The treatment of uncertainties in global water models

R code

Arnald Puy

Contents

1 Preliminary functions

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

# Function to read in all required packages in one go
loadPackages <- function(x) {
  for(i in x) {
    if(!require(i, character.only = TRUE)) {
      install.packages(i, dependencies = TRUE)
      library(i, character.only = TRUE)
    }
  }
}

# Load the packages
loadPackages(c(
  "bibliometrix", "tidyverse", "data.table", "scales", "pdfsearch", "pdftools",
  "openxlsx", "cowplot", "wesanderson", "sjmisc", "ggpubr", "tm", "syuzhet",
  "qdapRegex", "tidytext", "igraph", "ggraph"))

# Create custom theme
theme_AP <- function() {
  theme_bw() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          legend.background = element_rect(fill = "transparent",
                                           color = NA),
          legend.key = element_rect(fill = "transparent",
                                    color = NA),
          strip.background = element_rect(fill = "white"),
          legend.margin = margin(0.5, 0.1, 0.1, 0.1),
          legend.box.margin = margin(0.2,-2,-7,-7))
}

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

checkpoint("2022-05-30",
           R.version ="4.2.0",
           checkpointLocation = getwd())
```

```
# FUNCTION TO CLEAN TEXT ###########################################################

# Function to remove punctuation, citations, numbers, stopwords in english,
# bring to lowercase and strip whitespace, and especial characters, etc...
clear_text <- function(x) {
  y <- gsub("-", "", x)
```

```
  y <- rm_citation(y)
  y <- tm::removePunctuation(y)
  y <- tm::removeNumbers(y)
  y <- tm::removeWords(y, stopwords::stopwords(language = "en"))
  y <- tolower(y)
  y <- str_replace_all(y, "[[:punct:]]", "") # Remove punctuation characters
  y <- y <- str_remove_all(y, "[^[\\da-zA-Z ]]")# Remove all non-alphanumerical
  y <- stemDocument(y) # Stem the document and keep only the root of the word
  y <- tm::stripWhitespace(y)
  y <- str_squish(y)
  y <- tm::removeWords(y, c(" et ", "al", "table", "figure", "fig",
                            "figs", "can", "eg", "mm", "yr",
                            "last", "access", "see", "section"))
  y <- gsub(" ?doi\\w+ ?", "", y) # Remove words that start with doi
  y <- str_replace(y, "http", "") # Remove https
  y <- tm::removeWords(y, stopwords::stopwords(language = "en"))
  y <- trimws(y) # Remove leading/trailing white space
  y <- tm::stripWhitespace(y)
  y <- gsub("\\s[A-Za-z](?= )", "", y, perl = TRUE) # Remove isolated letters
  y <- gsub("\\s[A-Za-z]$", "", y, perl = TRUE) # Remove isolated letters end of string
  y <- str_squish(y)

  return(y)
}
```

2 Models under study

```
# VECTOR WITH NAME OF MODELS ########################################################

models <- c("WaterGAP", "PCR-GLOBWB", "MATSIRO", "H08", "JULES-W1", "MPI-HM",
            "MHM", "LPJmL", "CWatM", "CLM", "DBHM", "ORCHIDEE", "GR4J")

models_vec <- paste(models, "_ref.bib", sep = "")
```

3 Bibliometric analysis

```
# BIBLIOMETRIC ANALYSIS #############################################################

output <- results <- years <- journals <- dt <- dt.clean <- list()

selected_cols <- c("title", "abstract", "keywords", "keywords.plus")

for (i in 1:length(models_vec)) {

  output[[i]] <- convert2df(file = models_vec[i],
                            dbsource = "wos",
                            format = "bibtex")

  # Extract title ----------------------------------------------------------

  title <- output[[i]]$TI

  # Extract Authors, Countries and Universities ----------------------------

  # Authors
  tmp.authors <- output[[i]]$AU
  first.author <- sub(" *\\;.*", "", tmp.authors)
  last.author <- sub(".*\\;","", tmp.authors)

  # First author affiliation and country
  country.first <- sub(".*\\,", "", output[[i]]$RP)
  university.first <- sub(" *\\;.*", "", output[[i]]$affiliations)

  # Last author affiliation and country
  last.affiliation <- sub(".*\\;", "", output[[i]]$C1)
  country.last <- sub("\\.", "", sub(".*\\, ", "", last.affiliation))
  university.last <- sub(".*\\;", "", output[[i]]$affiliations)

  # Extract keywords -------------------------------------------------------

  keywords <- gsub(";;", ";", output[[i]]$DE)
```

```
  keywords.plus <- gsub(";;", ";", output[[i]]$ID)

  # Create data.table ---------------------------------------------------------

  dt[[i]] <- data.table("WOS" = output[[i]]$UT,
                        "title" = title,
                        "year" = output[[i]]$PY,
                        "keywords" = keywords,
                        "keywords.plus" = keywords.plus,
                        "first.author" = first.author,
                        "last.author" = last.author,
                        "country.first" = country.first,
                        "country.last" = country.last,
                        "university.first" = university.first,
                        "university.last" = university.last,
                        "abstract" = output[[i]]$AB)

  dt.clean[[i]] <- copy(dt[[i]])

  dt.clean[[i]][, (selected_cols):= lapply(.SD, function(x)
    clear_text(x)), .SDcols = selected_cols]

