

# Irrigation's real impact on global water and food security

R code

Arnald Puy

## Contents

<b>1</b>	<b>Retrieve all corpus</b>	<b>6</b>
1.1	Abstract corpus . . . . .	6
1.2	Policy corpus . . . . .	6
1.3	Full text corpus . . . . .	7
<b>2</b>	<b>Split full text corpus for analysis</b>	<b>8</b>
<b>3</b>	<b>Network analysis</b>	<b>9</b>
3.1	Network metrics . . . . .	14
3.2	Network plots . . . . .	17
3.3	Network through time . . . . .	28
<b>4</b>	<b>Analysis of paths</b>	<b>30</b>
<b>5</b>	<b>Session information</b>	<b>41</b>

```

# 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))
}

```

```

# 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 properly
##
## Please, take a look at the vignettes:
## - 'Data Importing and Converting' (https://www.bibliometrix.org/vignettes/Data-Importing-and-Converting)
## - 'A brief introduction to bibliometrix' (https://www.bibliometrix.org/vignettes/Introduction-to-Bibliometrix)
##
##
## 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 properly
##
## Please, take a look at the vignettes:

```

```
## - 'Data Importing and Converting' (https://www.bibliometrix.org/vignettes/Data-Importing-and-Converting)
## - 'A brief introduction to bibliometrix' (https://www.bibliometrix.org/vignettes/Introduction-to-Bibliometrix)
##
##
## 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)

# Filter 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 -----
```

```
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 -----
```

```

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

# 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

# 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

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

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

## 2 Split full text corpus for analysis

```
# 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    649
## 2: water         <NA>    56
## 3: water          T    349
## 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, SEnv, 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      8
##  3: fao aquastat 2010  water      5
##  4: fao aquastat 2020  water      2
##  5: fao aquastat 2011  water      2
##  6: fao aquastat 2012  water      3
##  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      4
## 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   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          594
## 2:   food           48

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

# 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   420        720 0.58333333
## 6: water    modelling       22        720 0.03055556

```

```
## 7: water      no citation  194      720 0.26944444
## 8: water      <NA>       22      720 0.03055556
## 9: water      no claim   56      720 0.07777778
```

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

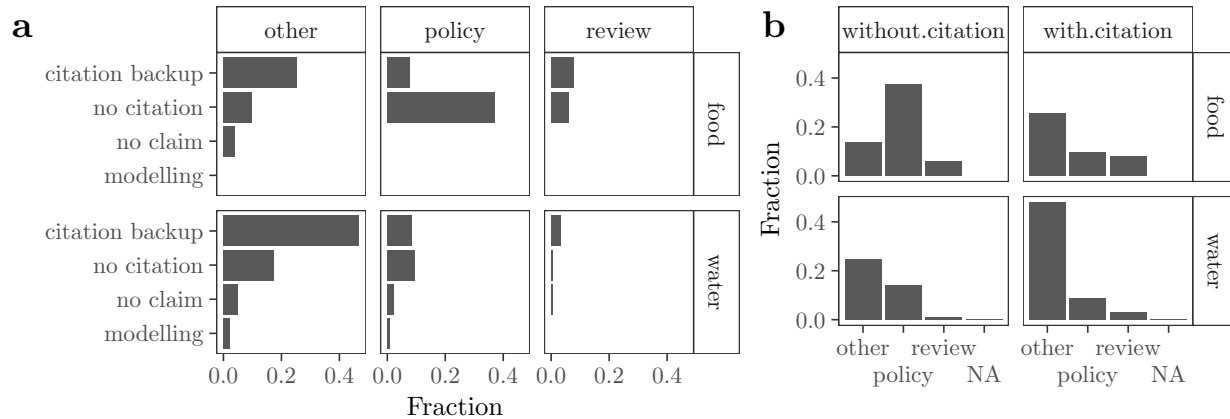
```
a <- network.dt[, .N, .(nature.claim, document.type, topic)] %>%
  merge(., total.rows, by = "topic") %>%
  .[, proportion:= N / number.rows] %>%
  na.omit() %>%
  ggplot(., aes(reorder(nature.claim, proportion), proportion)) +
  coord_flip() +
  geom_bar(stat = "identity") +
  facet_grid(topic~document.type) +
  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 -----
```

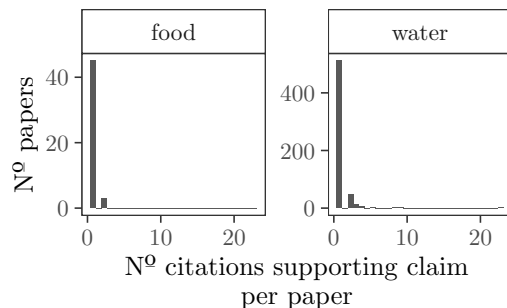
```
plot_grid(a, b, ncol = 2, rel_widths = c(0.6, 0.4), labels = "auto")
```



*# 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 = "N° citations supporting claim \n per paper", y = "N° 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)
```

*# Transform to graph -----*

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

```

# Calculate network metrics -----

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

## $food
## [1] 0.02083333
##
## $water
## [1] 0.002068218

# 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] 0.8355441

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     37         0          0.0        NaN 0.06901286
## 2: molden et al 2007     11         1         24.0 0.3333333 0.01294539
## 3:      fao 2011        10         1         10.5 1.0000000 0.01109908

```

```
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 0.001215815
## 2: wada et al 2014         1         9         10 0.09090909 0.001422503
## 3: wada et al 2016         0         8          0 0.06250000 0.001215815

```

```
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:      molden et al 2007     11         1      24.00000 0.3333333 0.012945387
## 2: boretti and rosa 2019       2         4      22.00000 0.0555556 0.003282700
## 3:      siebert et al 2010     6         3      16.33333 0.3333333 0.005136156
```

```
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 0.001215815
## 2:      world bank 2007       3         1        10         1 0.009168714
## 3:  brajovic et al 2015       0         1         0         1 0.001215815
```

## 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))

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)]),
    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 <- 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()
}

# 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]]$degree,
           degree.out = network_metrics[[i]]$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 c0c182c:
## [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                     turrall 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
## + 457/457 vertices, named, from 73e2020:
## [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
# NUMBER OF EDGES #####

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

## $food
## [1] 22
##
## $water
## [1] 431
# 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] 0.0008693552
# 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] 0.2157773
# PLOT NETWORK #####

