Irrigation's real impact on global water and food security

R code

Arnald Puy

Contents

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

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

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

```r
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
```

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

```
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

```
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
```

```
# 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

```
# 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
```

```
# 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

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

tmp <- list()
names.files <- c("WORK", "NETWORK")
topics <- c("water", "food")
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

sub(".*\\.([^\\.]+)_.*", "\\1", all.files)
```

```
##  [1] "water" "water" "water" "food"  "food"  "food"  "water" "water" "water"
## [10] "food"  "food"  "food"
```

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

# Work datasets -----------------------------------------------------------------

dataset.works <- all.files[str_detect(all.files, "_WORK")]
dataset.works.topics <- sub(".*\\.([^\\.]+)_.*", "\\1", dataset.works)

tmp.works <- tmp[dataset.works]
names(tmp.works) <- dataset.works.topics
lapply(tmp.works, function(dt) dt[, .(doi, title, claim.in.text)]) %>%
  rbindlist(., idcol = "topic") %>%
  .[, .N, .(topic, claim.in.text)]
```

```
##      topic claim.in.text     N
##     <char>        <char> <int>
##  1:  water             F   715
##  2:  water          <NA>   402
##  3:  water             T   471
##  4:  water     Paywalled     9
##  5:  water       Russian     1
##  6:  water        French     1
##  7:  water        Indian     1
##  8:  water      Ukranian     1
##  9:  water    Portuguese     1
## 10:   food          <NA>    82
## 11:   food             T    33
## 12:   food             F   524
```

```
# Network datasets --------------------------------------------------------------

dataset.networks <- all.files[str_detect(all.files, "NETWORK")]
dataset.networks.topics <- sub(".*\\.([^\\.]+)_.*", "\\1", dataset.networks)

tmp2 <- tmp[dataset.networks]
names(tmp2) <- dataset.networks.topics

network.dt <- rbindlist(tmp2, idcol = "topic") %>%
  .[, policy:= grepl("^policy", doi)]

# Retrieve year -----------------------------------------------------------------

network.dt[, year:= as.integer(sub(".* (\\d{4})[a-z]?$", "\\1", author))]
```

```
## Warning in eval(jsub, SDenv, parent.frame()): NAs introduced by coercion
```

```
# move policy to author ---------------------------------------------------------
```

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

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

aquastat.cites <- network.dt[citation %like% "fao aquastat"] %>%
  .[, .N, .(citation, topic)]

aquastat.cites
```

```
##              citation  topic     N
##                <char> <char> <int>
##  1: fao aquastat 2006  water     1
##  2:      fao aquastat  water    12
##  3: fao aquastat 2010  water     5
##  4: fao aquastat 2020  water     3
##  5: fao aquastat 2011  water     2
##  6: fao aquastat 2012  water     4
##  7: fao aquastat 2021  water     2
##  8: fao aquastat 2017  water     1
##  9: fao aquastat 2015  water     2
## 10: fao aquastat 2019  water     3
## 11: fao aquastat 2016  water     6
## 12: fao aquastat 2014  water     1
## 13: fao aquastat 2023  water     1
## 14: fao aquastat 2018  water     1
## 15: fao aquastat 2004  water     1
## 16: fao aquastat 2005  water     1
## 17:      fao aquastat   food     5
```

```
oldest.aquastat.cite <- min(as.integer(sub(".* (\\d{4})[a-z]?$", "\\1",
                                           aquastat.cites$citation)),
    na.rm = TRUE)
```

```
## Warning: NAs introduced by coercion
```

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

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

lookup.dt[, .(number.rows = nrow(.SD)), topic]
```

```
##     topic number.rows
##    <char>       <int>
## 1:  water         779
## 2:   food          48
```

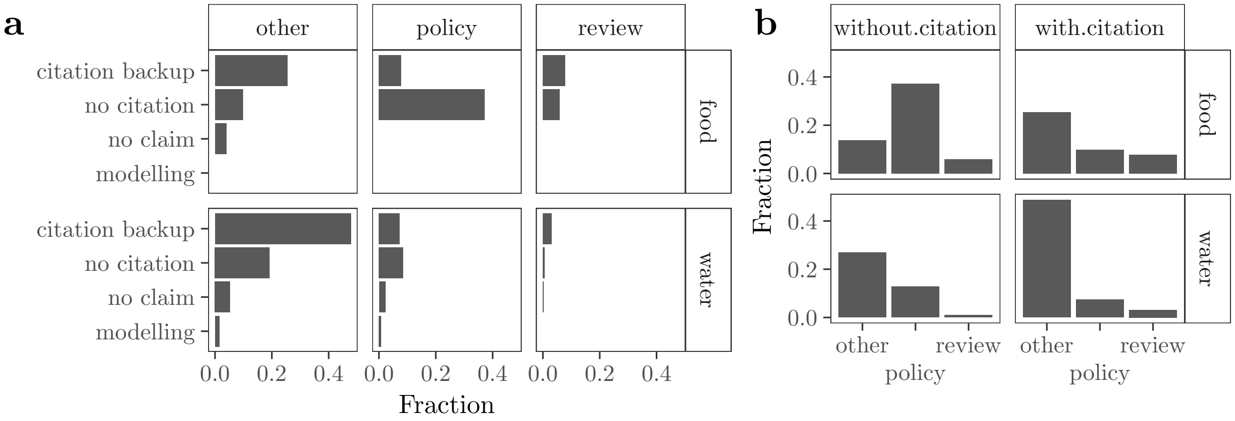
```r
# Export lookup tables ---------------------------------------------------------

write.xlsx(lookup.dt, "lookup.dt.xlsx")
write.xlsx(lookup.dt[topic == "water"], "lookup.water.dt.xlsx")
write.xlsx(lookup.dt[topic == "food"], "lookup.food.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"))

# Rename category --------------------------------------------------------------

network.dt[, category:= ifelse(!classification == "F", "Uncertain", "Fact")]

# 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, year, document.type, nature.claim,
                             classification, category, topic)]
cols_to_change <- colnames(network.dt)
network.dt[, (cols_to_change):= lapply(.SD, trimws), .SDcols = (cols_to_change)]
```

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

total.rows <- network.dt[, .(number.rows = nrow(.SD)), topic]

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

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

```
## Key: <topic>
##     topic    nature.claim     N number.rows   fraction
##    <char>          <char> <int>       <int>      <num>
## 1:   food     no citation    22          51 0.43137255
## 2:   food citation backup    19          51 0.37254902
## 3:   food        no claim     2          51 0.03921569
## 4:   food            <NA>     1          51 0.01960784
## 5:  water citation backup   534         921 0.57980456
## 6:  water       modelling    21         921 0.02280130
## 7:  water     no citation   257         921 0.27904452
## 8:  water            <NA>    29         921 0.03148751
## 9:  water        no claim    72         921 0.07817590
```

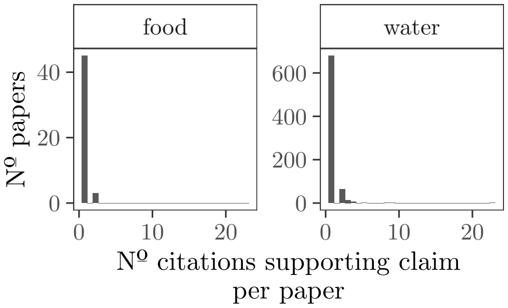
```
# Count document type by nature of claim ---------------------------------------

a <- network.dt[, .N, .(nature.claim, document.type, topic)] %>%
  merge(., total.rows, by = "topic") %>%
  .[, proportion:= N / number.rows] %>%
  na.omit() %>%
  ggplot(., aes(reorder(nature.claim, proportion), proportion)) +
  coord_flip() +
  geom_bar(stat = "identity") +
  facet_grid(topic~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, topic)] %>%
  melt(., measure.vars = c("without.citation", "with.citation")) %>%
  merge(., total.rows, by = "topic") %>%
  .[, proportion:= value / number.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_grid(topic~variable) +
  theme_AP()

# merge ------------------------------------------------------------------------
```

```
plot_grid(a, b, ncol = 2, rel_widths = c(0.6, 0.4), labels = "auto")
```



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

network.dt[, .N, .(from, topic)] %>%
  .[order(-N)] %>%
  ggplot(., aes(N)) +
  geom_histogram() +
  facet_wrap(~topic, scale = "free_y") +
  scale_x_continuous(breaks = breaks_pretty(n = 3)) +
  scale_y_continuous(breaks = breaks_pretty(n = 3)) +
  theme_AP() +
  labs(x = "Nº citations supporting claim \n per paper", y = "Nº papers")
```

```
## `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$to), ]
split.networks <- split(network.dt.complete, network.dt.complete$topic)

# Export-----
```

```r
write.xlsx(network.dt.complete, "network.dt.complete.xlsx")

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

citation_graph <- lapply(split.networks, function(dt)
  graph_from_data_frame(d = dt, directed = TRUE))
```

```
## Warning in graph_from_data_frame(d = dt, directed = TRUE): In `d' `NA' elements
## were replaced with string "NA"
```

```r
# Calculate network metrics ---------------------------------------------------

lapply(citation_graph, function(x) edge_density(x))
```

```
## $food
## [1] 0.02083333
##
## $water
## [1] 0.001534518
```

```r
# 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 <- lapply(citation_graph, function(x) cluster_walktrap(x))
lapply(wtc, function(x) modularity(x))
```

```
## $food
## [1] 0.7944215
##
## $water
## [1] 0.8471048
```

```r
network_metrics <- lapply(citation_graph, function(x)
  data.table(node = V(x)$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(x, mode = "in"),

             degree.out = degree(x, 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(x),

             # 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(x),
             pagerank = page_rank(x)$vector)
)

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

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

degree.nodes
```

```
## $food
##                 node degree degree.out betweenness closeness   pagerank
##               <char>  <num>      <num>       <num>     <num>      <num>
## 1:     fao aquastat      5          0           0       NaN 0.09168305
## 2: meier et al 2018      1          1           1 1.0000000 0.02743993
## 3:   wang et al 2012      1          1           2 0.3333333 0.03562376
##
## $water
##                   node degree degree.out betweenness closeness    pagerank
##                 <char>  <num>      <num>       <num>     <num>       <num>
## 1:       fao aquastat     46          0     0.00000       NaN 0.063790805
## 2: siebert et al 2010     12          3    37.33333 0.3333333 0.008580413
## 3:  molden et al 2007     10          1    22.00000 0.3333333 0.009077773
```

```
degree.nodes.out
```

```
## $food
##                   node degree degree.out betweenness closeness   pagerank
##                 <char>  <num>      <num>       <num>     <num>      <num>
## 1:     niu et al 2023      0          2           0      0.25 0.01925609
## 2: borsato et al 2020      0          2           0      0.50 0.01925609
## 3:         borin 2023      0          2           0      0.25 0.01925609
##
## $water
##                node degree degree.out betweenness  closeness     pagerank
##             <char>  <num>      <num>       <num>      <num>        <num>
## 1:       wada 2015      0          23           0 0.02702703 0.0009265398
## 2: wada et al 2014      1           9          10 0.09090909 0.0010840515
## 3: wada et al 2016      0           8           0 0.06250000 0.0009265398
```

```
betweenness.nodes
```

```
## $food
##                       node degree degree.out betweenness closeness   pagerank
##                     <char>  <num>      <num>       <num>     <num>      <num>
## 1:         wang et al 2012      1          1           2 0.3333333 0.03562376
## 2: hanjra and qureshi 2010      1          1           2 1.0000000 0.04953629
## 3:        meier et al 2018      1          1           1 1.0000000 0.02743993
##
## $water
##                     node degree degree.out betweenness  closeness    pagerank
##                   <char>  <num>      <num>       <num>      <num>       <num>
## 1: boretti and rosa 2019      4          4    44.00000 0.05555556 0.003682996
## 2:    siebert et al 2010     12          3    37.33333 0.33333333 0.008580413
## 3:     molden et al 2007     10          1    22.00000 0.33333333 0.009077773
```

```
pagerank.nodes
```

```
## $food
##                     node degree degree.out betweenness closeness   pagerank
##                   <char>  <num>      <num>       <num>     <num>      <num>
## 1: okorogbona et.al 2018      0          1           0         1 0.01925609
## 2:   du preez et al 2018      0          1           0         1 0.01925609
## 3:      meier et al 2018      1          1           1         1 0.02743993
##
## $water
##                     node degree degree.out betweenness closeness     pagerank
##                   <char>  <num>      <num>       <num>     <num>        <num>
## 1: sharma and irmak 2012      0          1           0         1 0.0009265398
## 2:       world bank 2007      3          1          11         1 0.0076566564
## 3:   brajovic et al 2015      0          1           0         1 0.0009265398
```

3.2 Network plots

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

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

graph <- lapply(split.networks, function(nt)
  tidygraph::as_tbl_graph(nt, directed = TRUE))
```

```
## Warning in graph_from_data_frame(x, directed = directed): In `d' `NA' elements
## were replaced with string "NA"
```

```
vec.names <- lapply(graph, function(graph)
  graph %>%
    activate(nodes) %>%
    pull() %>%
    data.table(name = .))