  # Export data dirty and clean
  write.xlsx(dt[[i]], file = paste(models[i], "_bibliometric.xlsx", sep = ""))
  write.xlsx(dt.clean[[i]], file = paste(models[i], "_bibliometric_clean.xlsx", sep = ""))

  # Retrieve analysis bibliometrix --------------------------------------------

  results[[i]] <- biblioAnalysis(output[[i]], sep = ";")
  years[[i]] <- data.table(results[[i]]$Years)
  journals[[i]] <- data.table(results[[i]]$Sources) %>%
    .[, SO:= str_to_title(SO)]
}

# Fill out affiliations erroneously labelled as NA ------------------------------

# Watergap (1)
for(i in c(1, 4, 5)) {
  output[[1]]$affiliations[[i]] <- "UNIVERSITAT KASSEL"
}


# Add names of models -------------------------------------------------------
names(years) <- models
names(journals) <- models
names(dt.clean) <- models
names(dt.clean) <- models
```

4 Keywords analysis: "uncertainty" and "sensitivity"

```
# KEYWORDS ANALYSIS ###############################################################

# Define vectors for search ---------------------------------------------------
directory <- "/Users/arnaldpuy/Documents/papers/ghms_bibliometric/"
directory_vec <- paste(directory, models, "_pdfs", sep = "")
filename_keywords <- paste(models, "keywords", sep = "_")

# Define vectors with keywords ------------------------------------------------
keywords_vec <- c("uncertainty", "sensitivity")
keywords_vec_stemmed <- stemDocument(keywords_vec)

# Loop ------------------------------------------------------------------------
dt.keyword <- dt.keyword.clean <- output <- list()
for (i in 1:length(directory_vec)) {

  output[[i]] <- keyword_directory(directory_vec[i],
                                   keyword = keywords_vec_stemmed,
                                   split_pdf = TRUE)

  dt.keyword[[i]] <- data.table("name" = output[[i]]$pdf_name,
                                "keyword" = output[[i]]$keyword,
                                "text" = output[[i]]$line_text)

  dt.keyword.clean[[i]] <- copy(dt.keyword[[i]])

  # Clean the text where the keywords are located
  dt.keyword.clean[[i]] <- dt.keyword.clean[[i]][, text:= clear_text(text)]

  # Write dirty and clean data
  fwrite(dt.keyword[[i]], file = paste(filename_keywords[i], ".csv", sep = ""))
  fwrite(dt.keyword.clean[[i]], file = paste(filename_keywords[i], "_clean.csv", sep = ""))
}

names(output) <- models
names(dt.keyword) <- models
names(dt.keyword.clean) <- models
```

5 Arrange the data

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

# Bibliometric analysis -------------------------

# Correct for USA and China
```

```
colsName <- c("country.first", "country.last")
full.dt <- rbindlist(dt.clean, idcol = "Model") %>%
  .[, (colsName):= lapply(.SD, function(x)
    ifelse(grepl("USA", x), "USA", x)), .SDcols = colsName] %>%
  .[, (colsName):= lapply(.SD, function(x)
    ifelse(grepl("CHINA", x), "CHINA", x)), .SDcols = colsName]

# Export
fwrite(full.dt, "full.dt.csv")

# Keywords analysis ----------------------------
full.keyword.dt <- rbindlist(dt.keyword.clean, idcol = "Model")

# Export
fwrite(full.keyword.dt, "full.keyword.dt.csv")
```

6 Descriptive analysis

```
# DESCRIPTIVE STUDY ##################################################################

# Total number of studies
total.n <- full.dt[, .(Model, WOS)] %>%
  .[, .(total.papers = .N), Model] %>%
  .[order(-total.papers)]

total.n
```

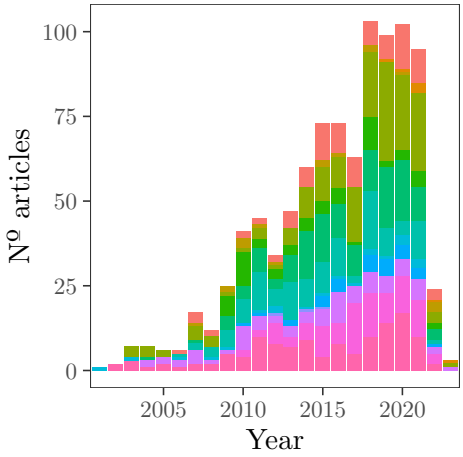
```
##          Model total.papers
##  1:       GR4J          167
##  2:   JULES-W1          136
##  3:   WaterGAP          126
##  4:      LPJmL          116
##  5: PCR-GLOBWB           95
##  6:        CLM           92
##  7:   ORCHIDEE           75
##  8:        H08           61
##  9:        MHM           29
## 10:    MATSIRO           21
## 11:       DBHM           17
## 12:      CWatM            7
## 13:     MPI-HM            3
```

```
sum(total.n$total.papers)
```