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

# 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.
```

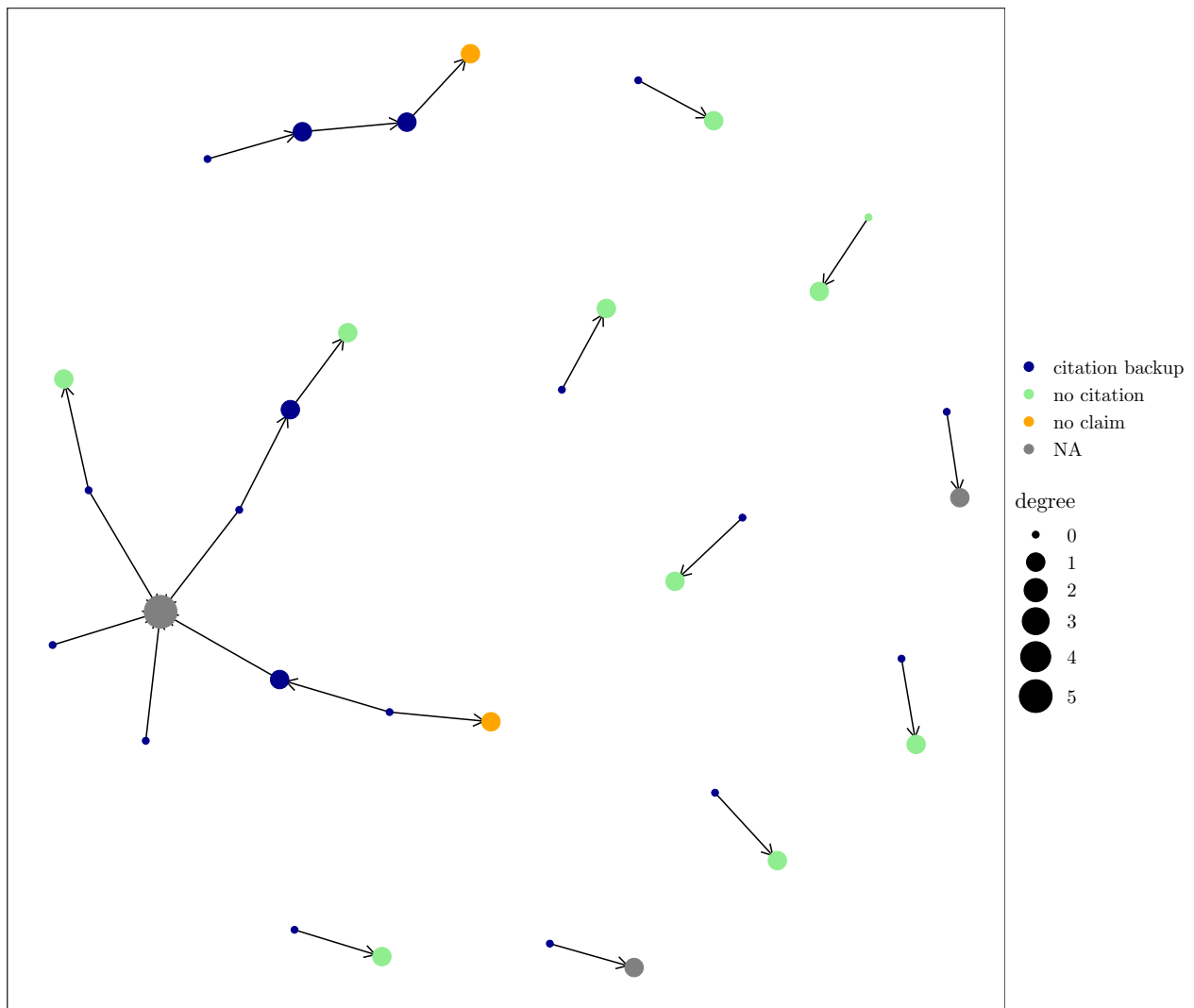
```
## i Please use `linewidth` in the `default_aes` field and elsewhere instead.
```

```
## This warning is displayed once every 8 hours.
```

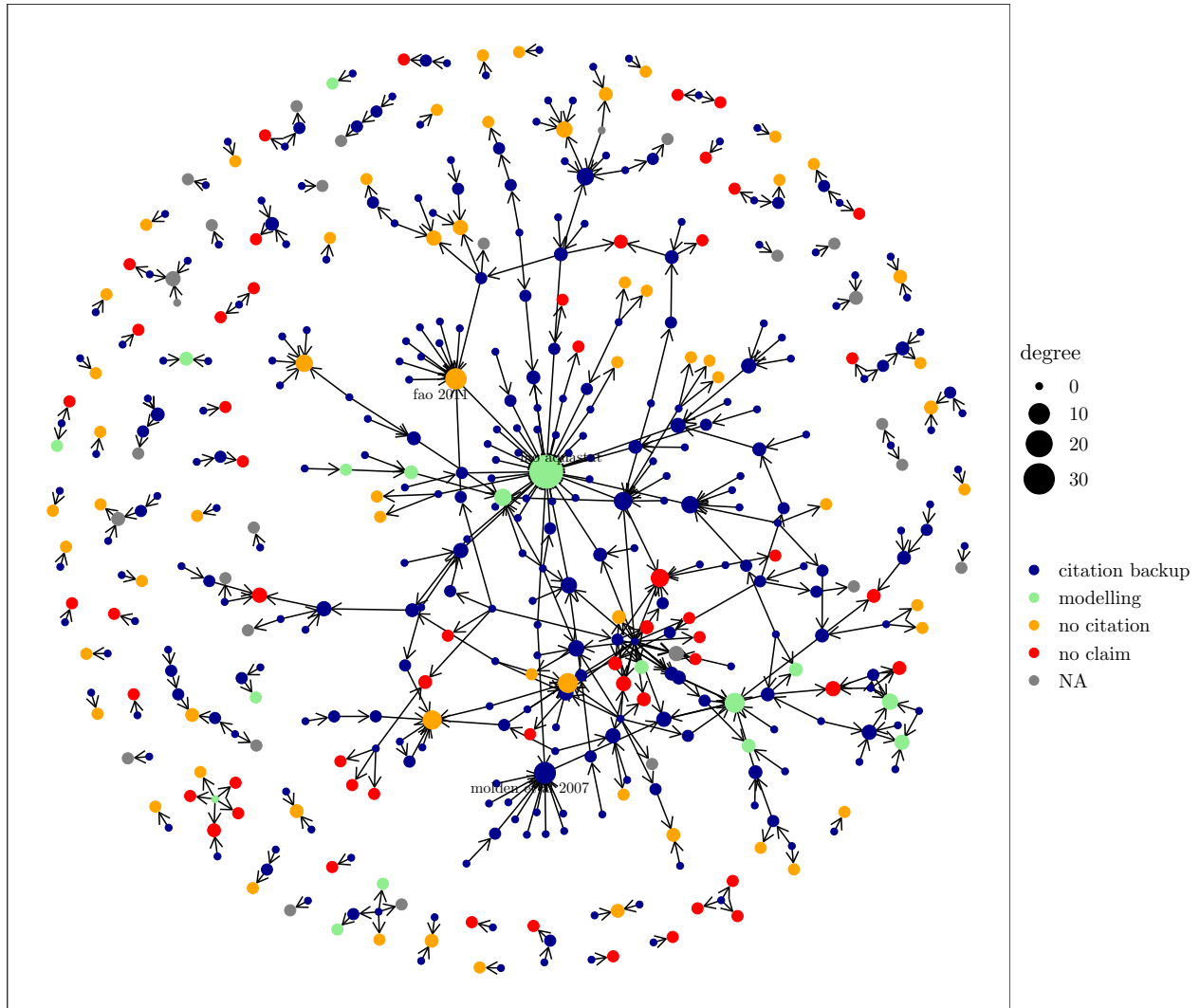
```
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
```

