The treatment of uncertainties in water models

R code

Arnald Puy

Contents

1 Preliminary functions

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

# Load the packages
sensobol::load_packages(c(
  "bibliometrix", "tidyverse", "data.table", "scales", "pdfsearch", "pdftools",
  "openxlsx", "cowplot", "wesanderson", "sjmisc", "ggpubr", "tm", "syuzhet",
  "qdapRegex", "tidytext", "igraph", "ggraph", "wordcloud2", "parallel", "maps",
  "lsa", "LSAfun", "pheatmap", "ggrepel"))

# 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))
}
```

2 Models to analyze

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

water.models <- c("VIC", "TOPMODEL", "ORCHIDEE", "CLM4.5", "GR4J", "JULES-W1",
                  "WaterGAP", "LPJmL", "PCR-GLOBWB", "H08","SACRAMENTO",
                  "MHM", "MATSIRO", "DBHM", "CWatM", "MPI-HM")

models.vec <- paste(water.models, "_ref.bib", sep = "")
models.tolower <- tolower(water.models)

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

# Function to remove the name of models from text
removeWords <- function(str, stopwords) {
  x <- unlist(strsplit(str, " "))
  paste(x[!x %in% stopwords], collapse = " ")
}

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

  y <- tolower(x)
  y <- str_replace_all(y, "[[:punct:]]", " ") # Remove punctuation characters
  y <- tm::removeNumbers(y)
  y <- tm::removeWords(y, stopwords::stopwords(language = "en"))
  y <- str_remove_all(y, "[^[\\da-zA-Z ]]")# Remove all non-alphanumerical
  y <- gsub("\\s[A-Za-z](?= )", " ", y, perl = TRUE) # Remove isolated letters
  #y <- tm::stripWhitespace(y)
  y <- str_squish(y)

  if (stem == TRUE) {
    y <- stemDocument(y) # Stem the document and keep only the root of the word
  }

  return(y)
}

# Function to extract the first 30 words before and after the mention of the
# model name in the abstract
grab_text <- function(text, model) {
  vec <- paste("(( \\S+){30} ", tolower(model), "[[:punct:]\\s]*( \\S+){30})", sep = "")
  str_extract(text, vec)
}
```

3 Bibliometric analysis

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

# Define vectors with keywords -------------------------------------------------

keywords_vec <- c("uncertainty", "sensitivity")
keywords_vec_stemmed <- stemDocument(keywords_vec)

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

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

for (i in 1:length(water.models)) {
  output[[i]] <- convert2df(file = paste(water.models[i], "_ref.bib", sep = ""),
                            dbsource = "wos",
                            format = "bibtex")

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

  title <- output[[i]]$TI

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

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

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

  dt[[i]] <- data.table("WOS" = output[[i]]$UT,
                        "title" = title,
                        "title.large" = tolower(title),
                        "year" = output[[i]]$PY,
                        "keywords" = keywords,
                        "abstract" = output[[i]]$AB,
                        "abstract.large" = output[[i]]$AB)

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

  # Clean text
  dt.clean[[i]][, (selected_cols):= lapply(.SD, function(x)
    clear_text(x)), .SDcols = selected_cols] %>%
    .[, abstract.large:= tolower(abstract.large)]

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

# Add names of models ----------------------------------------------------------

names(years) <- water.models
names(journals) <- water.models
names(dt.clean) <- water.models
```

4 Arrange dataset

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

full.dt <- rbindlist(dt.clean, idcol = "Model") %>%
  .[, year:= ifelse(year == 2023, 2022, year)] # Because
# eight papers were published Early Access end of
# 2022, and ended up in 2023 issues. We count these papers
# as if published in 2022.

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