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

tmp.network <- split(network.dt, network.dt$topic)

vec.nature.claim <- list()

for(i in names(tmp.network)) {

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

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

order_indices <- final.vec.nature.claim <- final.vec.document.type <-
  final.vec.year <- final.vec.classification <- final.vec.category <- list()

for (i in names(vec.names)) {

  order_indices[[i]] <- match(vec.names[[i]]$name, vec.nature.claim[[i]]$name)
  final.vec.nature.claim[[i]] <- vec.nature.claim[[i]][order_indices[[i]], ] %>%
    .[, nature.claim]
  final.vec.document.type[[i]] <- vec.nature.claim[[i]][order_indices[[i]], ] %>%
    .[, document.type]
  final.vec.year[[i]] <- vec.nature.claim[[i]][order_indices[[i]], ] %>%
    .[, year] %>%
    as.numeric()
  final.vec.classification[[i]] <- vec.nature.claim[[i]][order_indices[[i]], ] %>%
    .[, classification]
  final.vec.category[[i]] <- vec.nature.claim[[i]][order_indices[[i]], ] %>%
    .[, category]
}

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

graph.final <- list()

for (i in names(graph)) {
  graph.final[[i]] <- graph[[i]] %>%
    activate(nodes) %>%
    mutate(nature.claim = final.vec.nature.claim[[i]],
           document.type = final.vec.document.type[[i]],
           year = final.vec.year[[i]],
           degree = network_metrics[[i]]$degree,
           classification = final.vec.classification[[i]],
           category = final.vec.category[[i]],
           degree.out = network_metrics[[i]]$degree.out,
           betweenness = network_metrics[[i]]$betweenness,
           pagerank = network_metrics[[i]]$pagerank)


}
```

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

lapply(graph.final, function(graph) V(graph))
```

```
## $food
## + 33/33 vertices, named, from c33b012:
##  [1] okorogbona et.al 2018          du preez et al 2018
##  [3] niu et al 2023                  meier et al 2018
##  [5] lobell et al 2006               rosa 2022
##  [7] rolle et al 2021                mitchell et al 2018
##  [9] wang et al 2012                 hanjra and qureshi 2010
## [11] de pascale et al 2011           borsato et al 2020
## [13] borin 2023                      turral et al 2010
## [15] meier et al 2017                policy.1666264
## [17] policy.1869122                  policy.1898497
## [19] policy.1667934                  fernandez-cirelli et al 2009
## + ... omitted several vertices
##
## $water
## + 597/597 vertices, named, from 43d1a43:
##   [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 ##############################################################

lapply(graph.final, function(graph) ecount(graph))
```

```
## $food
## [1] 22
##
## $water
## [1] 546
```

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

lapply(graph.final, function(graph) {

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

  })
```

```
## $food
## [1] 0.01136364
##
## $water
## [1] 0.0007789694
```

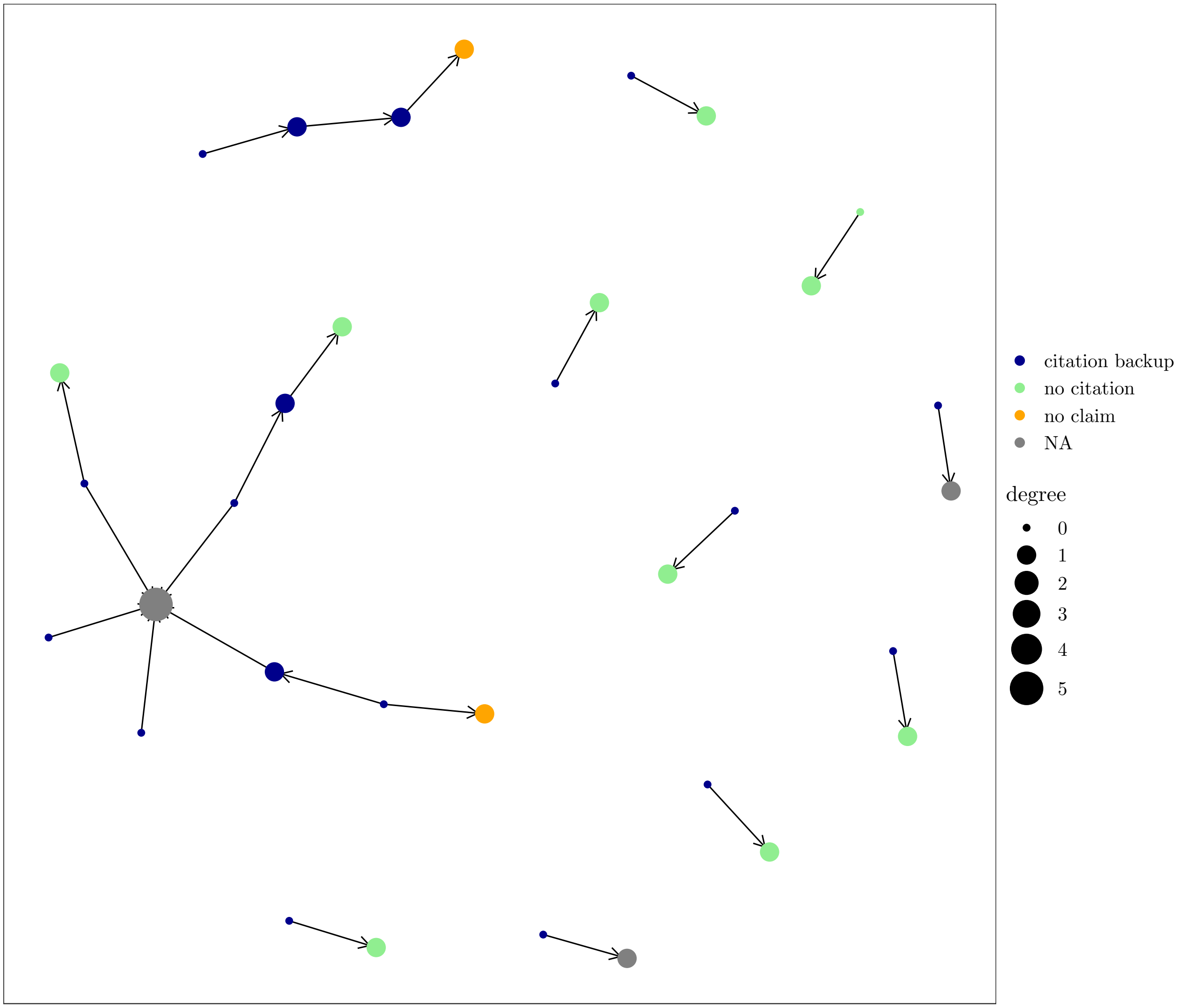
```r
# PROPORTION OF LINKS CONNECTED TO THE 5 NODES WITH HIGHEST DEGREE ############

lapply(graph.final, function(graph) {

  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

})
```

```
## $food
## [1] 0.5909091
##
## $water
## [1] 0.1941392
```

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

seed <- 123
selected_colors <- c("darkblue", "lightgreen", "orange", "red", "grey")
```

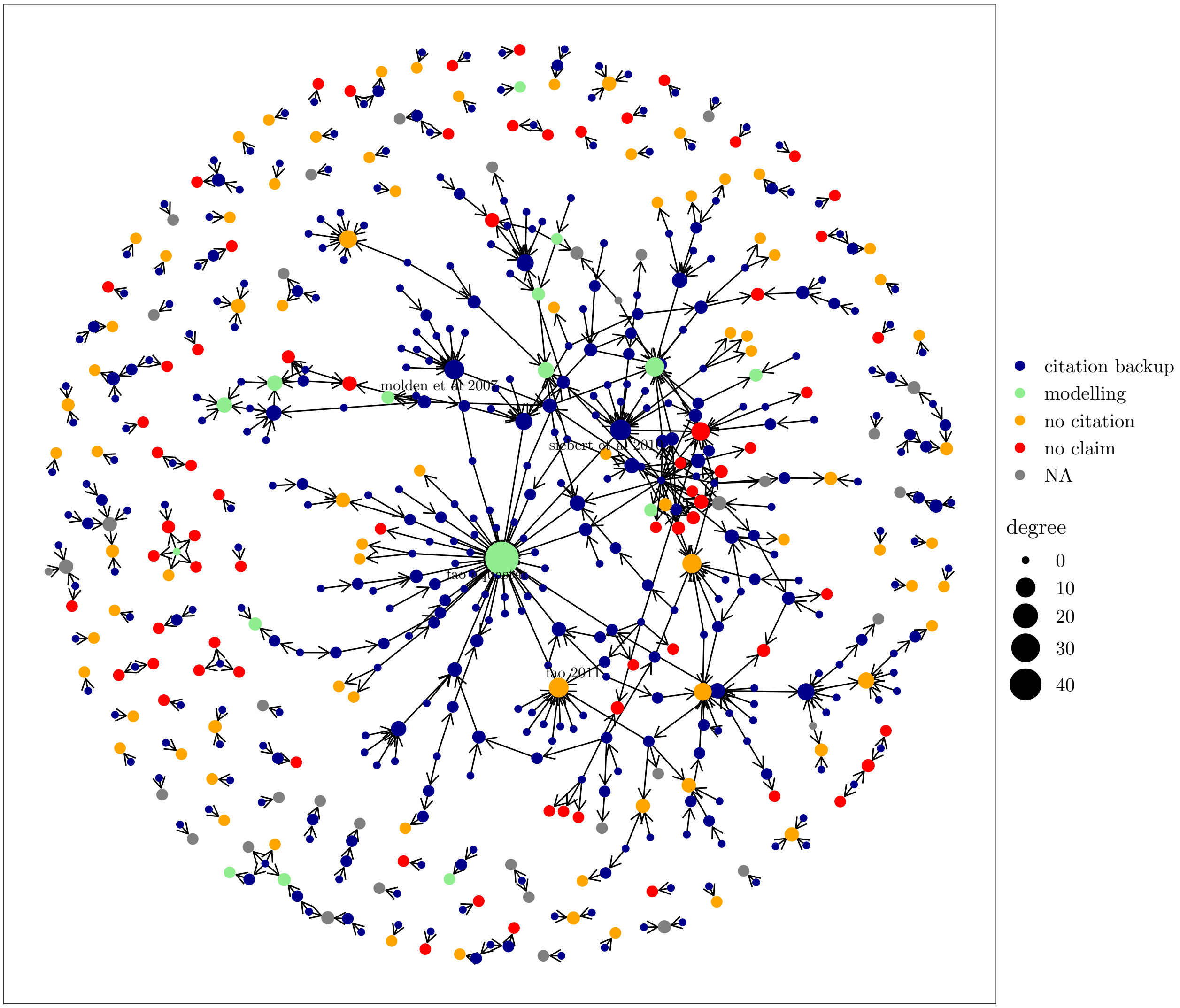
```r
# by nature of claim -----------------------------------------------------------

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

p1 <- p2 <- p3 <- p4 <- list()

for(i in names(graph.final)) {

  set.seed(seed)

  p1[[i]] <- ggraph(graph.final[[i]], 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[[i]]$degree), name, NA)),
                   repel = TRUE, size = 2.2) +
    labs(x = "", y = "") +
    scale_color_manual(name = "",
                       values = selected_colors) +
    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")
}

p1
```

```
## $food

## 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 33 rows containing missing values (`geom_text_repel()`).
```



```
##
## $water
```

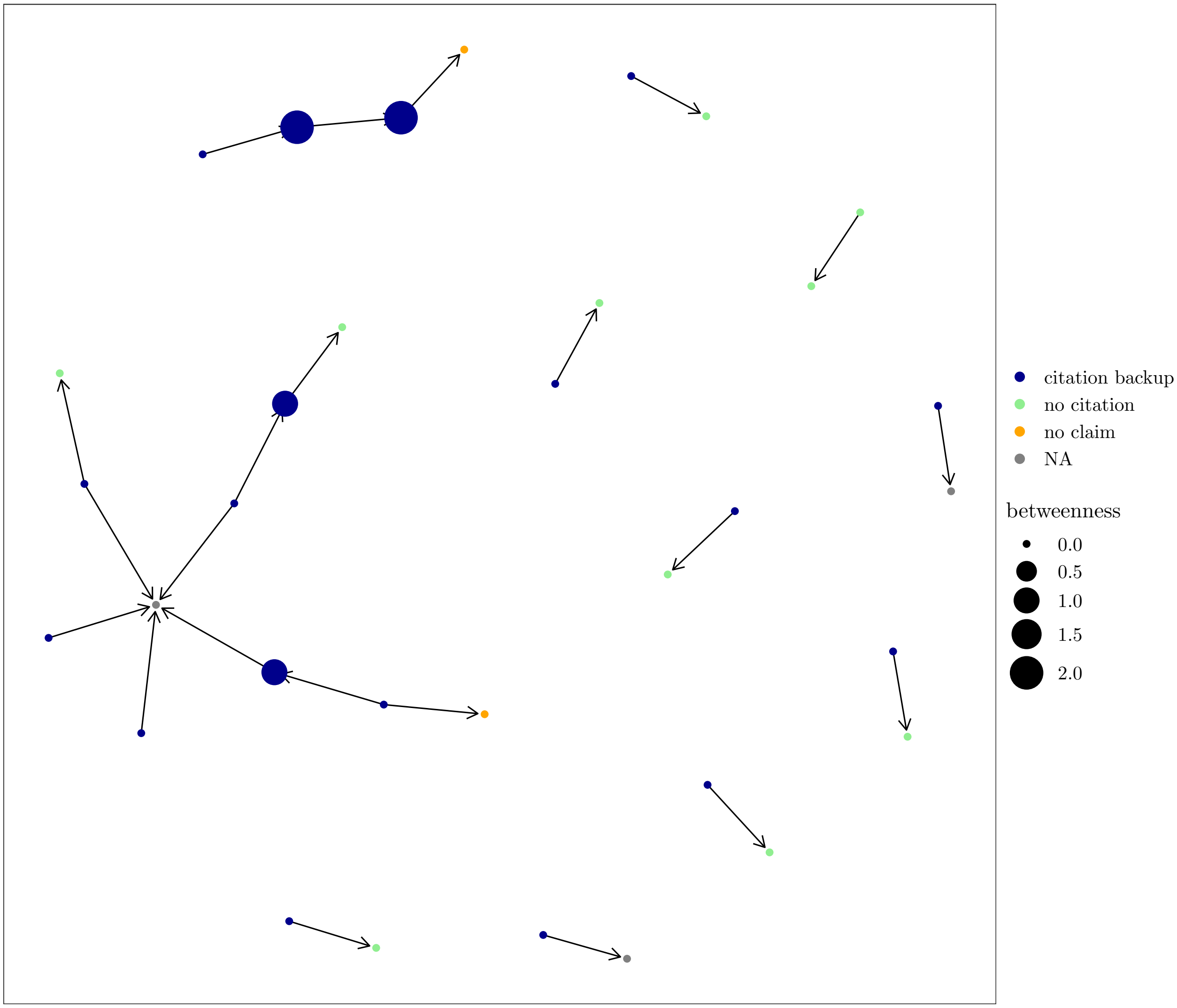
```
## Warning: Removed 593 rows containing missing values (`geom_text_repel()`).
```