```
## generated.
```

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



```
##
## $water
## Warning: Removed 454 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) +

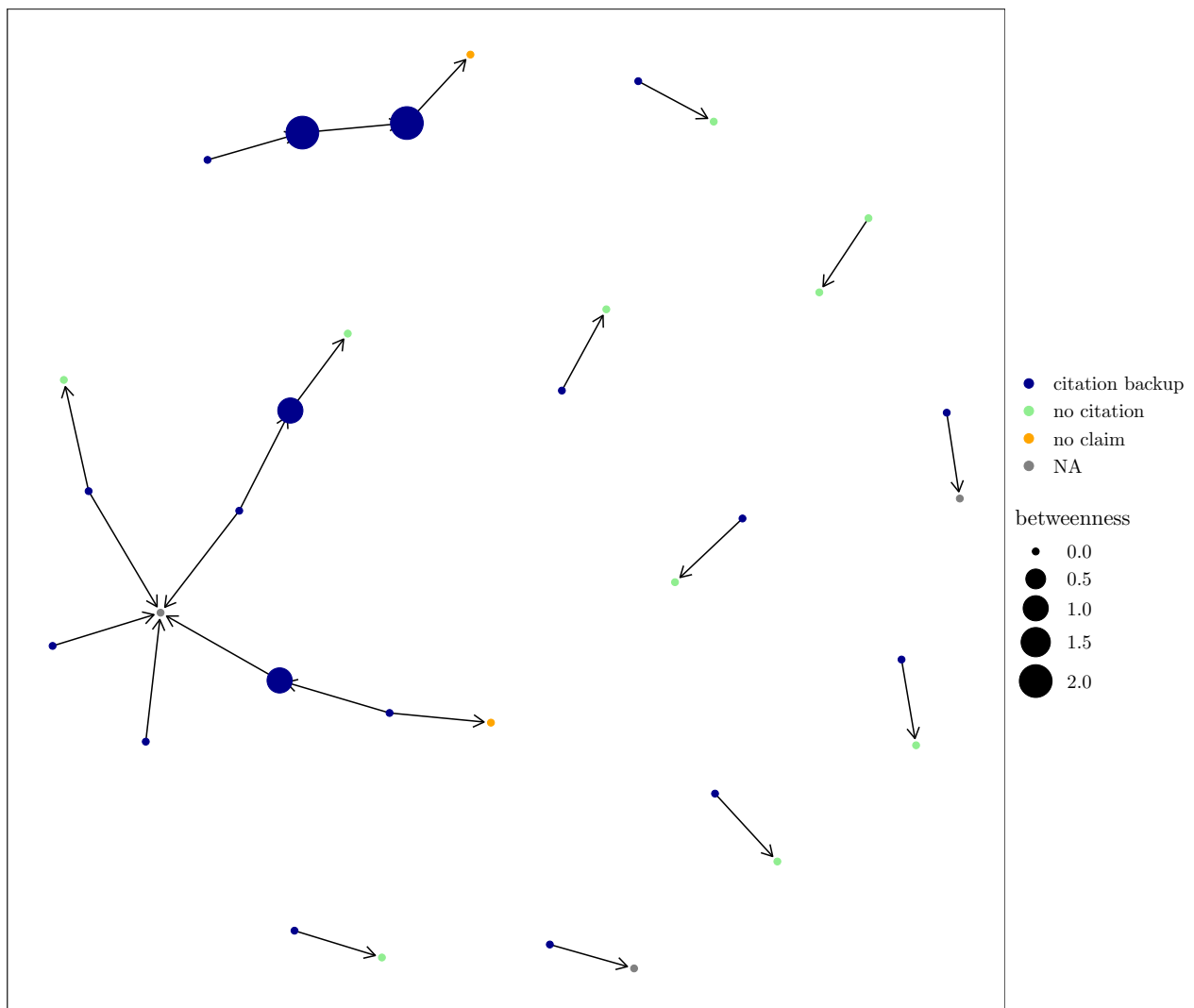
theme_AP() +
theme(axis.text.x = element_blank(),
      axis.ticks.x = element_blank(),
      axis.text.y = element_blank(),
      axis.ticks.y = element_blank(),
      legend.position = "right")
}

```

p2