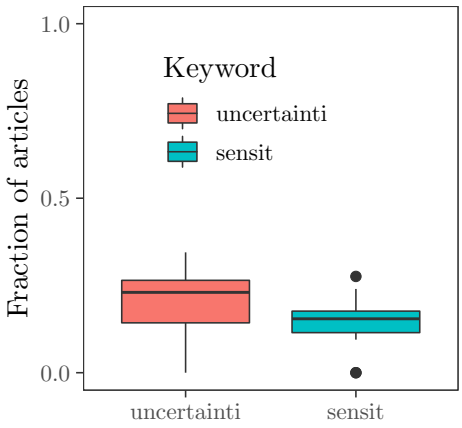
```
## [1] 945
```

```
# NUMBER OF STUDIES THROUGHT TIME ################################################
```

```
plot.time <- rbindlist(years, idcol = "Model")[, .N, .(V1, Model)] %>%
  .[, V1:= as.factor(V1)] %>%
  ggplot(., aes(V1, N, fill = Model)) +
  geom_col() +
  scale_x_discrete(breaks = pretty_breaks(n = 3)) +
  labs(x = "Year", y = "Nº articles") +
  theme_AP() +
  theme(legend.position = "none")

plot.time
```




```
# FRACTION OF STUDIES WITH UNCERTAINTI AND SENSIT* IN THE ABSTRACT ##############

full.dt <- full.dt[, `:=` (uncertainti = str_detect(abstract, keywords_vec_stemmed[1]),
                           sensit = str_detect(abstract, keywords_vec_stemmed[2]))]

plot.n.keywords <- full.dt[, lapply(.SD, function(x)
  sum(x) / .N), .SDcols = (keywords_vec_stemmed), Model] %>%
  melt(., measure.vars = keywords_vec_stemmed) %>%
  ggplot(., aes(variable, value, fill = variable)) +
  geom_boxplot() +
  labs(y = "Fraction of articles", x = "") +
  scale_y_continuous(breaks = pretty_breaks(n = 3),
                     limits = c(0, 1)) +
  scale_fill_discrete(name = "Keyword") +
  theme_AP() +
  theme(legend.position = c(0.45, 0.75))

plot.n.keywords
```

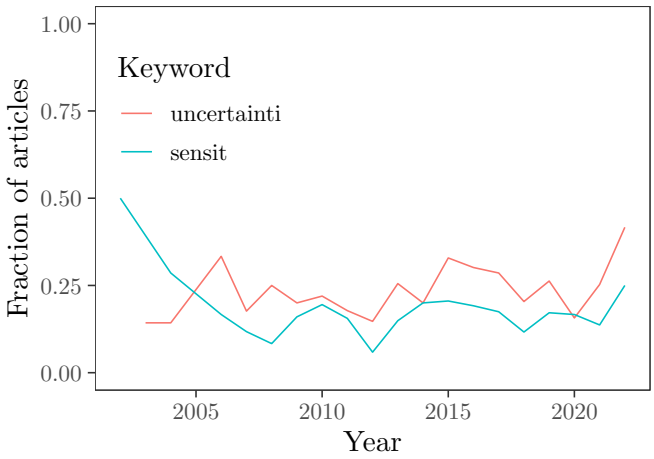



```
# Fraction of studies with both keywords in the abstract
full.dt[uncertainti == "TRUE" & sensit == "TRUE", .N] / full.dt[, .N]
```