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

for (i in names(graph.final)) {

  set.seed(seed)

  p2[[i]] <- ggraph(graph.final[[i]], 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[[i]]$betweenness),
                                      name, NA)),
                   repel = TRUE, size = 2.2) +
    labs(x = "", y = "") +
    scale_color_manual(name = "",
                       values = selected_colors) +
    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")
}

p2
```

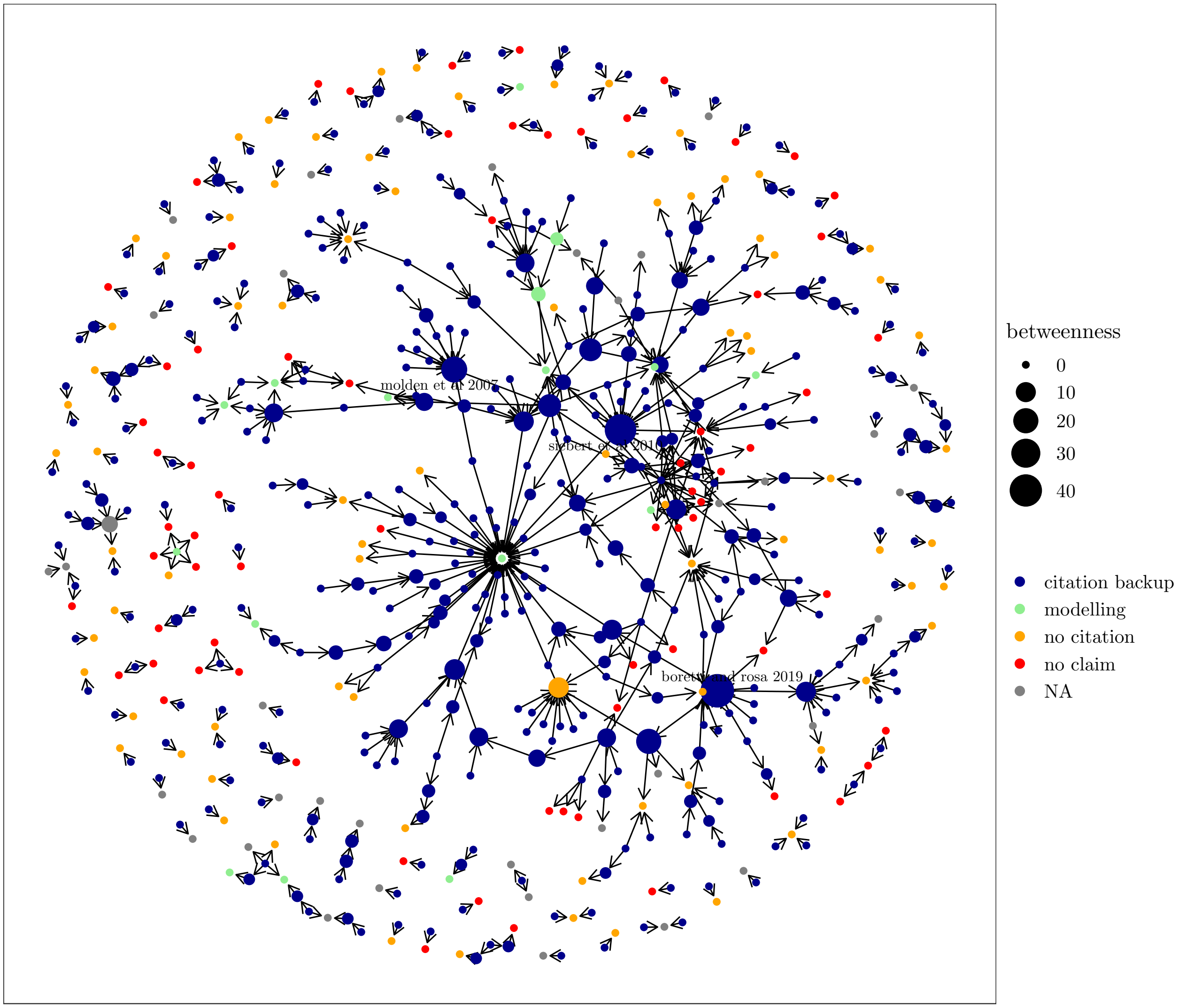
```
## $food
```

```
## Warning: Removed 33 rows containing missing values (`geom_text_repel()`).
```



```
##
## $water
```

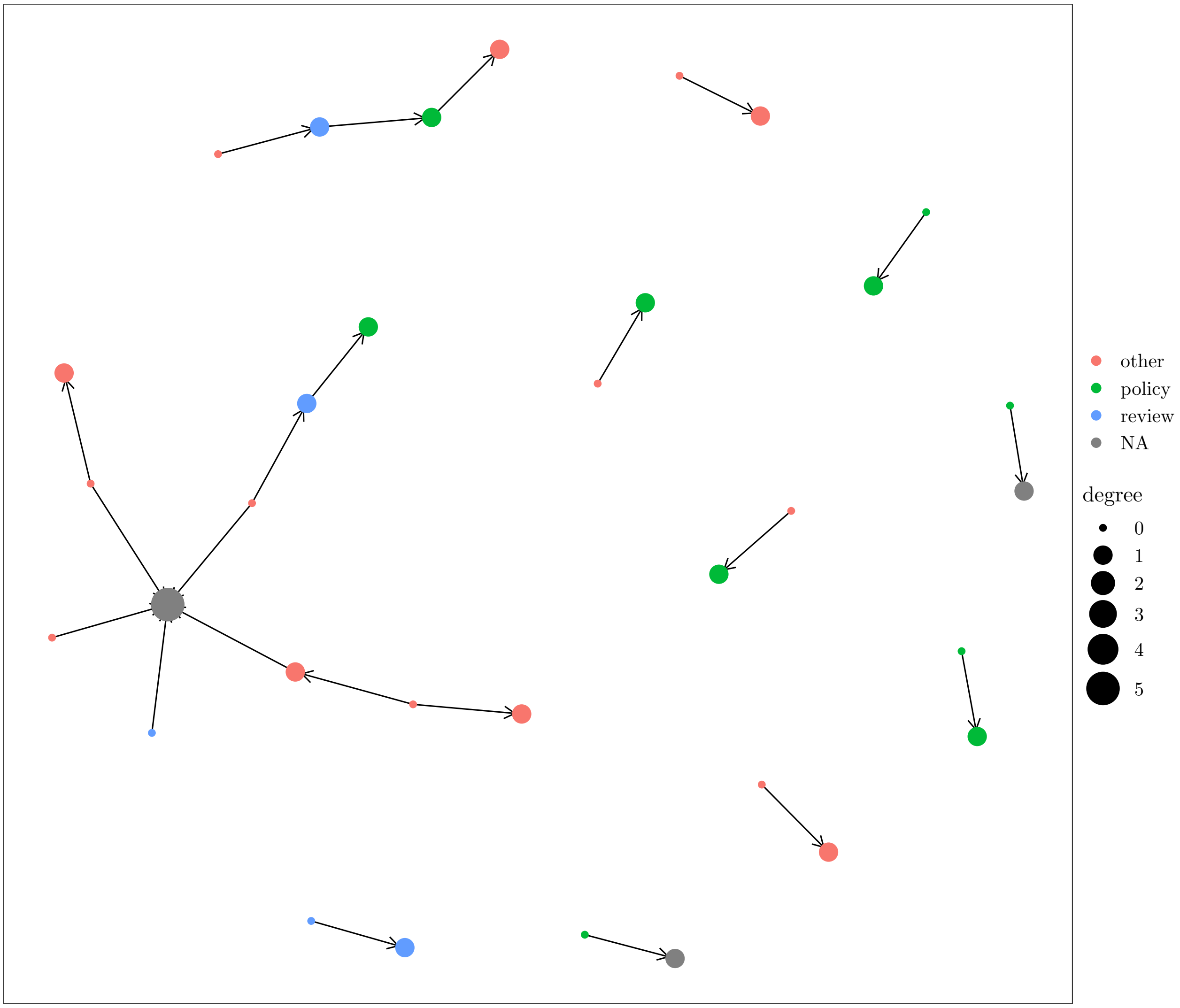
```
## Warning: Removed 594 rows containing missing values (`geom_text_repel()`).
```



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

for (i in names(graph.final)) {

  set.seed(seed)

  p3[[i]] <- ggraph(graph.final[[i]], 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[[i]]$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")
}

p3
```

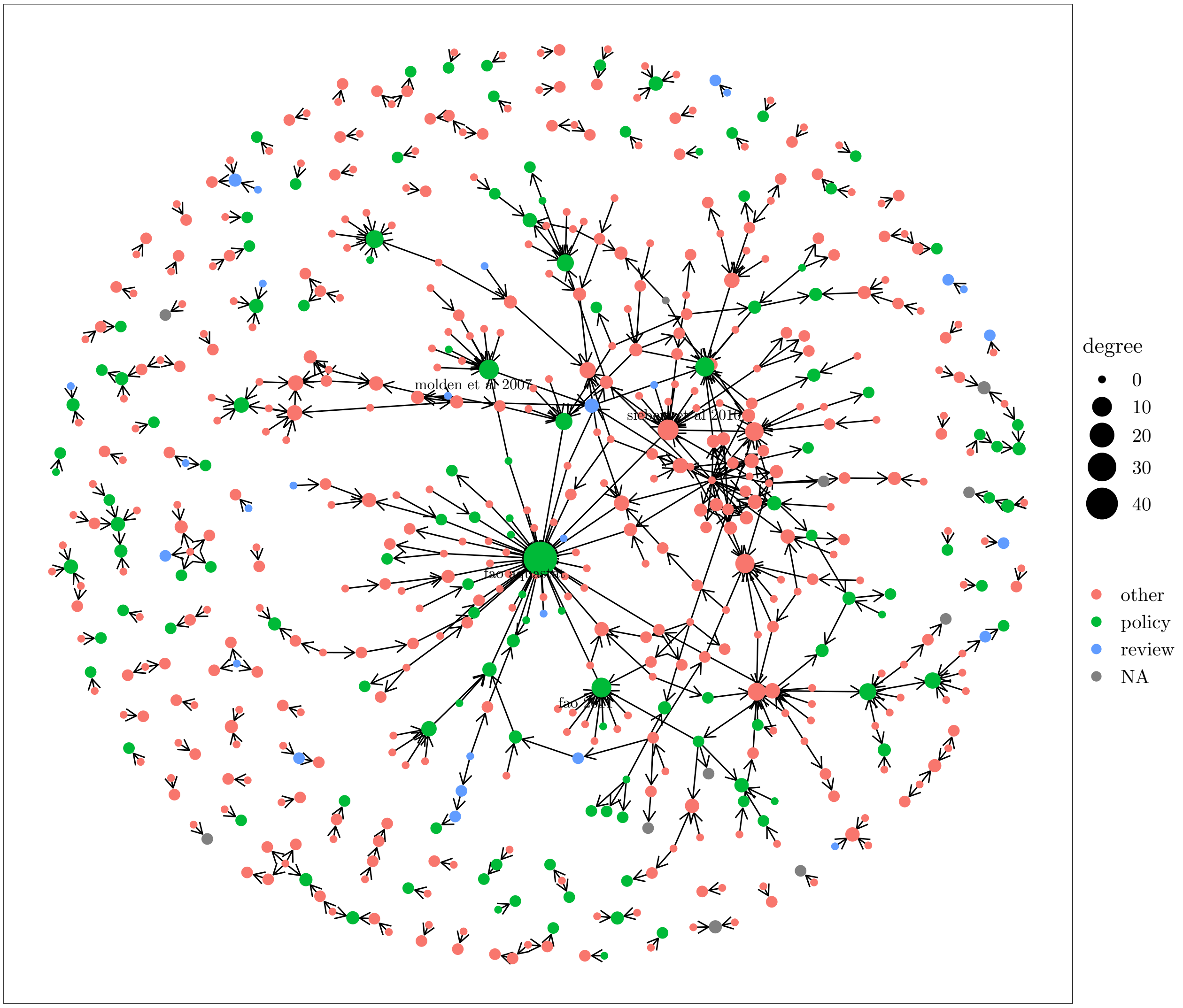
```
## $food
```

```
## Warning: Removed 33 rows containing missing values (`geom_text_repel()`).
```



```
##
## $water
```

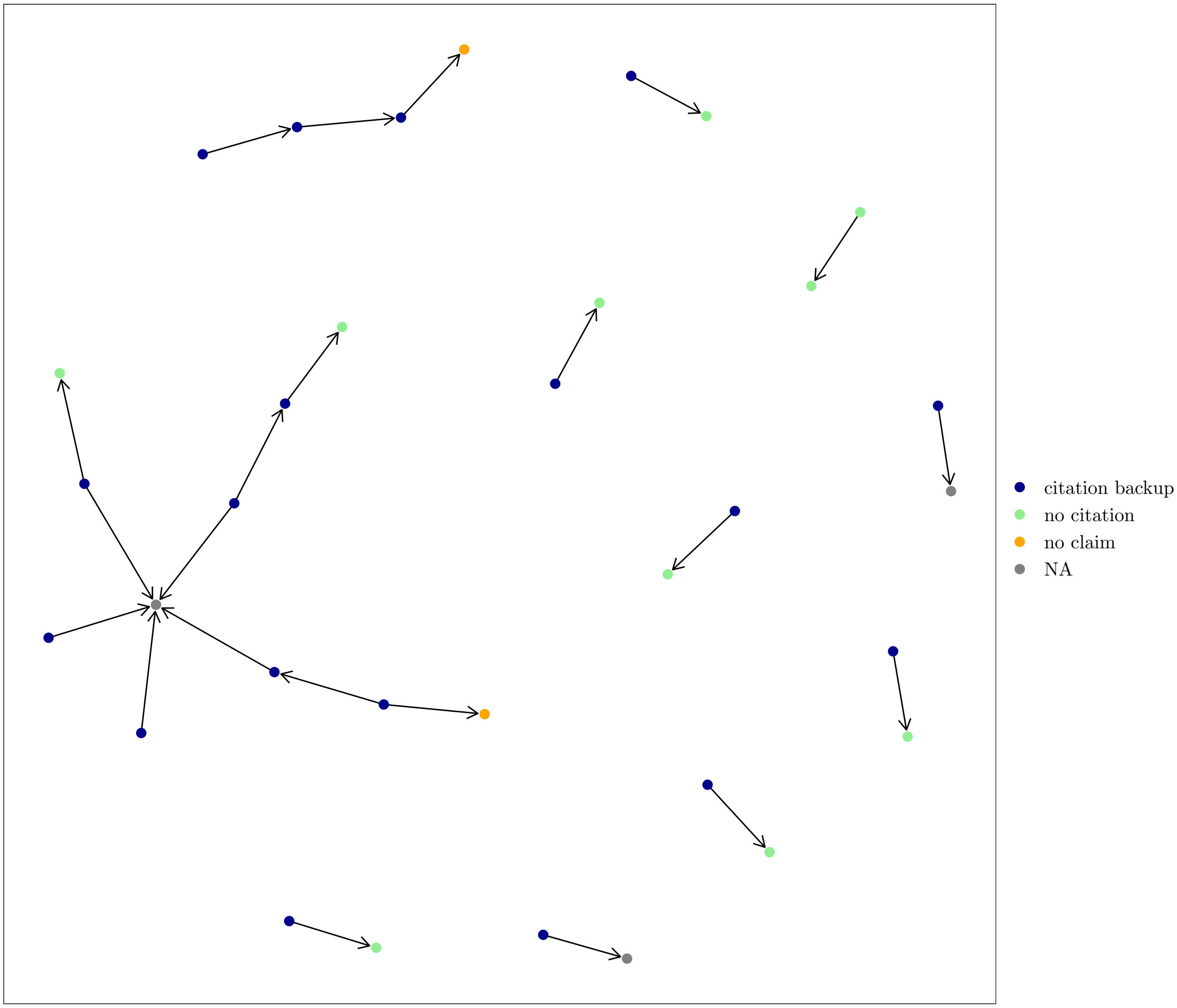
```
## Warning: Removed 593 rows containing missing values (`geom_text_repel()`).
```