```
## $food
```

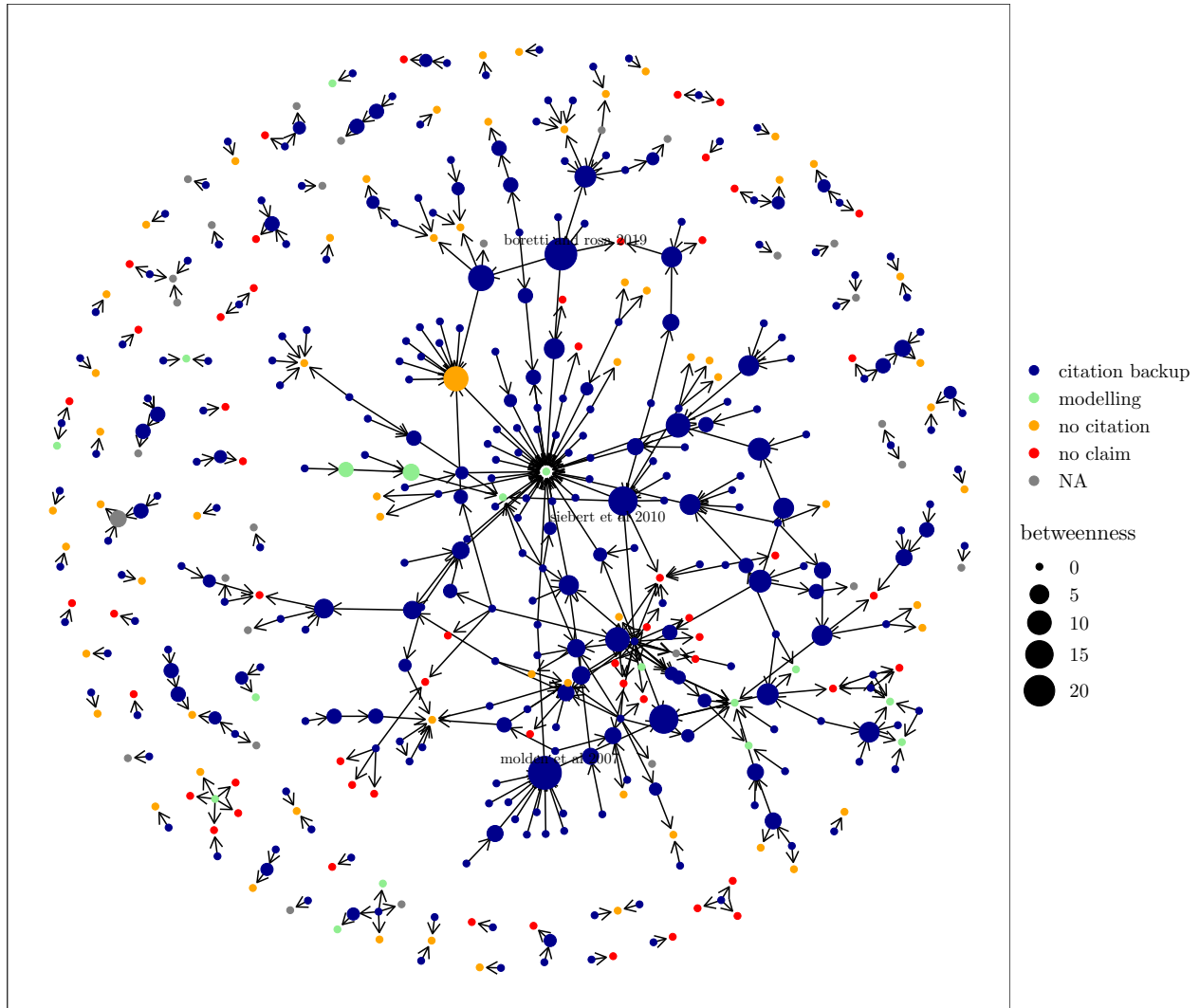
```
## Warning: Removed 33 rows containing missing values (`geom_text_repel()`).
```



```
##
```

```
## $water
```

```
## Warning: Removed 454 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 = "") +
```



```

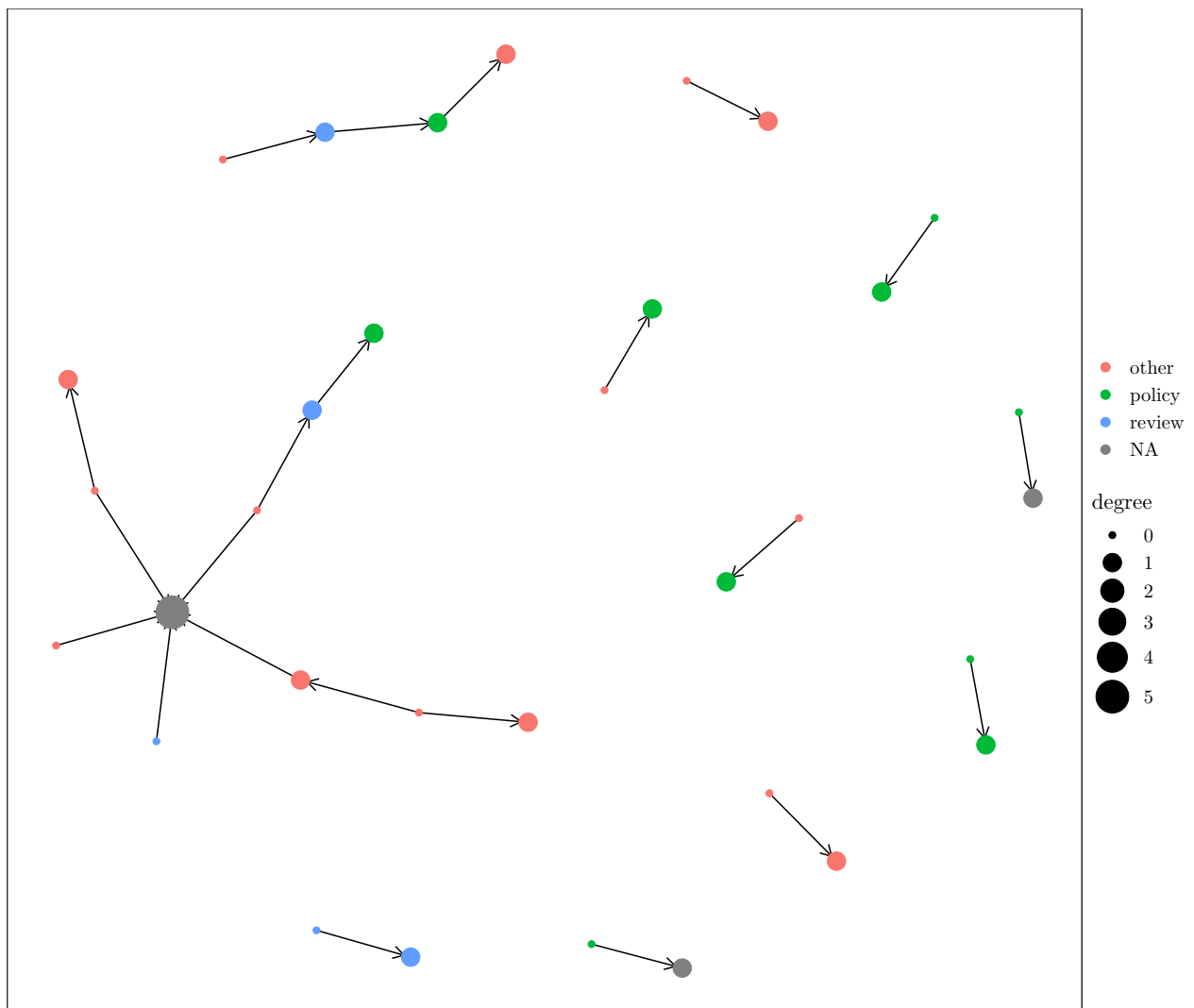
theme_AP() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "right")
}

```

p3