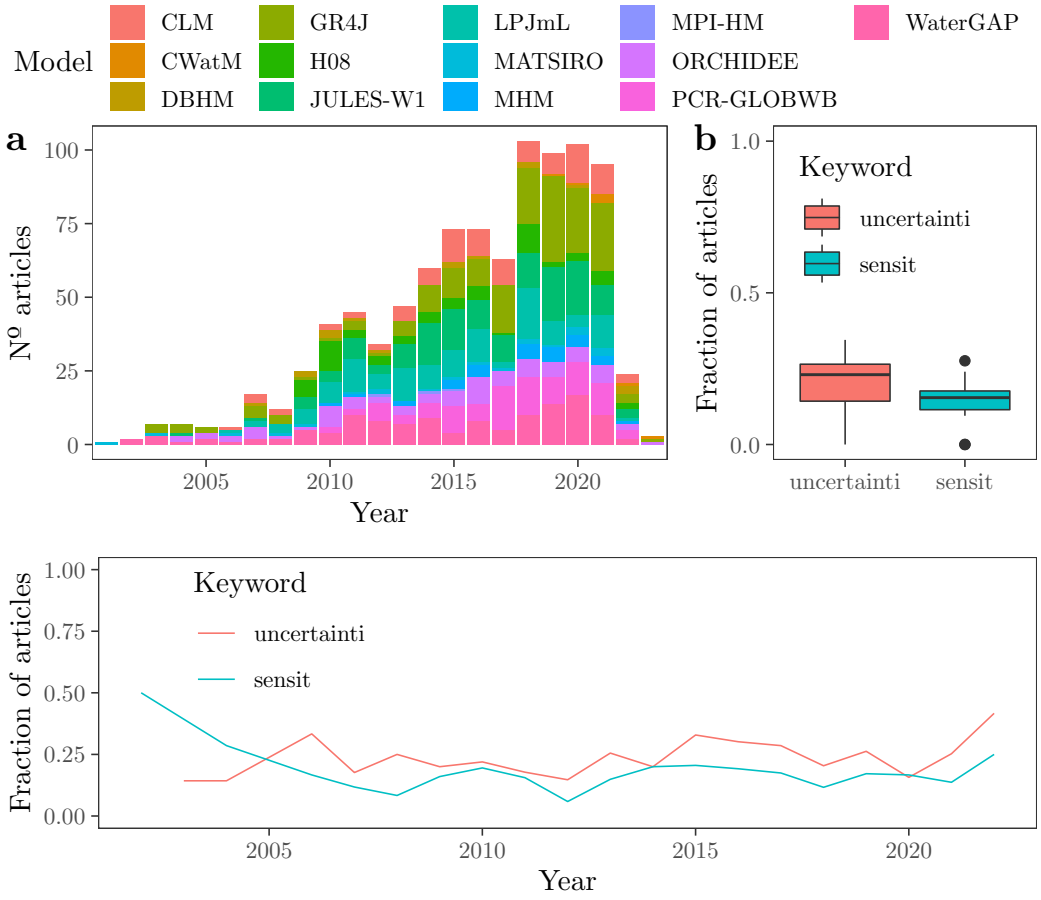
```
## [1] 0.06137566
```

```
# FRACTION OF STUDIES WITH KEYWORDS THROUGH TIME ################################

total.n.year <- rbindlist(years, idcol = "Model") %>%
  .[, .(total.n = .N), V1] %>%
  setnames(., "V1", "year")

plot.fraction.years <- full.dt[, .(WOS, uncertainti, sensit, year)] %>%
  melt(., measure.var = keywords_vec_stemmed) %>%
  .[value == TRUE, .N, .(year, variable)] %>%
  merge(., total.n.year, by = "year") %>%
  .[, fraction:= N / total.n] %>%
  ggplot(., aes(year, fraction, color = variable, group = variable)) +
  geom_line() +
  scale_color_discrete(name = "Keyword") +
  scale_y_continuous(limits = c(0, 1)) +
  labs(x = "Year", y = "Fraction of articles") +
  theme_AP() +
  theme(legend.position = c(0.2, 0.75))

plot.fraction.years
```

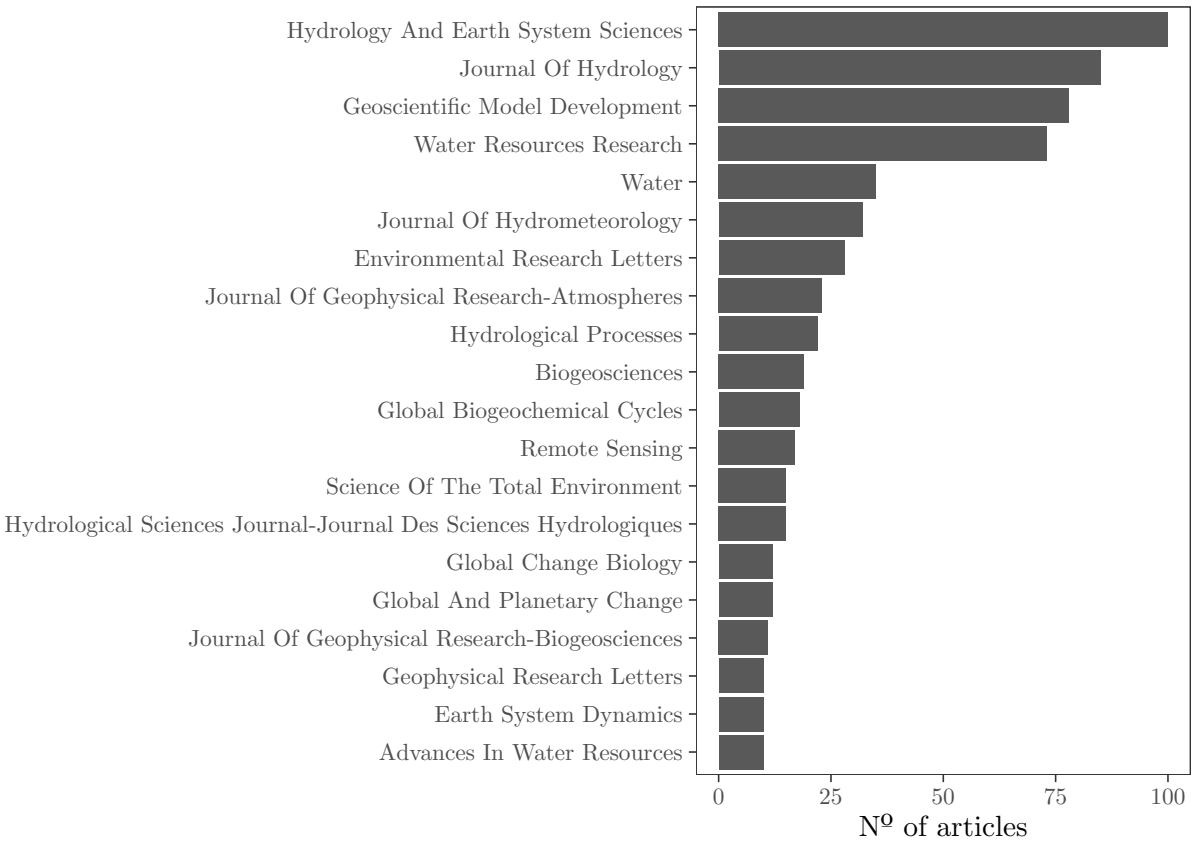
```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```