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

for (i in names(graph.final)) {

  set.seed(seed)

  p4[[i]] <- ggraph(graph.final[[i]], 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 = selected_colors) +
    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")
}

p4
```

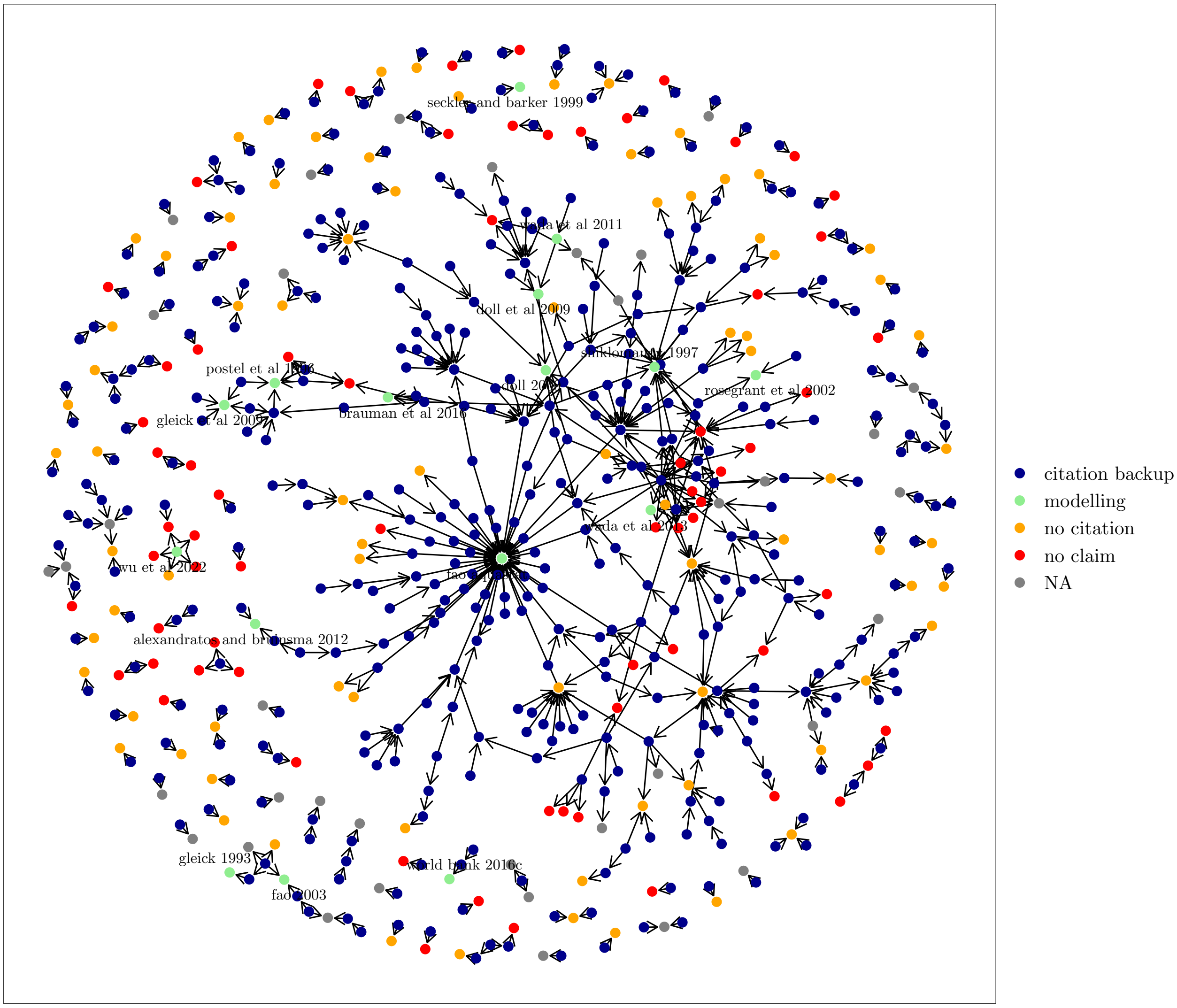
```
## $food
```

```
## Warning: Removed 33 rows containing missing values (`geom_text_repel()`).
```



```
##
## $water
```

```
## Warning: Removed 581 rows containing missing values (`geom_text_repel()`).
```



3.3 Uncertainties turned into facts

```
# COUNT PROPORTION OF NODES THAT STATE AS FACT A CLAIM UTTERED AS UNCERTAIN ####

uncertainty_plot_fun <- function(graph) {

# Extract name of all studies ----------------------------------------------------

all.names <- graph %>%
    activate(nodes) %>%
    pull(name)

# Extract name of studies stating claim as fact --------------------------------

  f.names <- graph %>%
    activate(nodes) %>%
    data.frame() %>%
    filter(classification == "F") %>%
    pull(name)

  # Add names to edges -----------------------------------------------------------

  add.names.edges <- graph %>%
    activate(edges) %>%
    mutate(from.name = all.names[from],
           to.name = all.names[to])

  # Calculate, for each study stating claim as fact, the studies it cites --------
  out.classes <- lapply(f.names, function(x) {

    out_nodes <- add.names.edges %>%
      activate(edges) %>%
      filter(from.name == x) %>%
      pull(to.name)

  })

  # unlist names of studies cited by studies uttering claim as fact --------------

  di <- sort(unlist(out.classes))

  # Extract only those that do not state claim as fact ---------------------------

  nodes.no.fact <- graph %>%
    activate(nodes) %>%
    data.frame() %>%
    data.table() %>%
    .[name %in% di] %>%
    .[!classification == "F"] %>%
    .$name

  name.edges <- add.names.edges  %>%
    activate(edges) %>%
    data.frame() %>%
    filter(from.name %in% f.names & to.name %in% nodes.no.fact) %>%
    .[, c("from.name", "to.name")] %>%
    c() %>%
    unlist() %>%
    unique(.)

  output <- add.names.edges  %>%
    activate(nodes) %>%
    filter(name %in% name.edges) %>%
    activate(edges) %>%
```

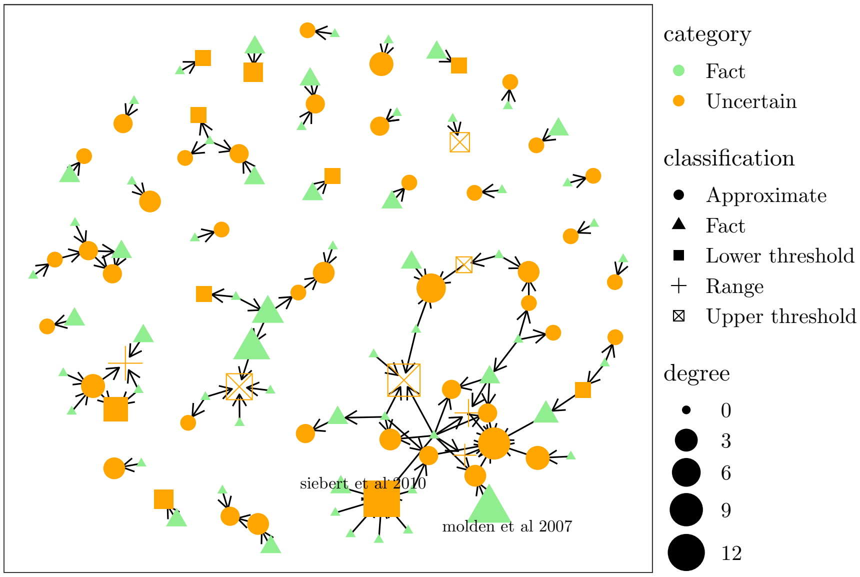
```r
    filter(from.name %in% name.edges & to.name %in% name.edges)

  return(output)

}
```

```r
# PLOT GRAPH UNCERTAINTIES TURNED INTO FACTS ##################################

out <- lapply(graph.final, function(x) uncertainty_plot_fun(x))

p7 <- list()

for (i in names(out)) {

  set.seed(seed)

  p7[[i]] <- ggraph(out[[i]], layout = "igraph", algorithm = "nicely") +
    geom_edge_link(arrow = arrow(length = unit(1.8, 'mm')),
                   end_cap = circle(1, "mm")) +
    geom_node_point(aes(color = category, size = degree, shape = classification)) +
    scale_color_manual(values = c("lightgreen", "orange")) +
    scale_shape_discrete(labels = c("Approximate", "Fact", "Lower threshold", "Range", "Upper t
    geom_node_text(aes(label = ifelse(degree >= min(degree.nodes[[i]]$degree), name, NA)),
                   repel = TRUE, size = 2.2) +
    labs(x = "", y = "") +
    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")
}

p7
```

```
## $food
```

```
##
## $water
```

```
## Warning: Removed 117 rows containing missing values (`geom_text_repel()`).
```



```
# FUNCTION TO CALCULATE ALL PATHS BETWEEN PAIRS OF NODES #######################

calculate_paths <- function(graph) {

  # Convert to igraph -----------------------------------------------------------

  igraph_graph <- as.igraph(graph)

  # Get all unique pairs of nodes ----------------------------------------------
  node_pairs <- expand.grid(from = V(igraph_graph), to = V(igraph_graph))
  node_pairs <- node_pairs[node_pairs$from != node_pairs$to, ]

  # Function to calculate all simple paths between a pair of nodes--------------

  calculate_paths <- function(from, to) {
    paths <- all_simple_paths(igraph_graph, from = from, to = to)
    lapply(paths, names)
  }

  # Apply the function to all node pairs and unnest the results-----------------

  all_paths <- node_pairs %>%
    rowwise() %>%
    mutate(paths = list(calculate_paths(from, to))) %>%
    unnest(cols = c(paths))

  out <- sum(sapply(all_paths$paths, function(x) length(x)))

  return(out)

}

# CALCULATE ALL PATHS / PATHS TURNING HYPOTHESIS INTO FACTS ####################

all.paths <- hypothesis.into.facts.paths <- list()

for (i in names(graph.final)) {

  all.paths[[i]] <- calculate_paths(graph.final[[2]])
  hypothesis.into.facts.paths[[i]] <- uncertainty_plot_fun(graph.final[[2]]) %>%
    calculate_paths(.)

}

# Print results: proportion of paths turning uncertainties into facts ------------------------

for (i in names(all.paths)) {
  print(hypothesis.into.facts.paths[[i]] / all.paths[[i]])
}
```

```
## [1] 0.1432161
## [1] 0.1432161
```

3.4 Network through time

```r
# PREPARE DATA TO PLOT NETWORK THROUGH TIME ####################################

# Extract vector with names ----------------------------------------------------

location_aquastat <- graph.final[[2]] %>%
  activate(nodes) %>%
  data.frame() %>%
  pull(name) %>%
  grep("aquastat", .)

# Extract vector with years ----------------------------------------------------

v_years <- graph.final[[2]] %>%
  activate(nodes) %>%
  data.frame() %>%
  pull(year)

# Substitute fao aquastat without year with the oldest aqustat citation --------

v_years[location_aquastat] <- oldest.aquastat.cite

# Find NA values ---------------------------------------------------------------

na_indices <- is.na(v_years)
sum(na_indices)
```

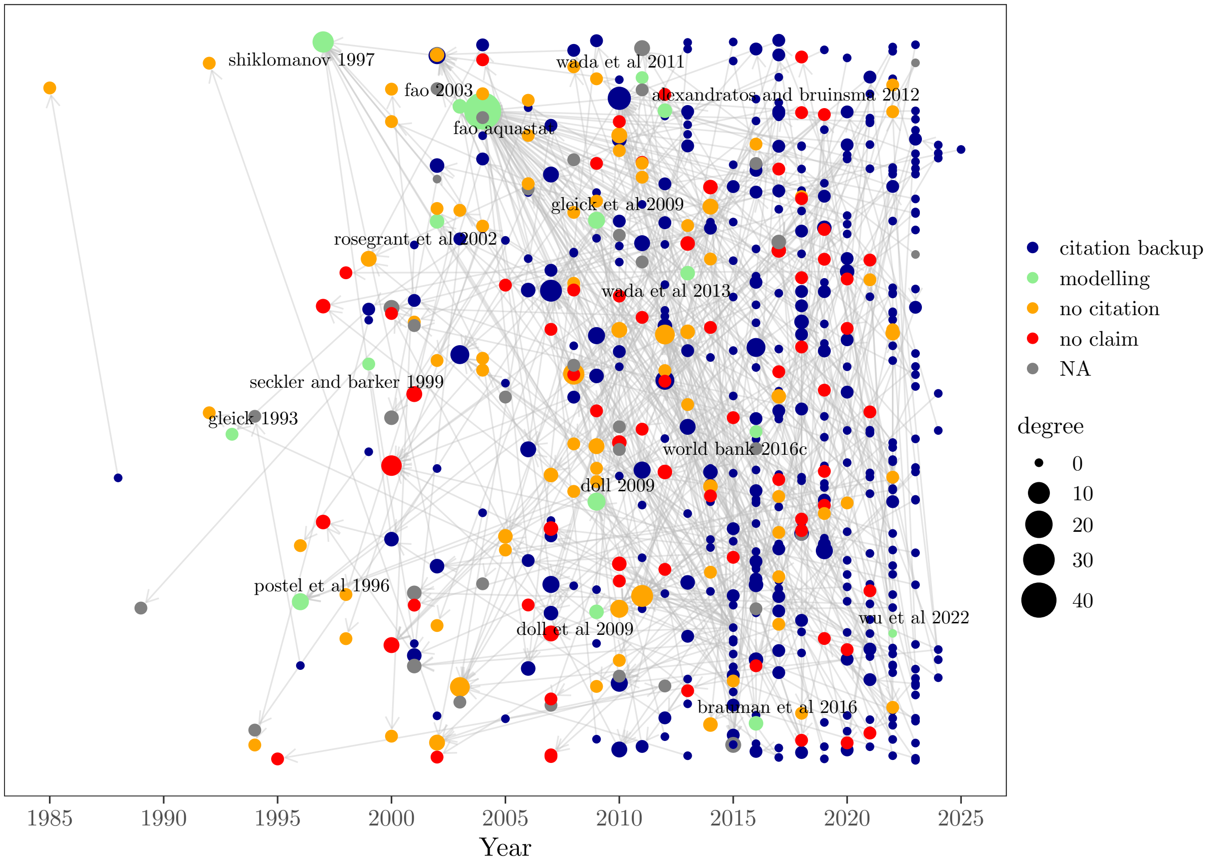
```
## [1] 17
```

```r
# Generate random values to replace NA -----------------------------------------

random_values <- sample(2000:2020, sum(na_indices), replace = TRUE)

# Replace NA with random values ------------------------------------------------

v_years[na_indices] <- random_values

# Define the coordinates--------------------------------------------------------

y_positions <- runif(length(v_years), min = -3, max = 3)  # Random y-axis positions
layout <- cbind(v_years, y_positions)  # Use actual years for x-axis
layout_matrix <- as.matrix(layout)
colnames(layout_matrix) <- c("x", "y")
```

