"
## [179] "policy.1252526"
## [180] "policy.1255933"
## [181] "policy.1257844"
## [182] "policy.1381456"
## [183] "policy.1435979"
## [184] "policy.1666264"
## [185] "policy.1781691"
## [186] "policy.1874989"
## [187] "policy.229461"
## [188] "policy.240747"
## [189] "policy.718260"
## [190] "postel 2001"
## [191] "qin et al 2019"
## [192] "rahmadian and widyartono 2019"
## [193] "ran et al 2016"
## [194] "redhu and jain 2023"
## [195] "ren et al 2018"
## [196] "ricart and rico 2019"
## [197] "ridgway et al 2019"
## [198] "ridoutt et al 2009"
## [199] "ringler et al 2022"
## [200] "ritchie and roser 2017"
## [201] "rivers et al 2015"
## [202] "rockstrom and gordon 2001"
## [203] "rockstrom et al 2007"
## [204] "rodriguez et al 2022"
## [205] "rodriguez-espinosa et al 2023"
## [206] "romano et al 2023"
## [207] "rosegrant and ringler 1998"
## [208] "rosegrant et al 2009"
## [209] "rost et al 2008"
## [210] "sadoff et al 2020"
```

```
## [211] "saeidian et al 2015"
## [212] "sahmat et al 2022"
## [213] "scanlon et al 2017"
## [214] "scanlon et al 2023"
## [215] "seckler et al 1998"
## [216] "sepaskhah and ahmadi 2010"
## [217] "shang et al 2024"
## [218] "shiklomanov 1999"
## [219] "shiklomanov 2000"
## [220] "shiklomanov and rodda 2003"
## [221] "shtull-trauring et al 2016"
## [222] "siebert and doll 2010"
## [223] "siebert et al 2005"
## [224] "siebert et al 2010"
## [225] "siebert et al 2013"
## [226] "siebert et al 2015"
## [227] "singh et al 2024"
## [228] "sophocleous 2004"
## [229] "steduto et al 2018"
## [230] "swatuk et al 2018"
## [231] "tabunshikov et al 2021"
## [232] "ti et al 2021"
## [233] "tsiropoulos et al 2022"
## [234] "tuninetti et al 2015"
## [235] "turner 2008"
## [236] "unctad 2011"
## [237] "unep 2011"
## [238] "unesco 2001"
## [239] "unesco 2006"
## [240] "unesco 2014"
## [241] "unesco 2017"
## [242] "united nations 1998"
## [243] "united nations 2003"
## [244] "united nations 2015"
## [245] "united nations 2021"
## [246] "united nations 2022"
## [247] "united nations 2023"
## [248] "velez sanchez et al 2023"
## [249] "vorosmarty et al 2000"
## [250] "vorosmarty et al 2005"
## [251] "vorosmarty et al 2010"
## [252] "wada 2015"
## [253] "wada et al 2013b"
## [254] "wada et al 2014"
## [255] "wada et al 2016"
## [256] "wajima 2018"
## [257] "walter et al 2017"
## [258] "wbcsd  2009"
```

```
## [259] "williams et al 2017"
## [260] "wisser et al 2008"
## [261] "wisser et al 2010"
## [262] "wmo 1997"
## [263] "world bank 2001"
## [264] "world bank 2017"
## [265] "world bank 2021"
## [266] "world water assessment programme 2003"
## [267] "world water assessment programme 2014"
## [268] "worldometers 2019"
## [269] "wri 2000"
## [270] "wu et al 2022"
## [271] "wwap 2018"
## [272] "wwf 2006"
## [273] "xing yuan et al 2024"
## [274] "xu et al 2020"
## [275] "yilmazkuday et al 2021"
## [276] "yin et al 2022"
## [277] "young et al 2019"
## [278] "zhao et al 2022"
## [279] "zhuo et al 2022"
```