```
# MERGE DESCRIPTIVE PLOTS ###########################################################

legend <- get_legend(plot.time + theme(legend.position = "top"))
top <- plot_grid(plot.time, plot.n.keywords, ncol = 2, labels = "auto",
                 rel_widths = c(0.65, 0.35))
all <- plot_grid(legend, top, ncol = 1, rel_heights = c(0.22, 0.78))
plot_grid(all, plot.fraction.years,
          ncol = 1, labels = "", rel_heights = c(0.6, 0.4))
```



```
# PLOT JOURNALS ######################################################################

rbindlist(journals, idcol = "Model") %>%
  .[, sum(N), SO] %>%
  .[order(-V1)] %>%
  .[1:20] %>%
  na.omit() %>%
  ggplot(., aes(reorder(SO, V1, sum), V1)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(x = "", y = "Nº of articles") +
  theme_AP()
```



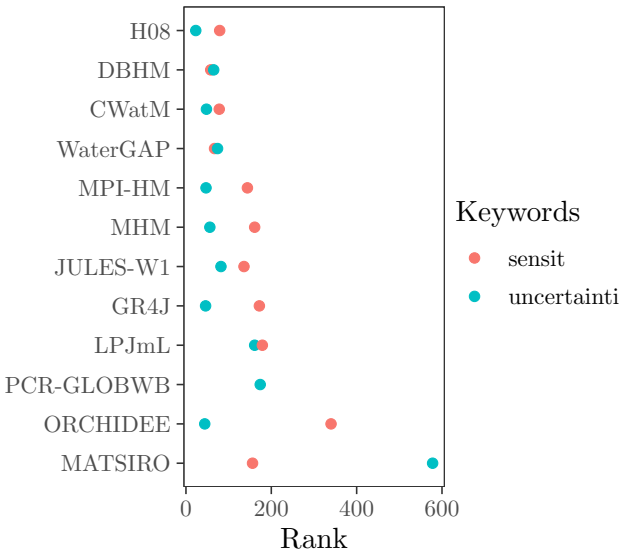

```
# WORDCLOUD OF WORDS IN ABSTRACT ###################################################

tmp <- split(full.dt, full.dt$Model)
names(tmp) <- models

out <- dtm <- m <- v <- word.count <- list()
for (i in names(tmp)) {
  out[[i]] <- Corpus(VectorSource(tmp[[i]]$abstract))
  dtm[[i]] <- tm::TermDocumentMatrix(out[[i]])
  m[[i]] <- as.matrix(dtm[[i]])
  v[[i]] <- sort(rowSums(m[[i]]), decreasing=TRUE)
```

```
  word.count[[i]] <- data.table(word = names(v[[i]]), freq = v[[i]])
}

word.count.dt <- rbindlist(word.count, idcol = "Model")

# Plot wordcloud ------------------------------------------

plots.wordcloud <- list()

for(i in names(word.count)) {
  plots.wordcloud[[i]] <- word.count.dt[Model == i] %>%
    .[1:50] %>%
    ggplot(., aes(label = word, size = freq)) +
    ggwordcloud::geom_text_wordcloud_area(eccentricity = 1, shape = "square") +
    scale_size_area(max_size = 10) +
    theme_AP() +
    ggtitle(names(word.count[i]))
}

# Check rank of the terms "uncertainty" and "sensitivity" in the abstract -----

word.count.dt[, rank:= frank(-freq, ties.method = "first"), Model]

rank.keywords <- word.count.dt[word %chin% keywords_vec_stemmed] %>%
  merge(., total.n, by = "Model")

rank.keywords[order(word, rank)]
```

```
##          Model        word freq rank total.papers
##  1:       DBHM      sensit   38   58           17
##  2:   WaterGAP      sensit   35   67          126
##  3:      CWatM      sensit   12   78            7
##  4:        H08      sensit   53   79           61
##  5:   JULES-W1      sensit   12  136          136
##  6:     MPI-HM      sensit   31  144            3
##  7:    MATSIRO      sensit    4  156           21
##  8:        MHM      sensit   26  161           29
##  9:       GR4J      sensit   26  172          167
## 10:      LPJmL      sensit    4  179          116
## 11:   ORCHIDEE      sensit   11  340           75
## 12:        H08 uncertainti  137   23           61
## 13:   ORCHIDEE uncertainti   52   44           75
## 14:       GR4J uncertainti   64   46          167
## 15:     MPI-HM uncertainti   68   47            3
## 16:      CWatM uncertainti   17   48            7
## 17:        MHM uncertainti   52   56           29
## 18:       DBHM uncertainti   34   65           17
```