```r
# PLOT NETWORK THROUGH TIME ####################################################

# Set seed ---------------------------------------------------------------------

set.seed(seed)
```

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

ggraph(graph.final[[2]], layout = layout_matrix, algorithm = "nicely") +
  geom_edge_link(arrow = arrow(length = unit(1.8, "mm")),
                 end_cap = circle(1, "mm"),
                 color = "grey",
                 alpha = 0.4) +
  geom_node_point(aes(color = nature.claim, size = degree)) +
  geom_node_text(aes(label = ifelse(nature.claim == "modelling", name, NA)),
                 repel = TRUE, size = 2.5) +
  scale_color_manual(name = "",
                     values = selected_colors) +
  scale_x_continuous(name = "Year",
                     limits = range(v_years),
                     breaks = seq(min(v_years),
                                  max(v_years), by = 5)) +
  labs(x = "Year", y = "") +
  theme_AP() +
  theme(axis.text.y = element_blank(),
        axis.ticks.y = element_blank())
```

```
## Warning: Removed 581 rows containing missing values (`geom_text_repel()`).
```



```
# FUNCTION TO PLOT EVOLUTION OF NETWORK THROUGH TIME ###########################

network_through_time_fun <- function(graph, Year, seed) {

  # Extract all names ----------------------------------------------------------

  all.names <- graph %>%
    activate(nodes) %>%
    pull(name)

  # Add names to edges ---------------------------------------------------------

  add.names.edges <- graph %>%
    activate(edges) %>%
    mutate(from.name = all.names[from],
           to.name = all.names[to])

  # Extract nodes by year ------------------------------------------------------

  names.targeted <- add.names.edges %>%
    activate(edges) %>%
    filter(year < Year) %>%
    data.frame() %>%
    .[, c("from.name", "to.name")] %>%
    c() %>%
    unlist() %>%
    unique(.)

  name.nodes <- add.names.edges %>%
    activate(nodes) %>%
    filter(name %in% names.targeted) %>%
    activate(edges) %>%
    filter(from.name %in% names.targeted & to.name %in% names.targeted)

  set.seed(seed)

  # Plot -----------------------------------------------------------------------

  out <- ggraph(name.nodes, layout = "igraph", algorithm = "nicely") +
    geom_edge_link(arrow = arrow(length = unit(1, 'mm')),
                   end_cap = circle(0.3, "mm")) +
    geom_node_point(aes(color = nature.claim), size = 0.5) +
    geom_node_text(aes(label = ifelse(nature.claim == "modelling", name, NA)),
                   repel = TRUE, size = 2.2) +
    scale_color_manual(name = "",
                       values = selected_colors) +
    labs(x = "", y = "") +
    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 = "none")

  return(out)

}

# DEFINE YEARS OF INTEREST #####################################################

years.vector <- c(seq(2000, 2020, 10), 2024)

# RUN FUNCTION #################################################################

plots.through.time <- list()

for (i in names(graph.final)) {

  plots.through.time[[i]] <- lapply(years.vector, function(year)
    network_through_time_fun(graph = graph.final[[i]], Year = year, seed = seed) +
      ggtitle(year))
}
```

```
# PLOT #########################################################################

# Extract legend ---------------------------------------------------------------

legend.plot <- list()

for (i in names(plots.through.time)) {

  legend.plot[[i]] <- get_legend(plots.through.time[[i]][[length(plots.through.time[[i]])]] +
                                   theme(legend.position = "top"))
}
```

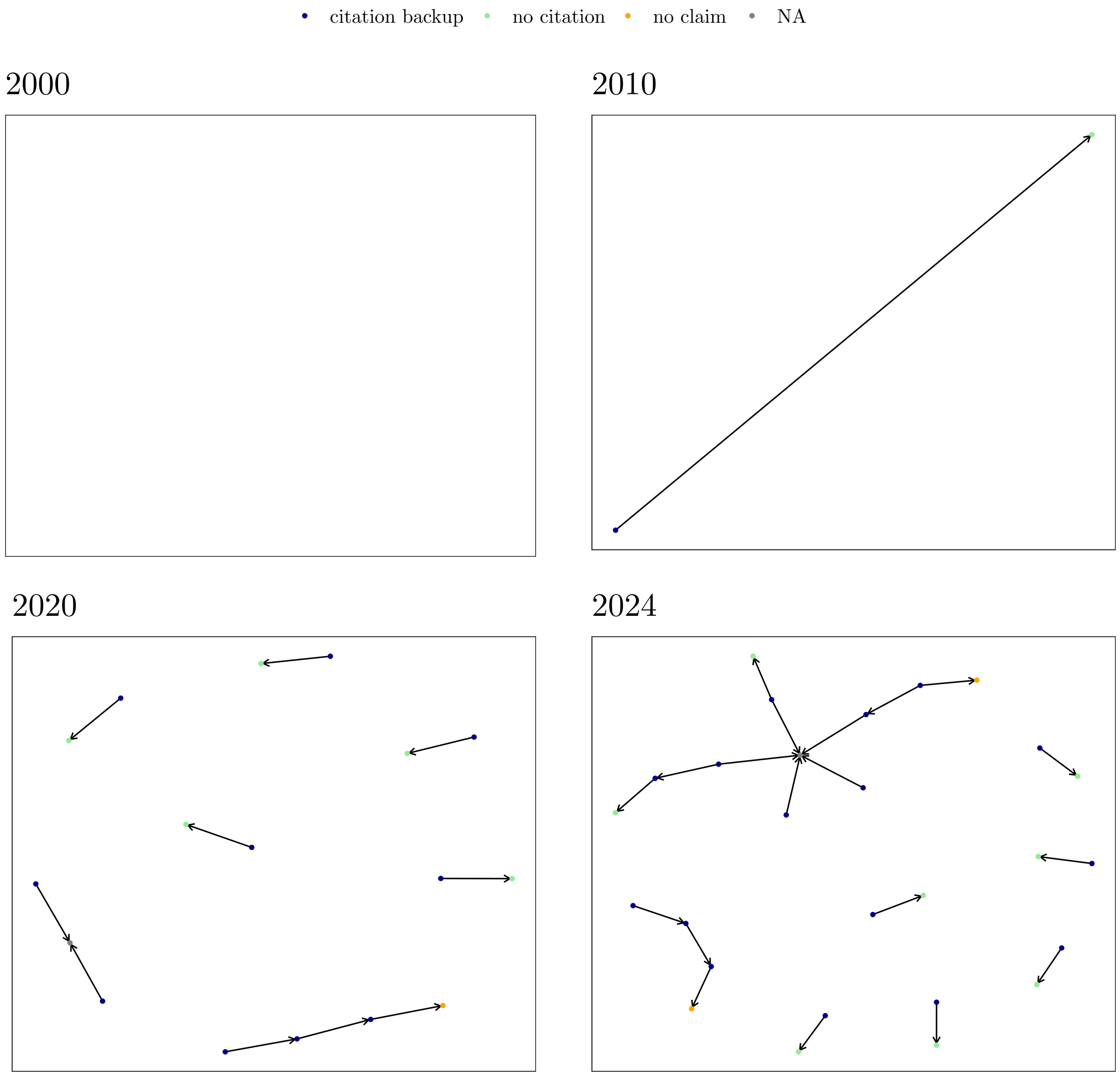
```
## Warning: Removed 27 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 567 rows containing missing values (`geom_text_repel()`).
```

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

bottom <- out.plot <- list()

for (i in names(plots.through.time)) {

  bottom[[i]] <- do.call(plot_grid, c(plots.through.time[[i]],
                                      nrow = floor(length(years.vector) / 2)))
  out.plot[[i]] <- plot_grid(legend.plot[[i]],
                             bottom[[i]], ncol = 1, rel_heights = c(0.1, 0.9))

}
```

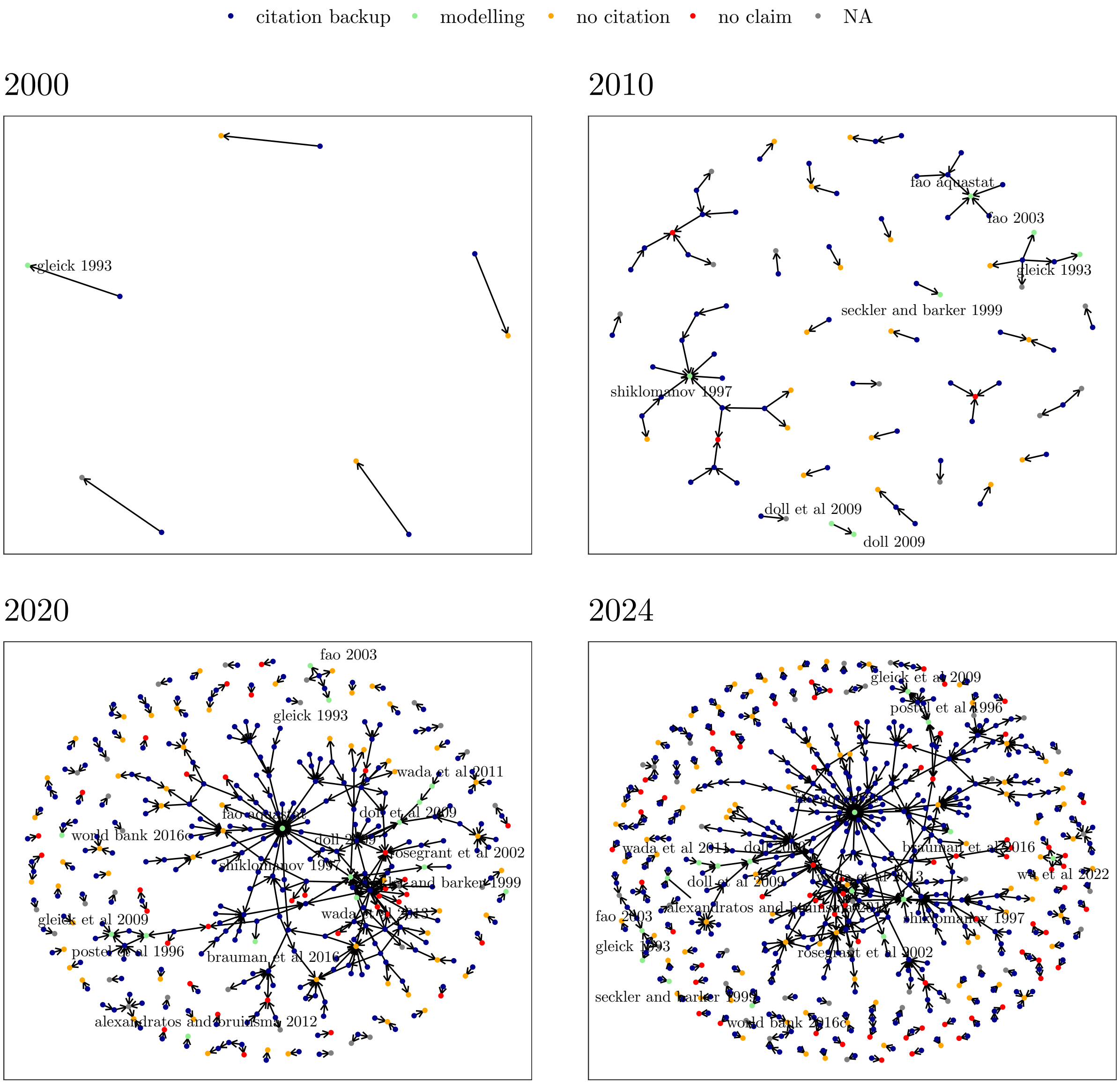
```
## Warning: Removed 2 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 17 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 27 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 9 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 86 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 371 rows containing missing values (`geom_text_repel()`).