```
# Calculate proportions ------------------------------------------------------

lapply(out, function(x) length(x) / length(all_nodes))
```

```
## $`path ending in no claim`
## [1] 0.2917647
##
## $`path ending in no claim or no citation`
## [1] 0.6564706
```

5 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] scales_1.3.0       wesanderson_0.3.6  benchmarkme_1.0.8  tidygraph_1.3.0
##  [5] cowplot_1.1.1      ggraph_2.1.0       igraph_1.6.0       bibliometrix_4.0.1
##  [9] lubridate_1.9.2    forcats_1.0.0      stringr_1.5.1      dplyr_1.1.4
## [13] purrr_1.0.2        readr_2.1.4        tidyr_1.3.0        tibble_3.2.1
## [17] ggplot2_3.4.4      tidyverse_2.0.0    data.table_1.14.99 openxlsx_4.2.5.2
##
## loaded via a namespace (and not attached):
##   [1] Rdpack_2.6             gridExtra_2.3          readxl_1.4.2
##   [4] rlang_1.1.3            magrittr_2.0.3         tidytext_0.4.1
##   [7] compiler_4.3.3         vctrs_0.6.5            crayon_1.5.2
##  [10] pkgconfig_2.0.3        fastmap_1.1.1          ellipsis_0.3.2
##  [13] labeling_0.4.3         utf8_1.2.4             promises_1.2.0.1
##  [16] rmarkdown_2.21         tzdb_0.3.0             tinytex_0.45
##  [19] bit_4.0.5              xfun_0.39              jsonlite_1.8.4
##  [22] flashClust_1.01-2      highr_0.10             SnowballC_0.7.1
##  [25] later_1.3.0            tweenr_2.0.2           cluster_2.1.6
##  [28] R6_2.5.1               stringi_1.8.3          RColorBrewer_1.1-3
##  [31] cellranger_1.1.0       estimability_1.4.1     iterators_1.0.14
##  [34] Rcpp_1.0.12            knitr_1.42             filehash_2.4-5
##  [37] httpuv_1.6.9           rentrez_1.2.3          Matrix_1.6-5
##  [40] timechange_0.2.0       tidyselect_1.2.0       viridis_0.6.4
##  [43] rstudioapi_0.15.0      stringdist_0.9.10      pubmedR_0.0.3
##  [46] yaml_2.3.7             codetools_0.2-19       doParallel_1.0.17
```

```
##  [49] lattice_0.22-5         plyr_1.8.8              shiny_1.7.4
##  [52] withr_3.0.0            benchmarkmeData_1.0.4   coda_0.19-4
##  [55] evaluate_0.20          polyclip_1.10-6         zip_2.3.0
##  [58] pillar_1.9.0           janeaustenr_1.0.0       foreach_1.5.2
##  [61] DT_0.27                plotly_4.10.1           generics_0.1.3
##  [64] vroom_1.6.1            hms_1.1.3               munsell_0.5.0
##  [67] sensobol_1.1.4         xtable_1.8-4            leaps_3.1
##  [70] glue_1.7.0             tikzDevice_0.12.4       emmeans_1.8.5
##  [73] scatterplot3d_0.3-43   lazyeval_0.2.2          tools_4.3.3
##  [76] tokenizers_0.3.0       mvtnorm_1.1-3           graphlayouts_1.0.2
##  [79] XML_3.99-0.14          grid_4.3.3              rbibutils_2.2.16
##  [82] rscopus_0.6.6          colorspace_2.1-0        dimensionsR_0.0.3
##  [85] ggforce_0.4.1          bibliometrixData_0.3.0  cli_3.6.2
##  [88] fansi_1.0.6            viridisLite_0.4.2       gtable_0.3.4
##  [91] digest_0.6.34          ggrepel_0.9.5           FactoMineR_2.8
##  [94] htmlwidgets_1.6.2      farver_2.1.1            htmltools_0.5.5
##  [97] factoextra_1.0.7       lifecycle_1.0.4         httr_1.4.5
## [100] multcompView_0.1-9     mime_0.12               bit64_4.0.5
## [103] 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
```