```
# ARRANGE DATASET ##############################################################

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

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

```
## [1] 2924
```

```
# Number of papers in more than one model
n_occur <- data.frame(table(full.dt$WOS))
WOS.repeated <- data.table(n_occur[n_occur$Freq > 1,])
length(WOS.repeated$Var1) # number of repeated papers
```

```
## [1] 73
```

```
# Fraction of repeated papers over the total
length(WOS.repeated$Var1) / nrow(full.dt)
```

```
## [1] 0.0249658
```

```
# How many papers are repeated twice, three times, etc...
WOS.repeated[, .(N.repeated.papers = .N), Freq]
```

```
##    Freq N.repeated.papers
## 1:    2                62
## 2:    3                 9
## 3:    4                 2
```

```
# Extract which papers are repeated for which model
dt.sample.repeated <- full.dt[WOS %in% WOS.repeated$Var1] %>%
  .[, .(WOS, Model)] %>%
  .[order(WOS)]

dt.sample.repeated
```

```
##                      WOS      Model
##   1: WOS000174380300003        VIC
##   2: WOS000174380300003   TOPMODEL
##   3: WOS000188887100002   TOPMODEL
##   4: WOS000188887100002       GR4J
##   5: WOS000225034000004   TOPMODEL
##  ---
## 155: WOS000752489000002 PCR-GLOBWB
## 156: WOS000802717200001        VIC
## 157: WOS000802717200001   TOPMODEL
## 158: WOSA1997XQ93700015        VIC
## 159: WOSA1997XQ93700015   TOPMODEL
```

```
# Randomly retrieve only one of the repeated studies per model
set.seed(6)
dt.no.repeated <- dt.sample.repeated[,.SD[sample(.N, min(1,.N))], WOS]

# Setkey to filter and retrieve
res <- setkey(full.dt, WOS, Model)[J(dt.no.repeated$WOS, dt.no.repeated$Model)]

# Make the final dataset without repeated papers across models
final.dt <- rbind(res, full.dt[!WOS %in% WOS.repeated$Var1])

# Total number of papers without any repetition
nrow(final.dt)
```

```
## [1] 2838
```

5 Uncertainty and sensitivity datasets

```
# CHECK MENTIONS OF UNCERTAINTY IN THE ABSTRACT, KEYWORDS OR TITLE ############

# Check which papers include "uncertainty" or "sensitivity" in the abstract
final.dt <- final.dt[, `:=` (uncertainti = str_detect(abstract, keywords_vec_stemmed[1]),
                             sensit = str_detect(abstract, keywords_vec_stemmed[2]))]

# Check which papers include "uncertainty" or "sensitivity"in the abstract,
# keywords or title
out <- list()
for(i in 1:length(keywords_vec_stemmed)) {
  out[[i]] <- final.dt[, lapply(.SD, function(x) str_detect(x, keywords_vec_stemmed[i])),
                       .SDcols = (selected_cols)]
}

names(out) <- keywords_vec_stemmed

# Fraction of papers with uncertainti and sensit in the abstract, title,
# keywords ---------------------------------------------------
```

```r
tmp <- lapply(out, function(x) x[, .(n = colSums(.SD)), .SDcols = (selected_cols)][
  , type:= selected_cols][
    , total.n:= nrow(final.dt)][, fraction:= n / total.n])

tmp
```

```
## $uncertainti
##      n     type total.n   fraction
## 1: 134    title    2838 0.04721635
## 2: 558 abstract    2838 0.19661734
## 3: 122 keywords    2838 0.04298802
##
## $sensit
##      n     type total.n   fraction
## 1:  97    title    2838 0.03417900
## 2: 501 abstract    2838 0.17653277
## 3:  66 keywords    2838 0.02325581
```

```r
# Fraction of papers that do not include the words in the abstract but do
# include them in keywords or title --------------------------------------------

tmp2 <- lapply(out, function(x) x[abstract == FALSE][title == TRUE | keywords == TRUE])

da <- rbindlist(tmp2, idcol = "word") %>%
  .[, n.row:= nrow(final.dt)]  %>%
  .[, .N, .(word, n.row)] %>%
  .[, fraction:= N / n.row]

print(da)
```