## Warning: Removed 567 rows containing missing values (`geom_text_repel()`).
```

```
out.plot
```

```
## $food
```



```
##
## $water
```



4 Analysis of paths

4.1 "no claim" or "no citation" 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 <- lapply(graph.final, function(graph)
  graph %>%
    activate(nodes) %>%
    pull(name))

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

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

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

tmp <- list()

for(i in names(graph.final)) {
  tmp[[i]] <- lapply(list(no.claim_nodes[[i]],
                          no.claim.and.no.citation.nodes[[i]]), function(x)
    sort(nodes_to_no_claim_node_fun(graph.final[[i]], terminal_nodes = x)))
}

for(i in names(graph.final)) {
  names(tmp[[i]]) <- c("path ending in no claim",
                               "path ending in no claim or no citation")
}

tmp
```

```
## $food
## $food$`path ending in no claim`
## [1] "hanjra and qureshi 2010" "mitchell et al 2018"
## [3] "molden et al 2010"       "niu et al 2023"
## [5] "siebert and doll 2010"   "wang et al 2012"
##
## $food$`path ending in no claim or no citation`
##  [1] "borin 2023"                     "borsato et al 2020"
##  [3] "chartzoulakis and bertaki 2015" "de pascale et al 2011"
##  [5] "du preez et al 2018"            "evans and sadler 2008"
##  [7] "fao 2002"                       "fao 2003"
##  [9] "fao 2007b"                      "fernandez-cirelli et al 2009"
## [11] "hanjra and qureshi 2010"        "lobell et al 2006"
## [13] "mitchell et al 2018"            "molden et al 2010"
## [15] "niu et al 2023"                 "okorogbona et.al 2018"
## [17] "policy.1667934"                 "policy.1898497"
## [19] "rolle et al 2021"               "salmon et al 2015"
## [21] "siebert and doll 2010"          "turral et al 2010"
## [23] "wang et al 2012"                "world bank 2020"
## [25] "world bank 2021"
##
##
## $water
## $water$`path ending in no claim`
##   [1] "abbot et al 2019"                  "acosta et al 2016"
##   [3] "ahmed et al 2022"                  "alcamo et al 2007"
##   [5] "antia 2022"                        "badrul masud et al 2019"
##   [7] "bar et al 2015"                    "barreto and amaral 2018"
##   [9] "besharat et al 2020"               "biemans et al 2011"
##  [11] "bjornlund et al 2013"              "bondeau et al 2007"
##  [13] "boretti and rosa 2019"             "braun et al 2022"
##  [15] "calzadilla et al 2010"             "carmona et al 2017"
##  [17] "carvalho 2019"                     "chai et al 2016"
##  [19] "chirone et al 2022"                "clay 2004"
##  [21] "coelho et al 2012"                 "cristache et al 2018"
##  [23] "d'odorico et al 2019"              "de graaf et al 2017"
##  [25] "deikman et al 2012"                "dirwai et al 2022"
##  [27] "doll et al 2014"                   "droppers et al 2020"
##  [29] "du et al 2015"                     "eckert and kovalevska 2021"
##  [31] "elmoneim badr et al 2021"          "epri 2002"
##  [33] "faiz alam et al 2023"              "falkenmark 2013"
##  [35] "falkenmark et al 1997"             "fao 2007b"
##  [37] "fao 2016"                          "fao 2018c"
##  [39] "fao 2020"                          "friha et al 2022"
##  [41] "gan et al 2013"                    "gerten et al 2007"
##  [43] "gheewala et al 2014"               "giordano 2007"
##  [45] "gleick and palaniappan 2010"       "gleick et al 2011"
##  [47] "gleick et al 2018"                 "gordon et al 2010"
##  [49] "gorjian et al 2020"                "gorjian et al 2022"
##  [51] "grigas et al 2023"                 "gumidyala et al 2020"
##  [53] "gustinasari et al 2020"            "hanasaki et al 2008"
##  [55] "hanasaki et al 2008b"              "harun and hanafiah 2018"
##  [57] "hochmuth et al 2015"               "hoekstra 2003"
##  [59] "hoekstra and mekonnen 2012"        "hofste et al 2019"
##  [61] "holdren 2008"                      "howden et al 2013"
##  [63] "huang et al 2023"                  "huo et al 2022"
##  [65] "iwmi 2000"                         "jaramillo and destouni 2015"
##  [67] "jez et al 2016"                    "jiang et al 2017"
##  [69] "johnson et al 2001"                "jury and vaux jr 2005"
##  [71] "kaba gurmessa and assefa 2023"     "kabir et al 2023"
##  [73] "karimi et al 2019"                 "kaur saggi and jain 2022"
##  [75] "khan and hanjra 2009"              "kiani et al 2023"
##  [77] "kilemo 2022"                       "kong et al 2017"
##  [79] "kumar dubey et al 2021"            "kumar ravi et al 2023"
##  [81] "laluet et al 2024"                 "lamastra et al 2014"
##  [83] "li and long 2019"                  "li et al 2023b"
##  [85] "liu and yang 2010"                 "liu et al 2016"
##  [87] "marston et al 2018"                "martinez sosa et al 2023"
##  [89] "mcdermid et al 2023"               "meghan salmon et al 2015"
##  [91] "mekonnen and hoekstra 2012"        "mekonnen et al 2015"
##  [93] "menceloglu et al 2022"             "mendonca et al 2023"
##  [95] "mohanty et al 2018"                "moldovan et al 2022"
##  [97] "nahar sumiya and khatun 2016"      "nnadi et al 2015"
##  [99] "oladosu et al 2019"                "oladosu et al 2022"
## [101] "opio et al 2011"                   "othmani et al 2021"
## [103] "ozdogan et al 2010b"               "ozturk et al 2022"
## [105] "pang et al 2021"                   "payero et al 2006"
## [107] "pellegrini et al 2016"             "pena-arancibia et al 2016"
## [109] "perry et al 2017"                  "pimentel et al 1995"
## [111] "poersch-bortolon et al 2016"       "policy.1255933"
## [113] "policy.1435979"                    "policy.1781691"
## [115] "policy.1874989"                    "postel and vickers 2004"
## [117] "qin et al 2019"                    "ramankutty et al 2018"
## [119] "ran et al 2016"                    "redhu and jain 2023"
## [121] "ren et al 2018"                    "rivera et al 2017"
## [123] "rockstrom et al 2007"              "rodriguez et al 2022"
## [125] "roy et al 2022"                    "sadoff et al 2020"
## [127] "sahmat et al 2022"                 "sahray et al 2023"
## [129] "scanlon et al 2007"                "scanlon et al 2017"
## [131] "scanlon et al 2023"                "schreinemachers and tipraqsa 2012"
## [133] "sepaskhah and ahmadi 2010"         "seyboth and plattner 2014"
## [135] "shiklomanov 2000"                  "shtull-trauring et al 2016"
## [137] "siebert et al 2005"                "siebert et al 2010"
## [139] "siebert et al 2015"                "singh et al 2024"
## [141] "steward and bernard 2006"          "tabunshikov et al 2021"
## [143] "tiwari et al 2023"                 "tortell 2020"
## [145] "tsur 2005"                         "turner 2008"
## [147] "unesco 2001"                       "united nations 1998"
## [149] "united nations 2003"               "united nations 2021"
## [151] "united nations 2022"               "velez sanchez et al 2023"
## [153] "velis et al 2017"                  "vorosmarty et al 2000"
## [155] "vorosmarty et al 2010"             "wada 2015"
## [157] "wada et al 2014"                   "wada et al 2016"
## [159] "wajima 2018"                       "walter et al 2017"
## [161] "wbcsd  2009"                       "weatherhead and howden 2009"
## [163] "wisser et al 2010"                 "wmo 1997"
## [165] "world bank 2001"                   "world bank 2017"
## [167] "worldometers 2019"                 "wri 2000"
## [169] "wu et al 2022"                     "xu et al 2020"
## [171] "ye et al 2023"                     "yilmazkuday et al 2021"
## [173] "yin et al 2022"                    "young et al 2019"
## [175] "yu et al 2019"                     "zeman et al 2006"
## [177] "zeng et al 2022"                   "zhang et al 2015"
## [179] "zhang et al 2022"                  "zhuo et al 2022"
## 
## $water$`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] "ackerman 2015"
##   [6] "acosta et al 2016"
##   [7] "adama et al 2020"
##   [8] "adhikari et al 2021"
##   [9] "ahmed et al 2022"
##  [10] "alan rotz 2020"
##  [11] "alcamo et al 2007"
##  [12] "alvarez et al 2004"
##  [13] "anderson et al 2017"
##  [14] "angaleeswari et al 2021"
##  [15] "antia 2022"
##  [16] "appelgren 2000"
##  [17] "appuhamy et al 2016"
##  [18] "arboleda et al 2022"
##  [19] "babel and wahid 2008"
##  [20] "bac-dang et al 2019"
##  [21] "bach et al 2017"
##  [22] "badrul masud et al 2019"
##  [23] "bai et al 2010"
##  [24] "balyaminu 2017"
##  [25] "bar et al 2015"
##  [26] "barker 2015"
##  [27] "baroni et al 2007"
##  [28] "barreto and amaral 2018"
##  [29] "basiri jahromi et al 2020"
##  [30] "besharat et al 2020"
##  [31] "bhaskar and jain 2018"
##  [32] "bicca rodrigues 2014"
##  [33] "biemans et al 2011"
##  [34] "biswas 1999"
##  [35] "biswas and tortajada 2010"
##  [36] "bjornlund et al 2013"
##  [37] "bondeau et al 2007"
##  [38] "bonsch et al 2015"
##  [39] "bonsch et al 2016"
##  [40] "boretti and rosa 2019"
##  [41] "borin 2023"
##  [42] "boucher et al 2004"
##  [43] "bowden 2002"
##  [44] "braimoh 2013"
##  [45] "brar et al 2022"
##  [46] "braun et al 2022"
##  [47] "braune et al 2021"
##  [48] "brillo 2022"
##  [49] "brown 2008"
##  [50] "brown 2009"
##  [51] "cai and rosegrant 2002"
##  [52] "caldera and breyer 2019"
##  [53] "calzadilla et al 2010"
##  [54] "carmona et al 2017"
##  [55] "carvalho 2019"
##  [56] "ceres 2009"
##  [57] "chai et al 2016"
##  [58] "chakraborty et al 2017"
##  [59] "chen et al 2018"
##  [60] "chilinda et al 2021"
##  [61] "chirone et al 2022"
##  [62] "clapp et al 2017"
##  [63] "clay 2004"
##  [64] "coelho et al 2012"
##  [65] "connor 2017"
##  [66] "connor et al 2017"
##  [67] "corcoran et al 2010"
##  [68] "cristache et al 2018"
##  [69] "d'odorico et al 2019"
##  [70] "dalin et al 2012"
##  [71] "dave and nalco 2004"
##  [72] "de graaf et al 2017"
##  [73] "de pascale et al 2011"
##  [74] "deikman et al 2012"
##  [75] "dirwai et al 2022"
##  [76] "doll 2008"
##  [77] "doll et al 2014"
##  [78] "doungmanee 2016"
##  [79] "droppers et al 2020"
##  [80] "du et al 2015"
##  [81] "dunkelman et al 2017"
##  [82] "eckert and kovalevska 2021"
##  [83] "egbeyemi et al 2023"
##  [84] "elbakidze and cobourn 2014"
##  [85] "elgallal et al 2016"
##  [86] "elhussiny et al 2023"
##  [87] "elmoneim badr et al 2021"
##  [88] "epri 2002"
##  [89] "evans and sadler 2008"
##  [90] "faiz alam et al 2023"
##  [91] "falkenmark 2013"
##  [92] "falkenmark et al 1997"
##  [93] "fao 1996"
##  [94] "fao 2002"
##  [95] "fao 2002b"
##  [96] "fao 2007"
##  [97] "fao 2007b"
##  [98] "fao 2010"
##  [99] "fao 2011"
## [100] "fao 2012"
## [101] "fao 2012b"
## [102] "fao 2015"
## [103] "fao 2016"
## [104] "fao 2017"
## [105] "fao 2018"
## [106] "fao 2018c"
## [107] "fao 2019"
## [108] "fao 2020"
## [109] "fereres and soriano 2006"
## [110] "firdayati et al 2022"
## [111] "fitton et al 2019"
## [112] "fitzgerald and auerbach 2016"
## [113] "fogel and palmer 2014"
## [114] "friha et al 2022"
## [115] "gallardo 2015"
## [116] "gan et al 2013"
## [117] "gavrilescu et al 2008"
## [118] "gerbens-leenes and nonhebel 2004"
## [119] "gerten et al 2007"
## [120] "gheewala et al 2014"
## [121] "giordano 2007"
## [122] "gleick and palaniappan 2010"
## [123] "gleick et al 2002"
## [124] "gleick et al 2011"
## [125] "gleick et al 2014"
## [126] "gleick et al 2018"
## [127] "gong et al 2020"
## [128] "gordon et al 2010"
## [129] "gorjian et al 2020"
## [130] "gorjian et al 2022"
## [131] "gourbesville 2008"
## [132] "grigas et al 2023"
## [133] "gumidyala et al 2020"
## [134] "gurung 2016"
## [135] "gustinasari et al 2020"
## [136] "gwp nd"
## [137] "haddeland et al 2013"
## [138] "hanasaki et al 2008"
## [139] "hanasaki et al 2008b"
## [140] "hannah 2017"
## [141] "harun and hanafiah 2018"
## [142] "he et al 2023"
## [143] "hegazi et al 2023"
## [144] "hochmuth et al 2015"
## [145] "hoekstra 2003"
## [146] "hoekstra and mekonnen 2012"
## [147] "hofste et al 2019"
## [148] "hofwegen and svendsen 2000"
## [149] "holdren 2008"
## [150] "howden et al 2013"
## [151] "huang et al 2023"
## [152] "huo et al 2022"
## [153] "hussein bapir and wasman hamad 2023"
## [154] "iaastd 2009"
## [155] "ingrao et al 2023"
## [156] "ipcc 2007"
## [157] "iwmi 2000"
## [158] "jagermeyr et al 2017"
## [159] "jaramillo and destouni 2015"
## [160] "jat et al 2016"
## [161] "jehan et al 2022"
## [162] "jez et al 2016"
## [163] "jiang et al 2017"
## [164] "jimenez-arias et al 2023"
## [165] "johansson et al 2015"
## [166] "johnson et al 2001"
## [167] "jury and vaux jr 2005"
## [168] "kaba gurmessa and assefa 2023"
## [169] "kabir et al 2023"
## [170] "kalboussi et al 2022"
## [171] "kang et al 2017"
## [172] "kapahi et al 2022"
## [173] "karimi et al 2019"
## [174] "kaur saggi and jain 2022"
## [175] "kayikcioglu 2012"
## [176] "khair et al 2020"
## [177] "khan and hanjra 2009"
## [178] "khosravifar et al 2020"
## [179] "kiani et al 2023"
## [180] "kilemo 2022"
## [181] "kiran kumara et al 2020"
## [182] "kocian and incrocci 2020"
## [183] "kong et al 2017"
## [184] "kopittke et al 2019"
## [185] "kumar dubey et al 2021"
## [186] "kumar ravi et al 2023"
## [187] "kundzewicz et al. 2007"
## [188] "kwasek 2013"
## [189] "laluet et al 2024"
## [190] "lamastra et al 2014"
## [191] "lang 2014"
## [192] "legesse lebre et al 2021"
## [193] "li and long 2019"
## [194] "li et al 2023"
## [195] "li et al 2023b"
## [196] "liu and yang 2010"
## [197] "liu et al 2016"
## [198] "lok 2001"
## [199] "lynch et al 2023"
## [200] "maldonado junior et al 2019"
## [201] "marston et al 2018"
## [202] "martinez sosa et al 2023"
## [203] "mashnik et al 2017"
## [204] "matile et al 2013"
## [205] "mcdaniel et al 2017"
## [206] "mcdermid et al 2023"
## [207] "meghan salmon et al 2015"
## [208] "mekonnen and hoekstra 2012"
## [209] "mekonnen et al 2015"
## [210] "menceloglu et al 2022"
## [211] "mendonca et al 2023"
## [212] "mettetal 2019"
## [213] "miao et al 2022"
## [214] "millenium ecosystem assessment 2005"
## [215] "millenium project 2004"
## [216] "mohanty et al 2018"
## [217] "mohorjy 1988"
## [218] "moldovan et al 2022"
## [219] "molle 2002"
## [220] "monteoliva-garcia et al 2020"
## [221] "nahar sumiya and khatun 2016"
## [222] "newell and taylor 2017"
## [223] "nnadi et al 2015"
## [224] "no author"
## [225] "nordin et al 2013"
## [226] "norton-brandao et al 2013"
## [227] "nunes correia 1999"
## [228] "o'connell and billingsley 2020"
## [229] "odeku 2020"
## [230] "oecd 2010"
## [231] "oecd 2017"
## [232] "oecd nd"
## [233] "ofori et al 2021"
## [234] "ohyama et al 2023"
## [235] "oladosu et al 2019"
## [236] "oladosu et al 2022"
## [237] "opio et al 2011"
## [238] "ortega-munoz et al 2024"
## [239] "ostberg et al 2018"
## [240] "othmani et al 2021"
## [241] "ozdogan et al 2010"
## [242] "ozdogan et al 2010b"
## [243] "ozturk et al 2022"
## [244] "pang et al 2021"
## [245] "parameshwari 2017"
## [246] "pardossi et al 2009"
## [247] "pastor et al 2019"
## [248] "pauzuolien et al 2022"
## [249] "payero et al 2006"
## [250] "pedrero et al 2010"
## [251] "pellegrini et al 2016"
## [252] "pena-arancibia et al 2016"
## [253] "pennisi 2008"
## [254] "perry et al 2017"
## [255] "pfister and bayer 2013"
## [256] "pimentel et al 1995"
## [257] "poersch-bortolon et al 2016"
## [258] "pokhrel et al 2012"
## [259] "pokhrel et al 2016"
## [260] "policy.1094742"
## [261] "policy.1252526"
## [262] "policy.1255933"
## [263] "policy.1257844"
## [264] "policy.1381456"
## [265] "policy.1435979"
## [266] "policy.1666264"
## [267] "policy.1781691"
## [268] "policy.1874989"
## [269] "policy.229461"
## [270] "policy.240747"
## [271] "policy.718260"
## [272] "postel 1985"
## [273] "postel 2001"
## [274] "postel and vickers 2004"
## [275] "prochazka et al 2018"
## [276] "qin et al 2019"
## [277] "rahmadian and widyartono 2019"
## [278] "ramankutty et al 2018"
## [279] "ran et al 2016"
## [280] "redhu and jain 2023"
## [281] "rehman et al 2022"
## [282] "ren et al 2018"
## [283] "ricart and rico 2019"
## [284] "ridgway et al 2019"
## [285] "ridoutt et al 2009"
## [286] "ringler et al 2022"
## [287] "ritchie and roser 2017"
## [288] "rivera et al 2017"
## [289] "rivers et al 2015"
## [290] "rockstrom and gordon 2001"
## [291] "rockstrom et al 2007"
## [292] "rodriguez and lozano-juste 2015"
## [293] "rodriguez et al 2022"
## [294] "rodriguez-espinosa et al 2023"
## [295] "romano et al 2023"
## [296] "romero-trigueros et al 2019"
## [297] "rosegrant and ringler 1998"
## [298] "rosegrant et al 2009"
## [299] "rost et al 2008"
## [300] "roudi-fahim et al 2018"
## [301] "roy et al 2022"
## [302] "sadoff et al 2020"
## [303] "saeidian et al 2015"
## [304] "sahmat et al 2022"
## [305] "sahray et al 2023"
## [306] "salman and salman 2002"
## [307] "samad et al 1992"
## [308] "scanlon et al 2007"
## [309] "scanlon et al 2017"
## [310] "scanlon et al 2023"
## [311] "schreinemachers and tipraqsa 2012"
## [312] "schulte et al 2014"
## [313] "seckler et al 1998"
## [314] "sepaskhah and ahmadi 2010"
## [315] "seyboth and plattner 2014"
## [316] "shang et al 2024"
## [317] "shiklomanov 1999"
## [318] "shiklomanov 2000"
## [319] "shiklomanov and rodda 2003"
## [320] "shtull-trauring et al 2016"
## [321] "siebert and doll 2010"
## [322] "siebert et al 2005"
## [323] "siebert et al 2010"
## [324] "siebert et al 2013"
## [325] "siebert et al 2015"
## [326] "singh et al 2022"
## [327] "singh et al 2024"
## [328] "sonen capital 2016"
## [329] "song and song 2023"
## [330] "sophocleous 2004"
## [331] "spiegel international 2009"
## [332] "steduto et al 2018"
## [333] "steffen 2017"
## [334] "steward and bernard 2006"
## [335] "swatuk et al 2018"
## [336] "tabunshikov et al 2021"
## [337] "tavares et al 2022"
## [338] "ti et al 2021"
## [339] "tian et al 2022"
## [340] "tiwari et al 2023"
## [341] "tortell 2020"
## [342] "tsiropoulos et al 2022"
## [343] "tsur 2005"
## [344] "tunc and sahin 2025"
## [345] "tuninetti et al 2015"
## [346] "turner 2008"
## [347] "unctad 2011"
## [348] "unep 2011"
## [349] "unesco 2001"
## [350] "unesco 2006"
## [351] "unesco 2014"
## [352] "unesco 2017"
## [353] "united nations 1998"
## [354] "united nations 2003"
## [355] "united nations 2015"
## [356] "united nations 2021"
## [357] "united nations 2022"
## [358] "united nations 2023"
## [359] "velez sanchez et al 2023"
## [360] "velis et al 2017"
## [361] "velpuri et al 2009"
## [362] "vorosmarty et al 2000"
## [363] "vorosmarty et al 2005"
## [364] "vorosmarty et al 2010"
## [365] "wada 2015"
## [366] "wada et al 2013b"
## [367] "wada et al 2014"
## [368] "wada et al 2016"
## [369] "wajima 2018"
## [370] "walter et al 2017"
## [371] "wbcsd  2009"
## [372] "weatherhead and howden 2009"
## [373] "who 2014"
## [374] "williams et al 2017"
## [375] "wilson 2013"
## [376] "winpenny et al 2010"
## [377] "wisser et al 2008"
## [378] "wisser et al 2010"
## [379] "wmo 1997"
## [380] "wood et al 2000"
## [381] "world bank 1992"
## [382] "world bank 2001"
## [383] "world bank 2017"
## [384] "world bank 2021"
## [385] "world watch institute 2004"
## [386] "world water assessment programme 2003"
## [387] "world water assessment programme 2014"
## [388] "worldometers 2019"
## [389] "wri 1994"
## [390] "wri 2000"
## [391] "wu et al 2022"
## [392] "wwap 2018"
## [393] "wwf 2006"
## [394] "xing yuan et al 2024"
## [395] "xu et al 2020"
## [396] "ye et al 2023"
## [397] "yilmazkuday et al 2021"
## [398] "yin et al 2022"
## [399] "young et al 2019"
## [400] "yu et al 2019"
## [401] "zeman et al 2006"
## [402] "zeng et al 2022"
## [403] "zhang et al 2015"
## [404] "zhang et al 2015b"
## [405] "zhang et al 2022"
## [406] "zhao et al 2022"
## [407] "zhou et al 2022"
## [408] "zhuo et al 2022"
```