```
## $food
```

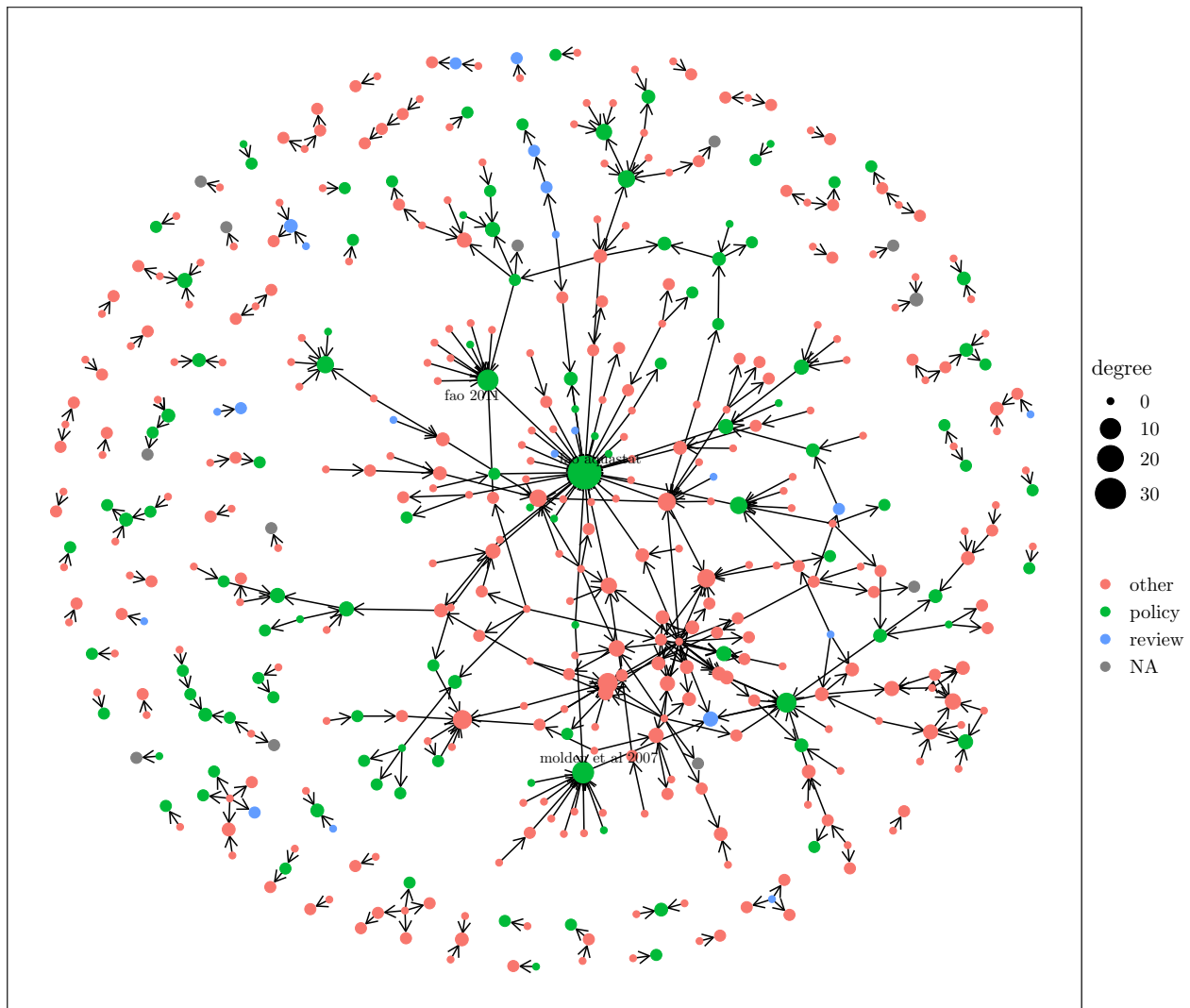
```
## Warning: Removed 33 rows containing missing values (`geom_text_repel()`).
```



```
##
```

```
## $water
```

```
## Warning: Removed 454 rows containing missing values (`geom_text_repel()`).
```



```
# 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(),
```

```

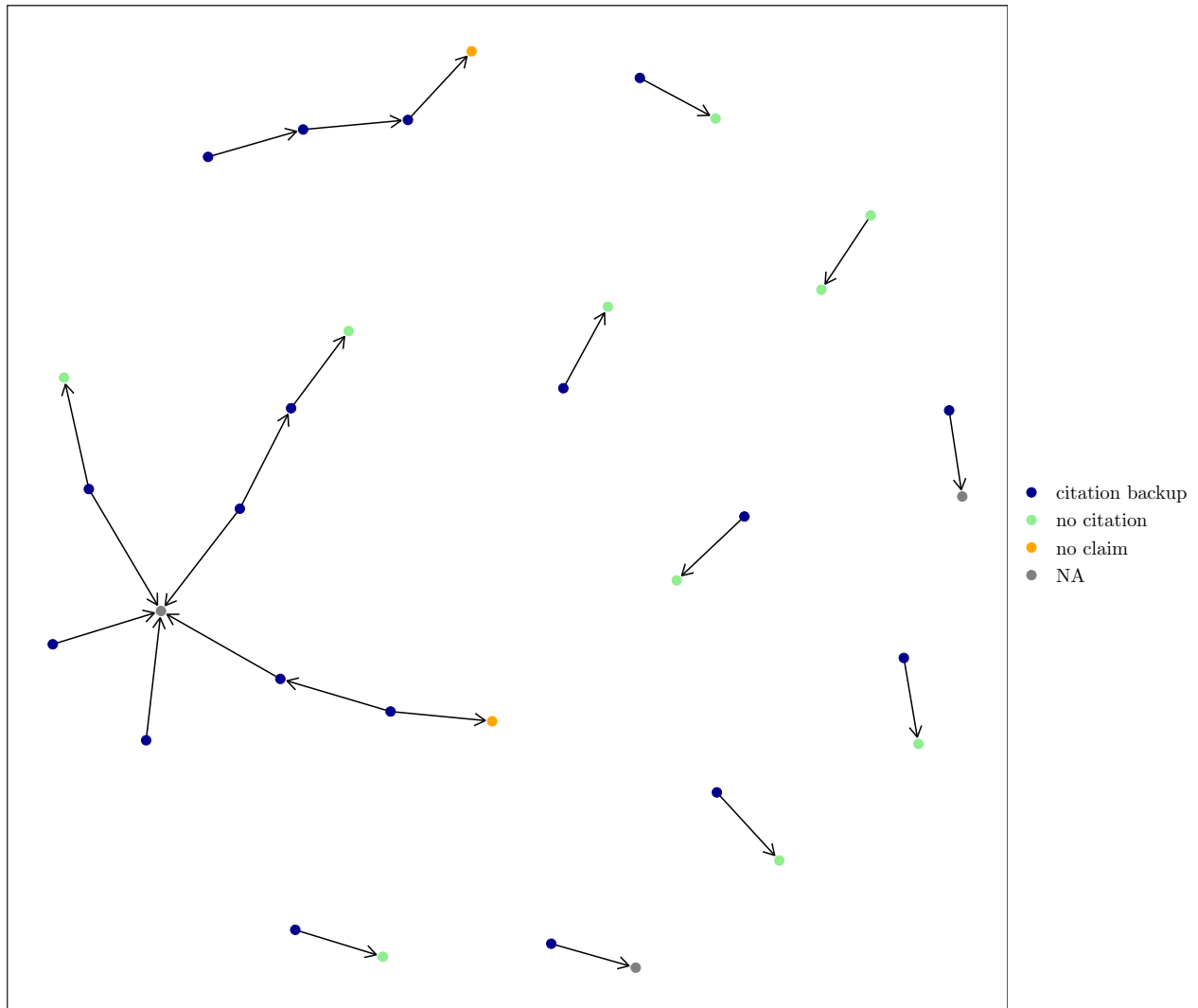
axis.ticks.x = element_blank(),
axis.text.y = element_blank(),
axis.ticks.y = element_blank(),
legend.position = "right")
}

```

p4