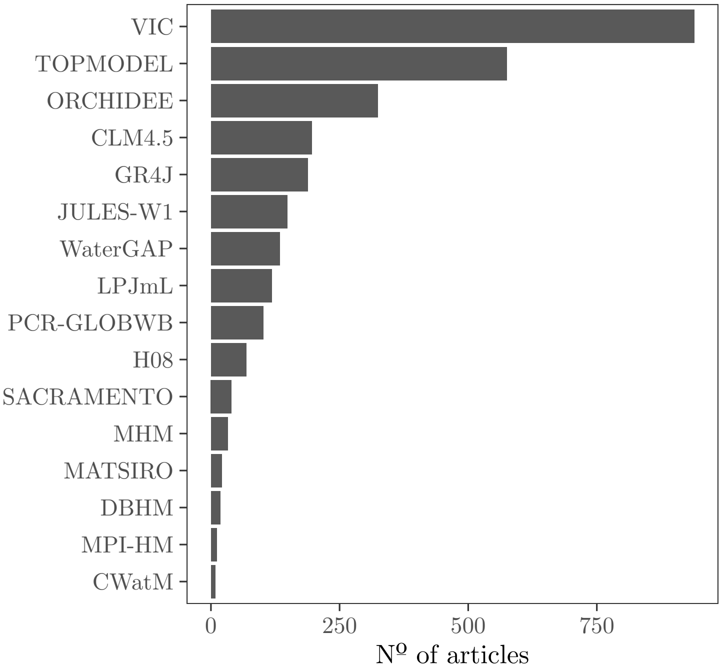
```
##           word n.row  N    fraction
## 1: uncertainti  2838 22 0.007751938
## 2:      sensit  2838 23 0.008104299
```

```r
# CREATE UNCERTAINTY AND SENSITIVITY DATASETS #################################

uncertainty.dt <- final.dt[uncertainti == TRUE]
sensitivity.dt <- final.dt[sensit == TRUE, .(WOS, Model, year, title, abstract, sensit)]
```

```r
# EXPORT FINAL DATASETS ########################################################

all.datasets <- list("final.dt" = final.dt,
                     "uncertainty.dt" = uncertainty.dt,
                     "sensitivity.dt" = sensitivity.dt)


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

  setorder(all.datasets[[i]], -Model, year)
  write.xlsx(all.datasets[[i]], file = paste0(names(all.datasets)[i], ".xlsx"))

}
```

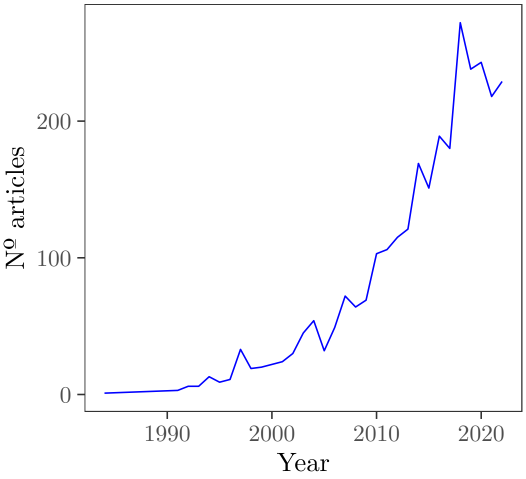
6 Plots

```r
# TOTAL NUMBER OF ARTICLES PER MODEL ##########################################

total.articles <- ggplot(total.n, aes(reorder(Model, total.papers),
                                      total.papers)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(y = "Nº of articles", x = "") +
  theme_AP() +
  theme(legend.position = c(0.65, 0.35))

total.articles
```

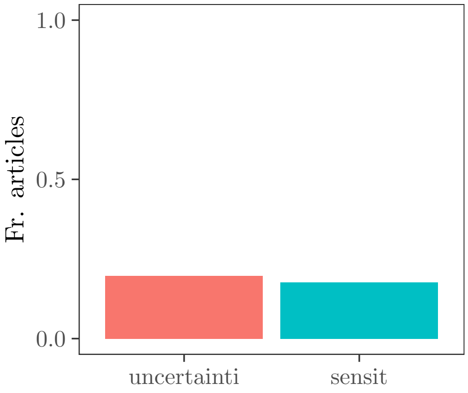


```r
# TOTAL NUMBER OF STUDIES THROUGH TIME ########################################

plot.time <- rbindlist(years, idcol = "Model") %>%
  .[, V1:= ifelse(V1 == 2023, 2022, V1)] %>%
  # For the reasons stated above
  .[, .N, V1] %>%
  ggplot(., aes(V1, N)) +
  geom_line(color = "blue") +
  labs(x = "Year", y = "Nº articles") +
  theme_AP()
```