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

out <- list()

for(i in names(tmp)) {
  out[[i]] <- lapply(tmp[[i]], function(x) length(x) / length(all_nodes[[i]]))
}

out
```

```
## $food
## $food$`path ending in no claim`
## [1] 0.1818182
##
## $food$`path ending in no claim or no citation`
## [1] 0.7575758
##
##
## $water
## $water$`path ending in no claim`
## [1] 0.3015075
##
## $water$`path ending in no claim or no citation`
## [1] 0.6834171
```

4.2 Calculation of amplification

```r
# CREATE FUNCTION TO CHECK AMPLIFICATION #######################################

# amplification measure for paper P: defined as the number of
# citation-paths originating at P and terminating at all other papers,
# except for paths of length 1 flowing directly to modelling papers.

amplification_fun <- function(graph) {

  # Convert tbl_graph to igraph object -----------------------------------------

  ig <- as.igraph(graph)
  nature_claims <- V(ig)$nature.claim

  # initialize counter to store results for each paper -------------------------

  results <- numeric(vcount(ig))

  # Loop over each paper --------------------------------------------------------

  for (P in V(ig)) {

    # Initialize counter for valid paths
    path_count <- 0

    # Traverse through all nodes and count paths avoiding direct "modelling"
    for (target in V(ig)) {

      if (P != target) {

        all_paths <- all_simple_paths(ig, from = P, to = target, mode = "out")

        # Filter out paths of length 1 that end in a "modelling" node
        valid_paths <- Filter(function(path) {
          !(length(path) == 2 && nature_claims[path[2]] == "modelling")
        }, all_paths)

        path_count <- path_count + length(valid_paths)
      }
    }

    results[P] <- path_count
  }

  return(results)
}

# RUN AMPLIFICATION FUNCTION ###################################################

amplification.indices <- lapply(graph.final, function(graph)
  amplification_fun(graph))

# Calculate average amplification index of the networks ------------------------
# (e.g., the number paths initiated by the average paper
# leading to studies that do # not flow directly to "primary" data)
lapply(amplification.indices, function(x) mean(x))
```

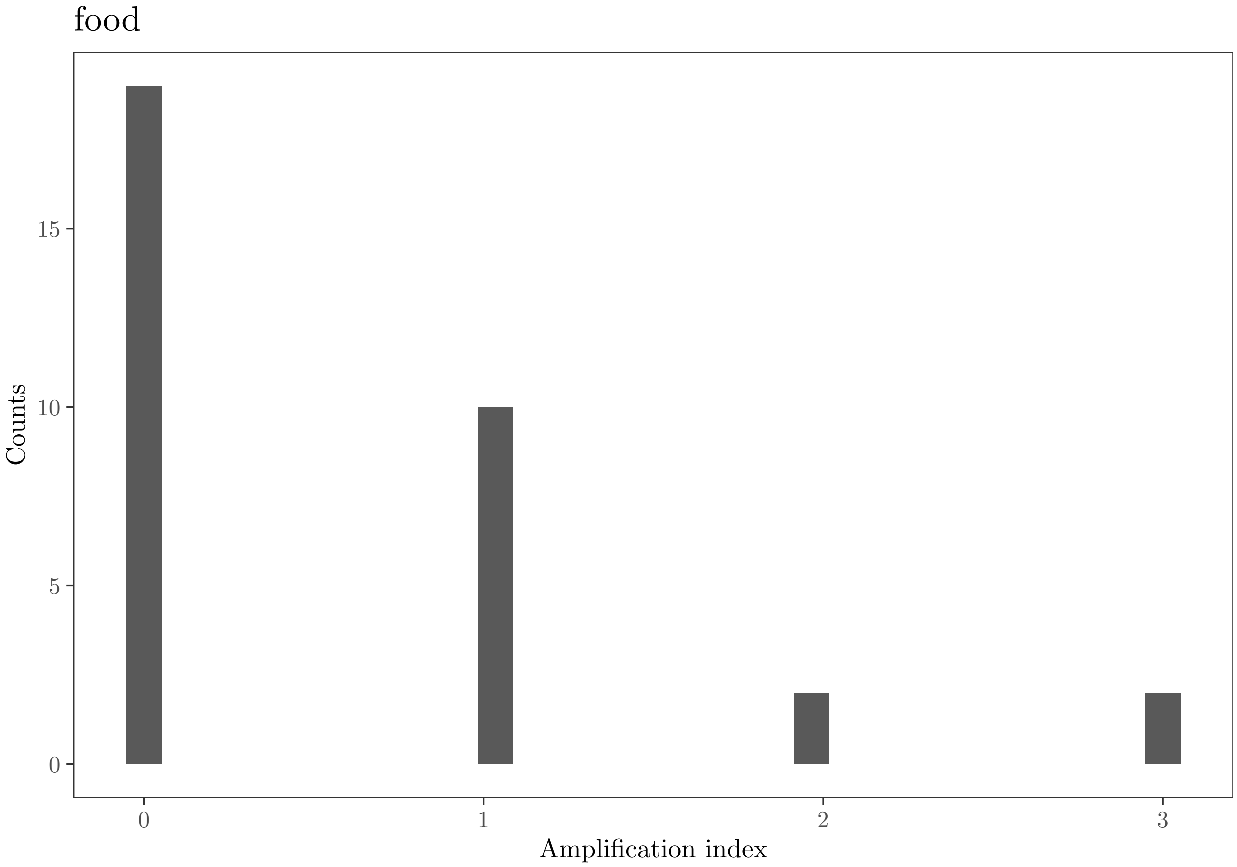
```
## $food
## [1] 0.6060606
```

```
##
## $water
## [1] 1.355109
```

```
# PLOT DISTRIBUTION OF AMPLIFICATION INDIXES ###################################

plot.amplification <- list()

for (i in names(amplification.indices)) {

  plot.amplification[[i]] <- amplification.indices[[i]] %>%
    data.frame("index" = .) %>%
    ggplot(., aes(index)) +
    geom_histogram() +
    theme_AP() +
    labs(y = "Counts", x = "Amplification index") +
    ggtitle(names(amplification.indices[i]))

}

plot.amplification
```

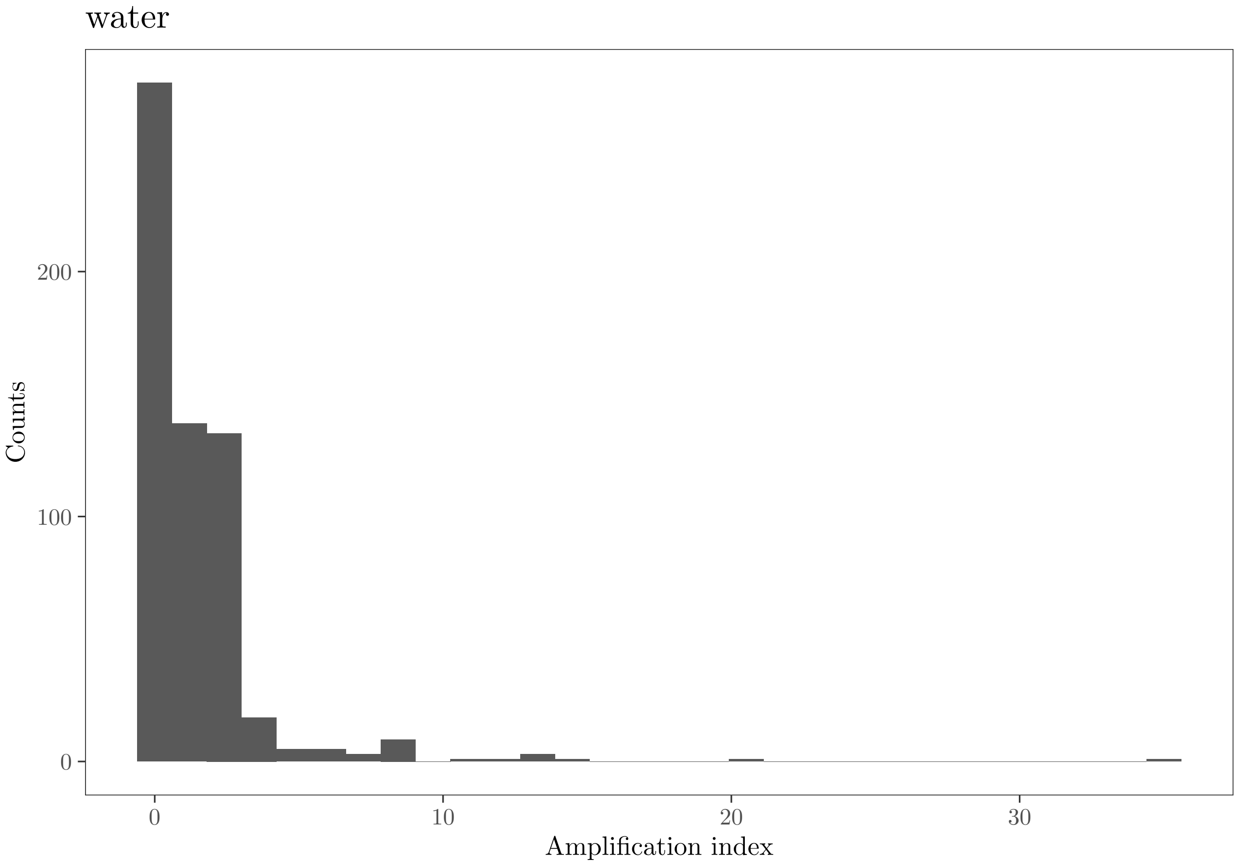
```
## $food
```

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



```
##
## $water
```

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



5 Study of Aquastat values

```
# STUDY OF AQUASTAT PERCENTAGES ################################################

# Read in aquastat dataset -----------------------------------------------------

aquastat.dt <- read.xlsx("aquastat_dt.xlsx") %>%
  data.table() %>%
  .[Year == 2020] %>%
  setnames(., c("Value", "Area"), c("percentage", "country")) %>%
  .[, .(country, percentage)] %>%
  .[, data:= "aquastat 2020"] %>%
  .[, country:= countrycode(country, origin = "country.name", destination = "country.name")]
```

```
## Warning: Some values were not matched unambiguously: Australia and New Zealand

## Warning: Some strings were matched more than once, and therefore set to <NA> in the result:
```

```
aquastat.dt[, continent:= countrycode(country, origin = "country.name", destination = "continen