```
## 19:   WaterGAP uncertainti   33   74          126
## 20:   JULES-W1 uncertainti   17   82          136
## 21:      LPJmL uncertainti    5  161          116
## 22: PCR-GLOBWB uncertainti    2  174           95
## 23:    MATSIRO uncertainti    1  578           21
##          Model        word freq rank total.papers
```

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

ggplot(rank.keywords, aes(reorder(Model, -rank), rank, color = word)) +
  geom_point() +
  coord_flip() +
  labs(x = "", y = "Rank") +
  scale_color_discrete(name = "Keywords") +
  theme_AP() +
  theme(legend.position = "right")
```



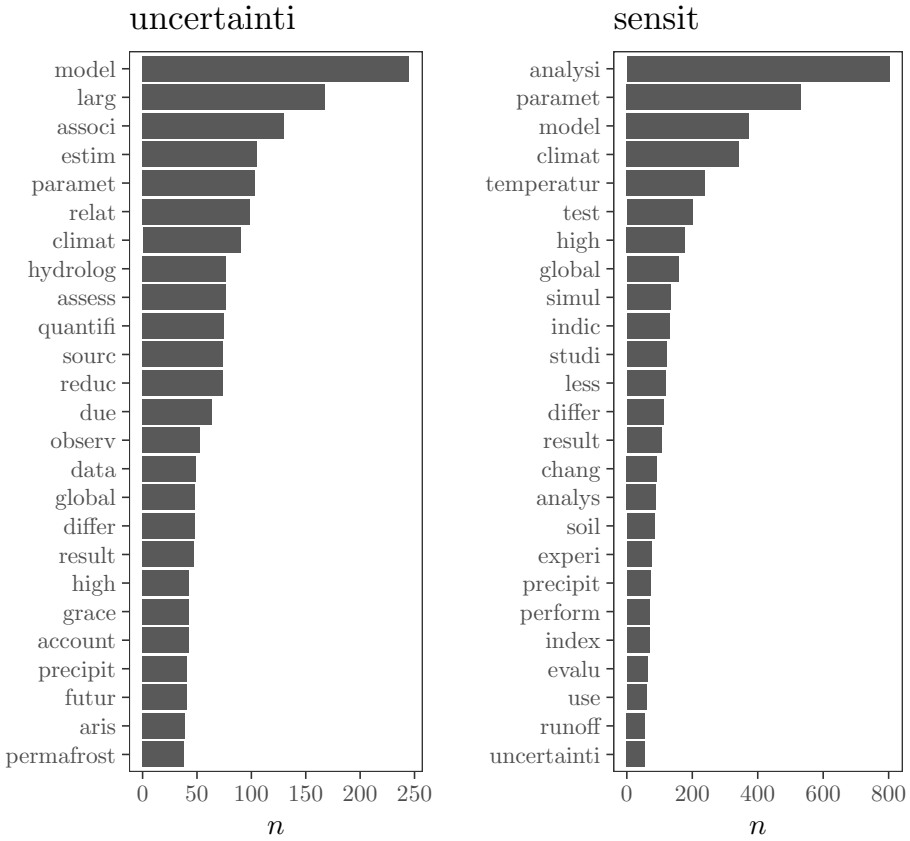
7 Study of n-tokens

```
# STUDY OF N-TOKENS ##############################################################

# Number of tokens ----------------------------
N.tokens <- 2

# For loop ------------------------------------
output <- token.analysis <- vec <- plot.token <-
  plot.token.model <- graph_plot <- list()

for (i in 1:length(keywords_vec)) {

  output[[i]] <- full.keyword.dt %>%
```

```
  .[keyword == keywords_vec_stemmed[i]]

# Token analysis ------------------------------------------------------------

token.analysis[[i]] <- output[[i]] %>%
  unnest_tokens(bigram, text, token = "ngrams", n = N.tokens) %>%
  separate(bigram, into = c("word1", "word2"), sep = " ") %>%
  # We count the co-occurences of words without taking into account their order
  # within the n-token
  .[, `:=`(word1= pmin(word1, word2), word2 = pmax(word1, word2))] %>%
  count(word1, word2, Model, sort = TRUE) %>%
  unite(., col = "bigram", c("word1", "word2"), sep = " ")

vec[[i]] <- token.analysis[[i]] %>%
  .[, str_detect(bigram, keywords_vec_stemmed[i])]

plot.token[[i]] <-   token.analysis[[i]][vec[[i]]] %>%
  .[, sum(n), bigram] %>%
  .[order(-V1)] %>%
  .[, head(.SD, 25)] %>%
  .[, bigram:= str_remove(bigram, keywords_vec_stemmed[i])] %>%
  ggplot(., aes(reorder(bigram, V1, sum), V1)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_AP() +
  labs(y = "$n$", x = "") +
  theme(legend.position = "none") +
  ggtitle(keywords_vec_stemmed[i])

plot.token.model[[i]] <- token.analysis[[i]][vec[[i]]] %>%
  .[, head(.SD, 5), Model] %>%
  .[, `:=` (bigram = str_remove(bigram, keywords_vec_stemmed[i]),
            Model = as.factor(Model))] %>%
  .[, bigram:= reorder_within(bigram, n, Model)] %>%
  ggplot(., aes(reorder(bigram, n, sum), n, fill = Model)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_AP() +
  labs(y = "$n$", x = "") +
  scale_x_reordered() +
  theme(legend.position = "none") +
  ggtitle(keywords_vec_stemmed[i]) +
  facet_wrap(~Model, scales = "free", ncol = 3)

# Graph analysis ------------------------------------------------------------

bigram_graph <- token.analysis[[i]] %>%
```

```
    separate(., col = "bigram", into = c("word1", "word2"), sep = " ") %>%
    .[n > 20] %>%
    graph_from_data_frame()

  set.seed(666)

  a <- grid::arrow(type = "closed", length = unit(.08, "inches"))

  graph_plot[[i]] <- ggraph(bigram_graph, layout = "fr") +
    geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                   arrow = a, end_cap = circle(.02, 'inches')) +
    geom_node_point(color = "lightblue", size = 1.2) +
    geom_node_text(aes(label = name), vjust = 1, hjust = 1, size = 2.5) +
    labs(x = "", y = "") +
    theme_AP() +
    ggtitle(keywords_vec_stemmed[i])
}
```