```
## $food
```

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



```
##
```

```
## $water
```

```
## Warning: Removed 440 rows containing missing values (`geom_text_repel()`).
```



```

pull(year)

# Substitute fao aquastat without year with the oldest aquastat citation -----
v_years[location_aquastat] <- oldest.aquastat.cite

# Find NA values -----

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

## [1] 14

# 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")

# PLOT NETWORK THROUGH TIME #####

# Set seed -----

set.seed(seed)

# 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 = 2.2) +
  scale_color_manual(name = "",
    values = selected_colors) +
  scale_x_continuous(name = "Year",
    limits = range(v_years),

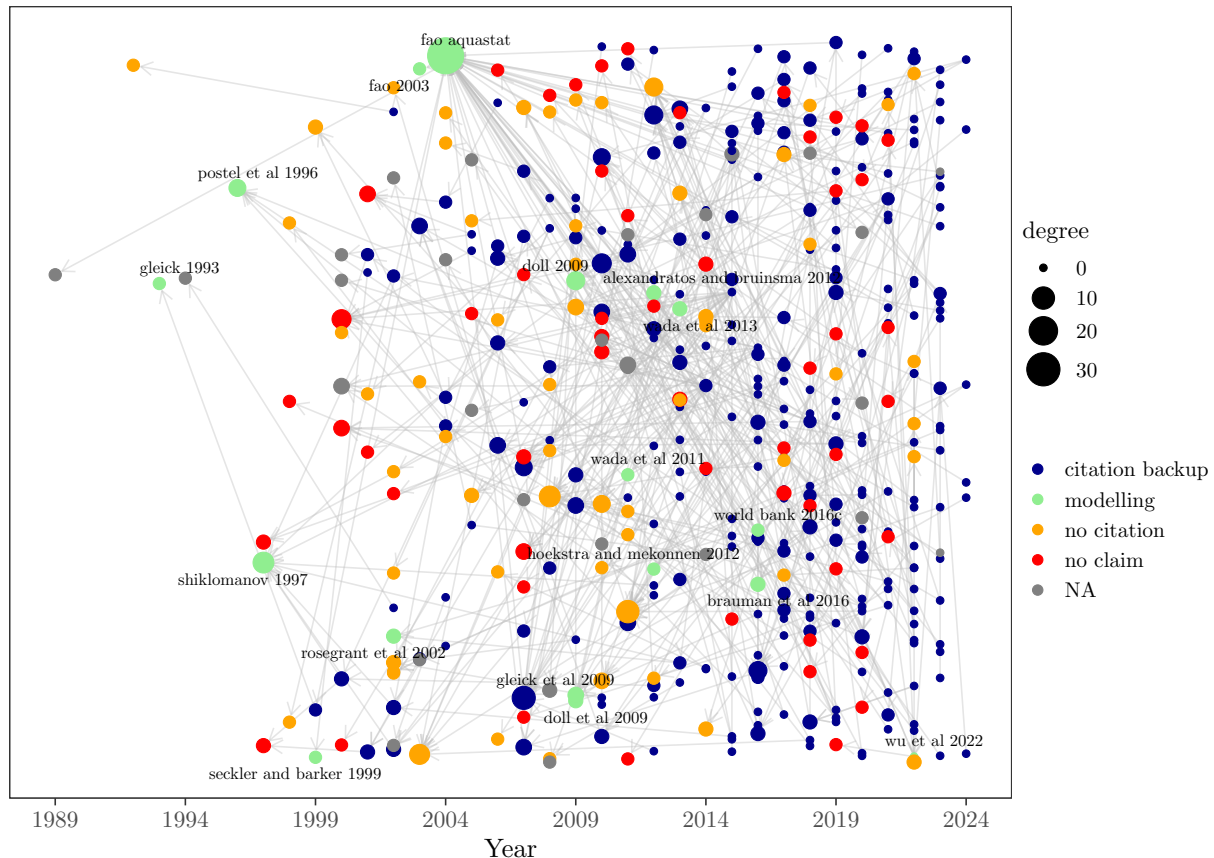
```

```

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 440 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 <- 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"      "tural 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] "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] "howden et al 2013"	"huang et al 2023"
## [45] "iwmi 2000"	"jaramillo and destouni 2015"
## [47] "johnson et al 2001"	"jury and vaux jr 2005"
## [49] "kaba gurmessa and assefa 2023"	"kabir et al 2023"
## [51] "karimi et al 2019"	"kaur saggi and jain 2022"
## [53] "kiani et al 2023"	"kilemo 2022"
## [55] "kumar dubey et al 2021"	"kumar ravi et al 2023"
## [57] "lalu et al 2024"	"lamastra et al 2014"
## [59] "liu and yang 2010"	"liu et al 2016"
## [61] "marston et al 2018"	"mcdermid et al 2023"
## [63] "meghan salmon et al 2015"	"mekonnen and hoekstra 2012"
## [65] "mekonnen et al 2015"	"mohanty et al 2018"
## [67] "moldovan et al 2022"	"nahar sumiya and khatun 2016"
## [69] "oladosu et al 2019"	"oladosu et al 2022"
## [71] "opio et al 2011"	"othmani et al 2021"
## [73] "ozdogan et al 2010b"	"payero et al 2006"
## [75] "pellegrini et al 2016"	"perry et al 2017"
## [77] "policy.1255933"	"policy.1435979"
## [79] "policy.1781691"	"policy.1874989"
## [81] "postel and vickers 2004"	"qin et al 2019"
## [83] "ran et al 2016"	"redhu and jain 2023"
## [85] "ren et al 2018"	"rockstrom et al 2007"
## [87] "rodriguez et al 2022"	"sadoff et al 2020"
## [89] "sahmat et al 2022"	"scanlon et al 2017"
## [91] "scanlon et al 2023"	"sepaskhah and ahmadi 2010"
## [93] "shiklomanov 2000"	"shtull-trauring et al 2016"
## [95] "siebert et al 2005"	"siebert et al 2010"
## [97] "siebert et al 2015"	"singh et al 2024"
## [99] "tabunshikov et al 2021"	"tsur 2005"
## [101] "turner 2008"	"unesco 2001"
## [103] "united nations 1998"	"united nations 2003"
## [105] "united nations 2021"	"united nations 2022"
## [107] "velez sanchez et al 2023"	"vorosmarty et al 2000"
## [109] "vorosmarty et al 2010"	"wada 2015"
## [111] "wada et al 2014"	"wada et al 2016"
## [113] "wajima 2018"	"walter et al 2017"
## [115] "wbcsd 2009"	"weatherhead and howden 2009"
## [117] "wisser et al 2010"	"wmo 1997"
## [119] "world bank 2001"	"world bank 2017"

```

## [121] "worldometers 2019"           "wri 2000"
## [123] "wu et al 2022"              "xu et al 2020"
## [125] "yilmazkuday et al 2021"     "yin et al 2022"
## [127] "young et al 2019"           "zeman et al 2006"
## [129] "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] "alan rotz 2020"
## [10] "alcamo et al 2007"
## [11] "alvarez et al 2004"
## [12] "anderson et al 2017"
## [13] "angaleeswari et al 2021"
## [14] "antia 2022"
## [15] "arboleda et al 2022"
## [16] "babel and wahid 2008"
## [17] "bac-dang et al 2019"
## [18] "bach et al 2017"
## [19] "badrul masud et al 2019"
## [20] "balyaminu 2017"
## [21] "barker 2015"
## [22] "baroni et al 2007"
## [23] "barreto and amaral 2018"
## [24] "basiri jahromi et al 2020"
## [25] "bhaskar and jain 2018"
## [26] "bicca rodrigues 2014"
## [27] "biemans et al 2011"
## [28] "biswas and tortajada 2010"
## [29] "bondeau et al 2007"
## [30] "bonsch et al 2016"
## [31] "boretti and rosa 2019"
## [32] "borin 2023"
## [33] "boucher et al 2004"
## [34] "bowden 2002"
## [35] "braimoh 2013"
## [36] "brar et al 2022"
## [37] "braun et al 2022"
## [38] "braune et al 2021"
## [39] "brillo 2022"
## [40] "brown 2008"
## [41] "brown 2009"