# Read in world resources institute dataset ------------------------------------

wri <- fread("world_resources_institut_guide_to_the_global_environment_1994.csv") %>%
  .[order(country)] %>%
  .[, data:= "wri 1994"] %>%
  .[, country:= countrycode(country, origin = "country.name", destination = "country.name")]
```

```
## Warning: Some values were not matched unambiguously: , Cote d'lvoire
```

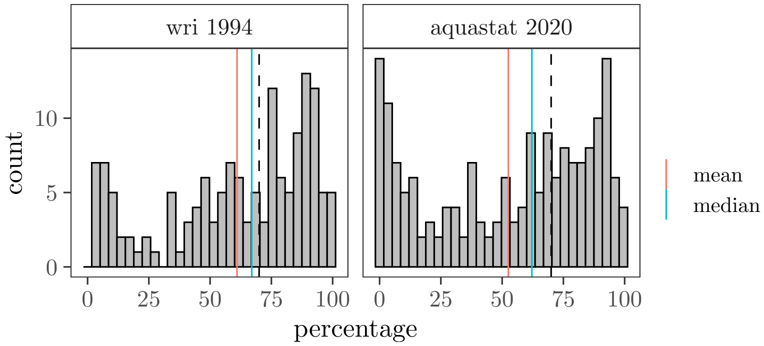
```
wri[, continent:= countrycode(country, origin = "country.name", destination = "continent")]
```

```
## Warning: Some values were not matched unambiguously: Czechoslovakia, Yugoslavia
```

```
# Compare distributions -------------------------------------------------------

dt.comparison <- rbind(aquastat.dt, wri) %>%
  .[, data:= factor(data, levels = c("wri 1994", "aquastat 2020"))]

dt.stats.comparison <- dt.comparison[, .(mean = mean(percentage, na.rm = TRUE),
                                        median = median(percentage, na.rm = TRUE)), data] %>%
  melt(., measure.vars = c("mean", "median"))


ggplot(dt.comparison, aes(percentage)) +
  geom_histogram(color = "black", fill = "grey") +
  facet_wrap(~data) +
  geom_vline(data = dt.stats.comparison, aes(xintercept = value, color = variable)) +
  scale_color_discrete(name = "") +
  geom_vline(xintercept = 70, lty = 2) +
  theme_AP()
```

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

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

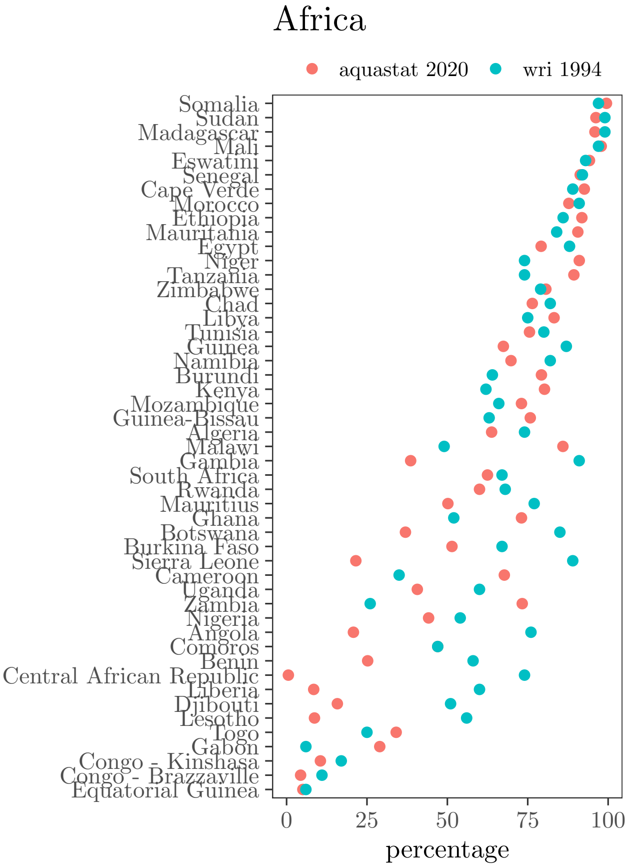


```
# At the country level --------------------------------------------------------

tmp <- aquastat.dt[wri, on = c("country", "continent")] %>%
  .[, .(country,continent, percentage, i.percentage)] %>%
  setnames(., c("percentage", "i.percentage"), c("aquastat 2020", "wri 1994")) %>%
  melt(., measure.vars = c("aquastat 2020", "wri 1994")) %>%
  .[, country:= ifelse(country == "Trinidad & Tobago", "Trinidad and Tobago", country)] %>%
  na.omit() %>%
  split(., .$continent)
```

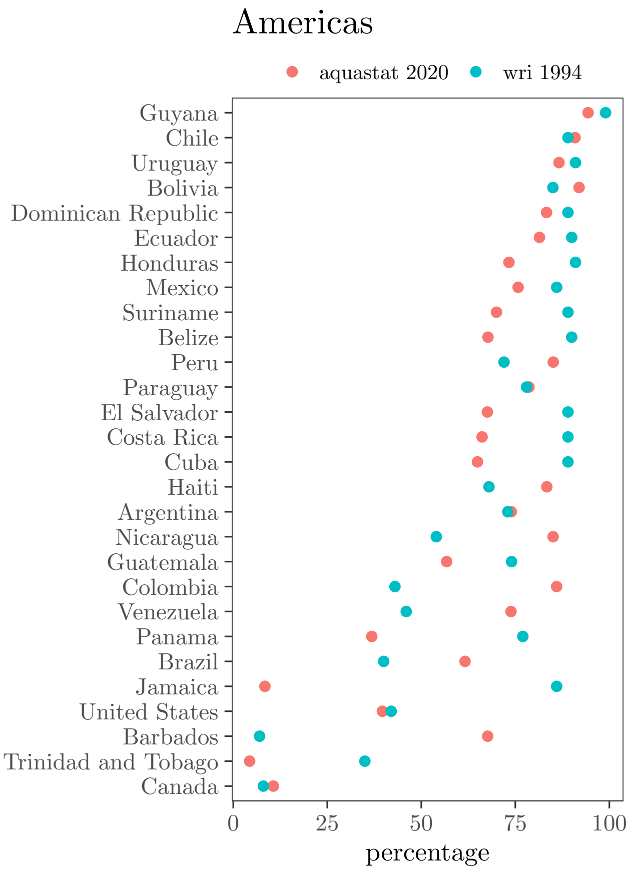
```
## Warning in melt.data.table(., measure.vars = c("aquastat 2020", "wri 1994")):
## 'measure.vars' [aquastat 2020, wri 1994] are not all of the same type. By order
## of hierarchy, the molten data value column will be of type 'double'. All
## measure variables not of type 'double' will be coerced too. Check DETAILS in
## ?melt.data.table for more on coercion.
```

```
out <- list()

for(i in names(tmp)) {

  out[[i]] <- ggplot(tmp[[i]], aes(reorder(country, value),
                                   value, color = variable)) +
    coord_flip() +
    scale_color_discrete(name = "") +
    geom_point() +
    theme_AP() +
    theme(legend.position = "top") +
    labs(x = "", y = "percentage") +
    ggtitle(names(tmp[i]))
}

out
```

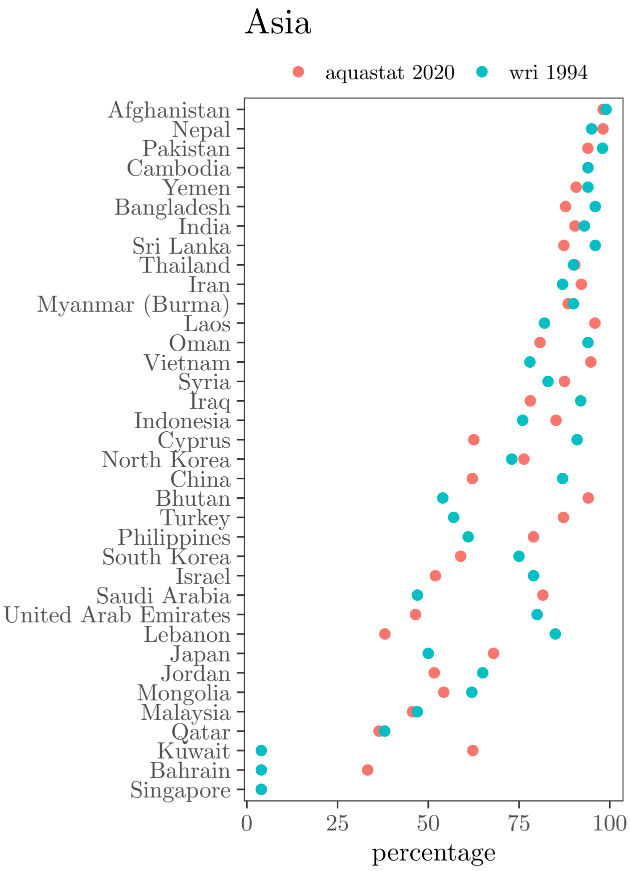
```
## $Africa
```


Africa
aquastat 2020
wri 1994
Somalia
Sudan
Madagascar
Mali
Eswatini
Senegal
Cape Verde
Morocco
Ethiopia
Mauritania
Egypt
Niger
Tanzania
Zimbabwe
Chad
Libya
Tunisia
Guinea
Namibia
Burundi
Kenya
Mozambique
Guinea-Bissau
Algeria
Malawi
Gambia
South Africa
Rwanda
Mauritius
Ghana
Botswana
Burkina Faso
Sierra Leone
Cameroon
Uganda
Zambia
Nigeria
Angola
Comoros
Benin
Central African Republic
Liberia
Djibouti
Lesotho
Togo
Gabon
Congo - Kinshasa
Congo - Brazzaville
Equatorial Guinea
0
25
50
75
100
percentage

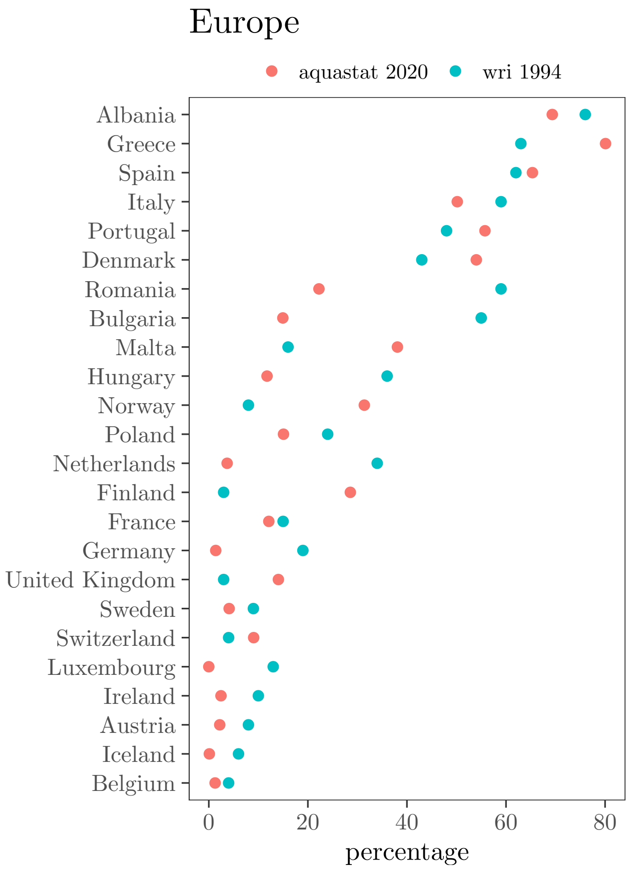
```
##
## $Americas
```


Americas
● aquastat 2020
● wri 1994
Guyana
Chile
Uruguay
Bolivia
Dominican Republic
Ecuador
Honduras
Mexico
Suriname
Belize
Peru
Paraguay
El Salvador
Costa Rica
Cuba
Haiti
Argentina
Nicaragua
Guatemala
Colombia
Venezuela
Panama
Brazil
Jamaica
United States
Barbados
Trinidad and Tobago
Canada
0
25
50
75
100
percentage

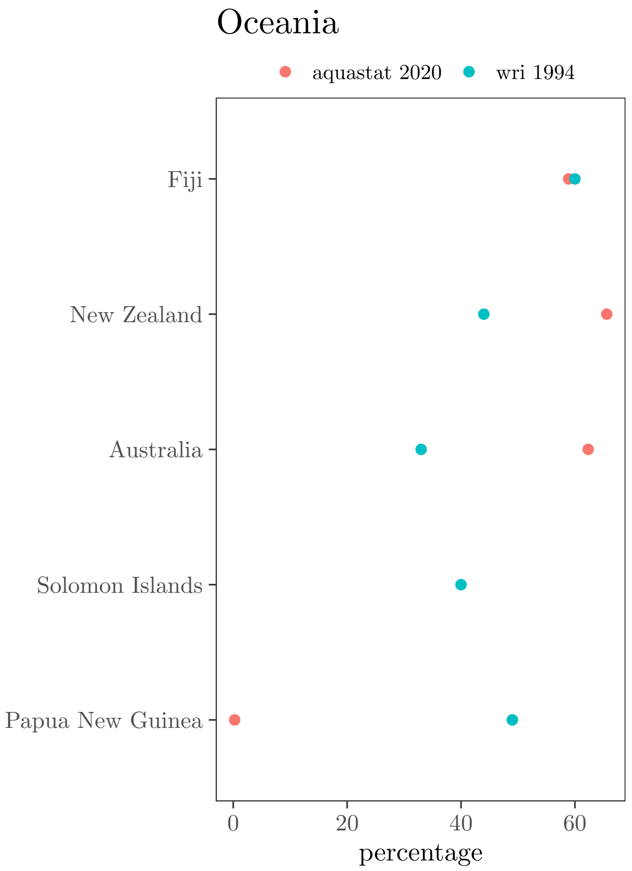
```
##
## $Asia
```


Asia
● aquastat 2020
● wri 1994
Afghanistan
Nepal
Pakistan
Cambodia
Yemen
Bangladesh
India
Sri Lanka
Thailand
Iran
Myanmar (Burma)
Laos
Oman
Vietnam
Syria
Iraq
Indonesia
Cyprus
North Korea
China
Bhutan
Turkey
Philippines
South Korea
Israel
Saudi Arabia
United Arab Emirates
Lebanon
Japan
Jordan
Mongolia
Malaysia
Qatar
Kuwait
Bahrain
Singapore
0
25
50
75
100
percentage

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


Europe
aquastat 2020
wri 1994
Albania
Greece
Spain
Italy
Portugal
Denmark
Romania
Bulgaria
Malta
Hungary
Norway
Poland
Netherlands
Finland
France
Germany
United Kingdom
Sweden
Switzerland
Luxembourg
Ireland
Austria
Iceland
Belgium
0
20
40
60
80
percentage

```
##
## $Oceania
```


Oceania
aquastat 2020
wri 1994
Fiji
New Zealand
Australia
Solomon Islands
Papua New Guinea
0
20
40
60
percentage

6 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
```