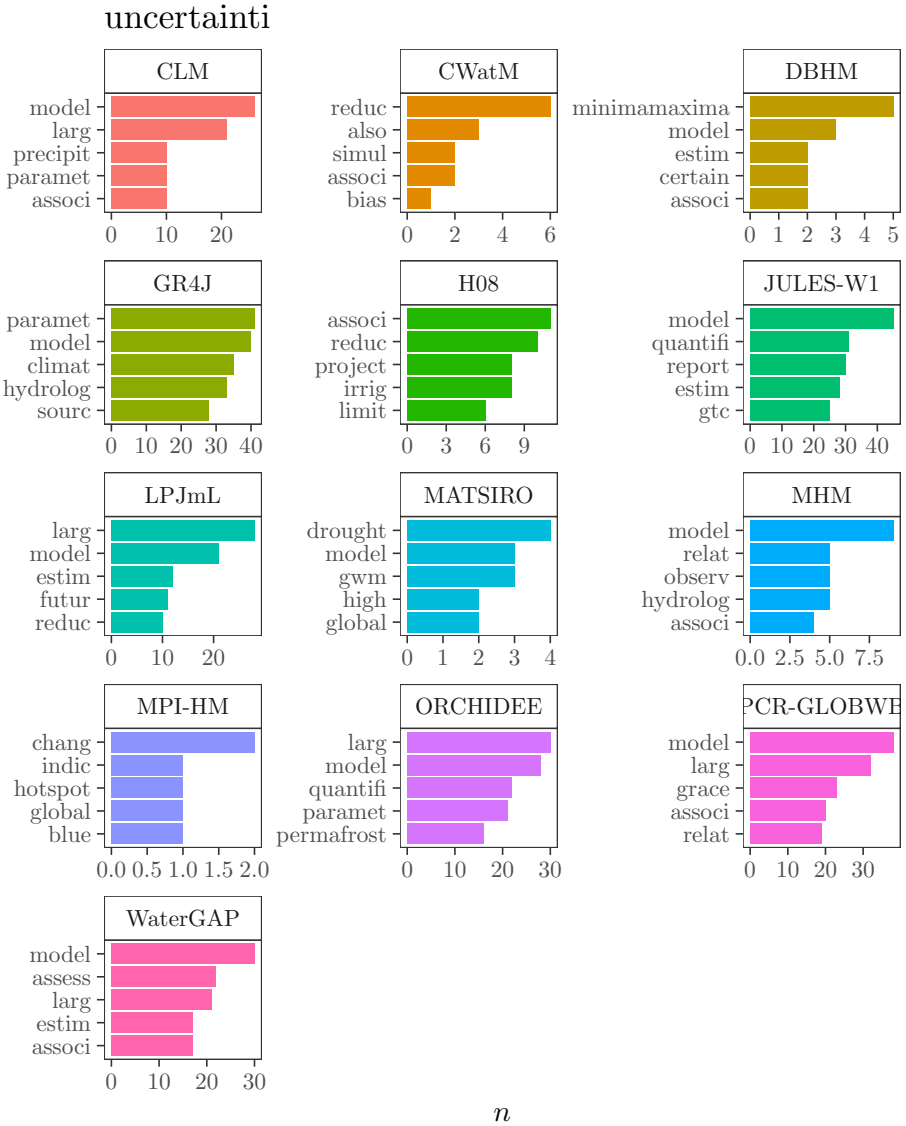
```
# PLOT 25 MOST COMMON BIGRAMS ######################################################

plot_grid(plotlist = plot.token, ncol = 2, labels = "")
```