```

## [42] "cai and rosegant 2002"  
## [43] "caldera and breyer 2019"  
## [44] "calzadilla et al 2010"  
## [45] "carmona et al 2017"  
## [46] "carvalho 2019"  
## [47] "chai et al 2016"  
## [48] "chen et al 2018"  
## [49] "chilinda et al 2021"  
## [50] "chirone et al 2022"  
## [51] "clapp et al 2017"  
## [52] "coelho et al 2012"  
## [53] "connor 2017"  
## [54] "cristache et al 2018"  
## [55] "d'odorico et al 2019"  
## [56] "dalín et al 2012"  
## [57] "dave and nalco 2004"  
## [58] "de pascale et al 2011"  
## [59] "doll 2008"  
## [60] "doll et al 2014"  
## [61] "dounghmanee 2016"  
## [62] "droppers et al 2020"  
## [63] "dunkelman et al 2017"  
## [64] "eckert and kovalevska 2021"  
## [65] "elbakidze and cobourn 2014"  
## [66] "elmoneim badr et al 2021"  
## [67] "epri 2002"  
## [68] "evans and sadler 2008"  
## [69] "faiz alam et al 2023"  
## [70] "falkenmark 2013"  
## [71] "falkenmark et al 1997"  
## [72] "fao 2002"  
## [73] "fao 2002b"  
## [74] "fao 2007"  
## [75] "fao 2010"  
## [76] "fao 2011"  
## [77] "fao 2012"  
## [78] "fao 2012b"  
## [79] "fao 2017"  
## [80] "fao 2018"  
## [81] "fao 2019"  
## [82] "fao 2020"  
## [83] "fereres and soriano 2006"  
## [84] "firdayati et al 2022"  
## [85] "fitzgerald and auerbach 2016"  
## [86] "fogel and palmer 2014"  
## [87] "friha et al 2022"  
## [88] "gallardo 2015"  
## [89] "gan et al 2013"

## [90] "gavrilescu et al 2008"  
 ## [91] "gerbens-leenes and nonhebel 2004"  
 ## [92] "gerten et al 2007"  
 ## [93] "giordano 2007"  
 ## [94] "gleick and palaniappan 2010"  
 ## [95] "gleick et al 2002"  
 ## [96] "gleick et al 2011"  
 ## [97] "gleick et al 2014"  
 ## [98] "gleick et al 2018"  
 ## [99] "gorjian et al 2020"  
 ## [100] "gorjian et al 2022"  
 ## [101] "gourbesville 2008"  
 ## [102] "grigas et al 2023"  
 ## [103] "gumidyala et al 2020"  
 ## [104] "gurung 2016"  
 ## [105] "haddeland et al 2013"  
 ## [106] "hanasaki et al 2008"  
 ## [107] "hanasaki et al 2008b"  
 ## [108] "hannah 2017"  
 ## [109] "he et al 2023"  
 ## [110] "hegazi et al 2023"  
 ## [111] "hoekstra 2003"  
 ## [112] "hofste et al 2019"  
 ## [113] "hofwegen and svendsen 2000"  
 ## [114] "howden et al 2013"  
 ## [115] "huang et al 2023"  
 ## [116] "hussein bapir and wasman hamad 2023"  
 ## [117] "iaastd 2009"  
 ## [118] "ingrao et al 2023"  
 ## [119] "ipcc 2007"  
 ## [120] "iwmi 2000"  
 ## [121] "jagermeyr et al 2017"  
 ## [122] "jaramillo and destouni 2015"  
 ## [123] "jat et al 2016"  
 ## [124] "jehan et al 2022"  
 ## [125] "johnson et al 2001"  
 ## [126] "jury and vaux jr 2005"  
 ## [127] "kaba gurmessa and assefa 2023"  
 ## [128] "kabir et al 2023"  
 ## [129] "kapahi et al 2022"  
 ## [130] "karimi et al 2019"  
 ## [131] "kaur saggi and jain 2022"  
 ## [132] "khosravifar et al 2020"  
 ## [133] "kiani et al 2023"  
 ## [134] "kilemo 2022"  
 ## [135] "kiran kumara et al 2020"  
 ## [136] "kocian and incrocci 2020"  
 ## [137] "kumar dubey et al 2021"