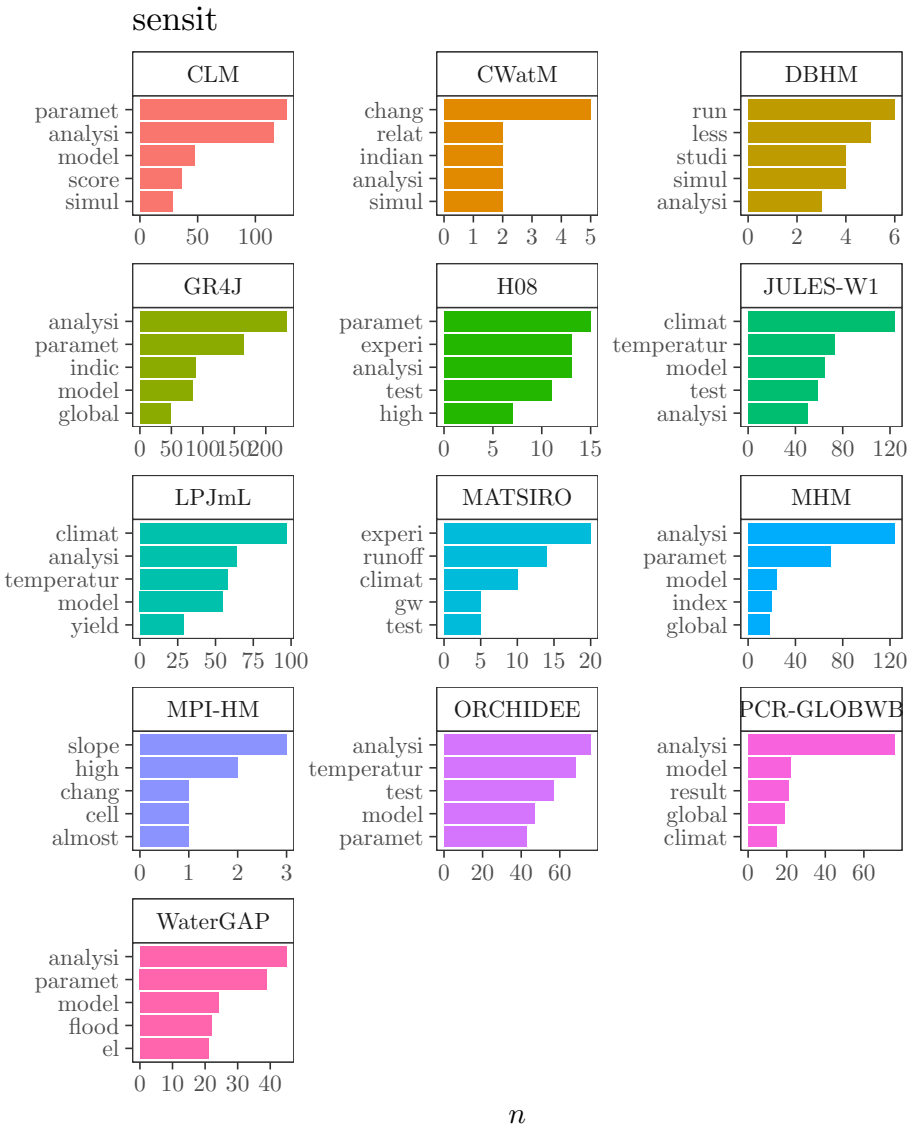

```
# PLOT 5 MOST BIGRAMS ##############################################################
```

```
sapply(1:2, function(i) plot.token.model[i])
```

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




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

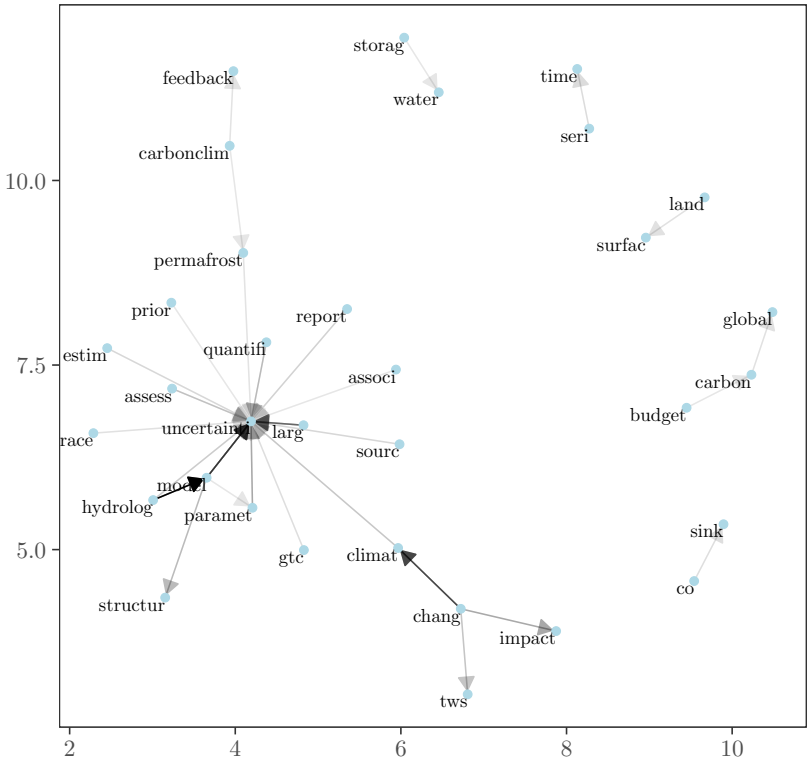



```
# PLOT NETWORKS OF BIGRAMS ##########################################################
```

```
sapply(1:2, function(i) graph_plot[i])
```

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

uncertainti




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

sensit