## [138] "kumar ravi et al 2023"  
 ## [139] "kundzewicz et al. 2007"  
 ## [140] "lalu et al 2024"  
 ## [141] "lamastra et al 2014"  
 ## [142] "lang 2014"  
 ## [143] "legesse lebre et al 2021"  
 ## [144] "liu and yang 2010"  
 ## [145] "liu et al 2016"  
 ## [146] "lynch et al 2023"  
 ## [147] "maldonado junior et al 2019"  
 ## [148] "marston et al 2018"  
 ## [149] "mashnik et al 2017"  
 ## [150] "mcdermid et al 2023"  
 ## [151] "meghan salmon et al 2015"  
 ## [152] "mekonnen and hoekstra 2012"  
 ## [153] "mekonnen et al 2015"  
 ## [154] "mettetal 2019"  
 ## [155] "millenium ecosystem assessment 2005"  
 ## [156] "millenium project 2004"  
 ## [157] "mohanty et al 2018"  
 ## [158] "moldovan et al 2022"  
 ## [159] "molle 2002"  
 ## [160] "nahar sumiya and khatun 2016"  
 ## [161] "newell and taylor 2017"  
 ## [162] "no author"  
 ## [163] "nordin et al 2013"  
 ## [164] "norton-brandao et al 2013"  
 ## [165] "o'connell and billingsley 2020"  
 ## [166] "odeku 2020"  
 ## [167] "oecd 2010"  
 ## [168] "oecd 2017"  
 ## [169] "ohyama et al 2023"  
 ## [170] "oladosu et al 2019"  
 ## [171] "oladosu et al 2022"  
 ## [172] "opio et al 2011"  
 ## [173] "ostberg et al 2018"  
 ## [174] "othmani et al 2021"  
 ## [175] "ozdogan et al 2010"  
 ## [176] "ozdogan et al 2010b"  
 ## [177] "parameshwari 2017"  
 ## [178] "pastor et al 2019"  
 ## [179] "pauzuolien et al 2022"  
 ## [180] "payero et al 2006"  
 ## [181] "pedrero et al 2010"  
 ## [182] "pellegrini et al 2016"  
 ## [183] "perry et al 2017"  
 ## [184] "pfister and bayer 2013"  
 ## [185] "pokhrel et al 2012"

## [186] "pokhrel et al 2016"  
 ## [187] "policy.1094742"  
 ## [188] "policy.1252526"  
 ## [189] "policy.1255933"  
 ## [190] "policy.1257844"  
 ## [191] "policy.1381456"  
 ## [192] "policy.1435979"  
 ## [193] "policy.1666264"  
 ## [194] "policy.1781691"  
 ## [195] "policy.1874989"  
 ## [196] "policy.229461"  
 ## [197] "policy.240747"  
 ## [198] "policy.718260"  
 ## [199] "postel 2001"  
 ## [200] "postel and vickers 2004"  
 ## [201] "prochazka et al 2018"  
 ## [202] "qin et al 2019"  
 ## [203] "rahmadian and widyartono 2019"  
 ## [204] "ran et al 2016"  
 ## [205] "redhu and jain 2023"  
 ## [206] "ren et al 2018"  
 ## [207] "ricart and rico 2019"  
 ## [208] "ridgway et al 2019"  
 ## [209] "ridoutt et al 2009"  
 ## [210] "ringler et al 2022"  
 ## [211] "ritchie and roser 2017"  
 ## [212] "rivers et al 2015"  
 ## [213] "rockstrom and gordon 2001"  
 ## [214] "rockstrom et al 2007"  
 ## [215] "rodriguez et al 2022"  
 ## [216] "rodriguez-espinosa et al 2023"  
 ## [217] "romano et al 2023"  
 ## [218] "roseggrant and ringler 1998"  
 ## [219] "roseggrant et al 2009"  
 ## [220] "rost et al 2008"  
 ## [221] "roudi-fahim et al 2018"  
 ## [222] "sadoff et al 2020"  
 ## [223] "saeidian et al 2015"  
 ## [224] "sahmat et al 2022"  
 ## [225] "salman and salman 2002"  
 ## [226] "scanlon et al 2017"  
 ## [227] "scanlon et al 2023"  
 ## [228] "seckler et al 1998"  
 ## [229] "sepaskhah and ahmadi 2010"  
 ## [230] "shang et al 2024"  
 ## [231] "shiklomanov 1999"  
 ## [232] "shiklomanov 2000"  
 ## [233] "shiklomanov and rodde 2003"

## [234] "shtull-trauring et al 2016"  
 ## [235] "siebert and doll 2010"  
 ## [236] "siebert et al 2005"  
 ## [237] "siebert et al 2010"  
 ## [238] "siebert et al 2013"  
 ## [239] "siebert et al 2015"  
 ## [240] "singh et al 2024"  
 ## [241] "sophocleous 2004"  
 ## [242] "spiegel international 2009"  
 ## [243] "steduto et al 2018"  
 ## [244] "swatuk et al 2018"  
 ## [245] "tabunshikov et al 2021"  
 ## [246] "ti et al 2021"  
 ## [247] "tsiropoulos et al 2022"  
 ## [248] "tsur 2005"  
 ## [249] "tuninetti et al 2015"  
 ## [250] "turner 2008"  
 ## [251] "unctad 2011"  
 ## [252] "unep 2011"  
 ## [253] "unesco 2001"  
 ## [254] "unesco 2006"  
 ## [255] "unesco 2014"  
 ## [256] "unesco 2017"  
 ## [257] "united nations 1998"  
 ## [258] "united nations 2003"  
 ## [259] "united nations 2015"  
 ## [260] "united nations 2021"  
 ## [261] "united nations 2022"  
 ## [262] "united nations 2023"  
 ## [263] "velez sanchez et al 2023"  
 ## [264] "velpuri et al 2009"  
 ## [265] "vorosmarty et al 2000"  
 ## [266] "vorosmarty et al 2005"  
 ## [267] "vorosmarty et al 2010"  
 ## [268] "wada 2015"  
 ## [269] "wada et al 2013b"  
 ## [270] "wada et al 2014"  
 ## [271] "wada et al 2016"  
 ## [272] "wajima 2018"  
 ## [273] "walter et al 2017"  
 ## [274] "wbcsd 2009"  
 ## [275] "weatherhead and howden 2009"  
 ## [276] "williams et al 2017"  
 ## [277] "wilson 2013"  
 ## [278] "wisser et al 2008"  
 ## [279] "wisser et al 2010"  
 ## [280] "wmo 1997"  
 ## [281] "world bank 1992"

```
## [282] "world bank 2001"
## [283] "world bank 2017"
## [284] "world bank 2021"
## [285] "world watch institute 2004"
## [286] "world water assessment programme 2003"
## [287] "world water assessment programme 2014"
## [288] "worldometers 2019"
## [289] "wri 2000"
## [290] "wu et al 2022"
## [291] "wwap 2018"
## [292] "wwf 2006"
## [293] "xing yuan et al 2024"
## [294] "xu et al 2020"
## [295] "yilmazkuday et al 2021"
## [296] "yin et al 2022"
## [297] "young et al 2019"
## [298] "zeman et al 2006"
## [299] "zhao et al 2022"
## [300] "zhuo et al 2022"
```

```
# 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] 0.2822757
##
## $water$path ending in no claim or no citation`
## [1] 0.6564551
```



## 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      httptr_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
```