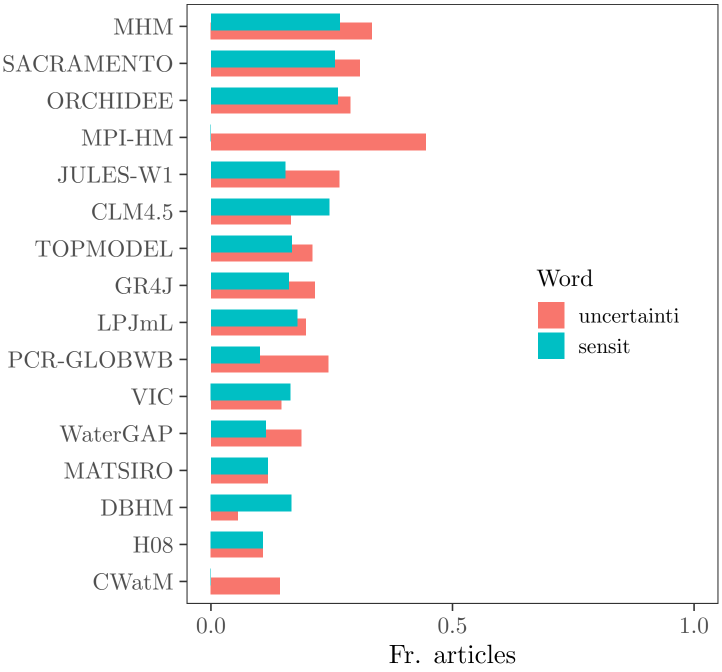
```
plot.time
```



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

plot.n.keywords.bar <- final.dt[, lapply(.SD, function(x)
  sum(x) / .N), .SDcols = (keywords_vec_stemmed)] %>%
  melt(., measure.vars = keywords_vec_stemmed) %>%
  ggplot(., aes(variable, value, fill = variable)) +
  geom_bar(stat = "identity") +
  labs(y = "Fr. articles", x = "") +
  scale_y_continuous(breaks = pretty_breaks(n = 3),
                     limits = c(0, 1)) +
  scale_fill_discrete(name = "Word") +
  theme_AP() +
  theme(legend.position = "none")

plot.n.keywords.bar
```



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

final.dt[, lapply(.SD, function(x)
  sum(x) / .N), .SDcols = (keywords_vec_stemmed), Model] %>%
  melt(., measure.vars = keywords_vec_stemmed) %>%
  ggplot(., aes(reorder(Model, value), value, fill = variable)) +
  geom_bar(stat = "identity",
           position = position_dodge(0.5)) +
  labs(y = "Fr. articles", x = "") +
  scale_y_continuous(breaks = pretty_breaks(n = 3),
                     limits = c(0, 1)) +
  scale_fill_discrete(name = "Word") +
  coord_flip() +
  theme_AP() +
  theme(legend.position = c(0.8, 0.5))
```

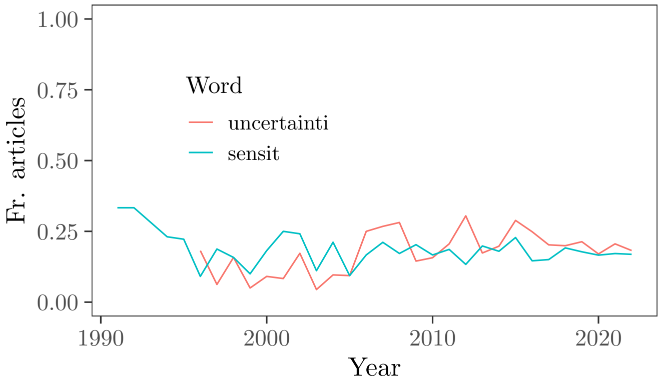


```
# Fraction of studies with both keywords in the abstract
final.dt[uncertainti == "TRUE" & sensit == "TRUE", .N] / full.dt[, .N]
```

```
## [1] 0.04856361
```

```
# FRACTION OF STUDIES WITH WORDS UNCERTAINTI AND SENSIT IN THE
# ABSTRACT, THROUGH TIME ########################################################

total.n.year <- final.dt[, .(total.n = .N), year]

plot.fraction.years <- final.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 = "Word") +
  scale_y_continuous(limits = c(0, 1)) +
  labs(x = "Year", y = "Fr. articles") +
  theme_AP() +
  theme(legend.position = c(0.3, 0.66))

plot.fraction.years
```

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

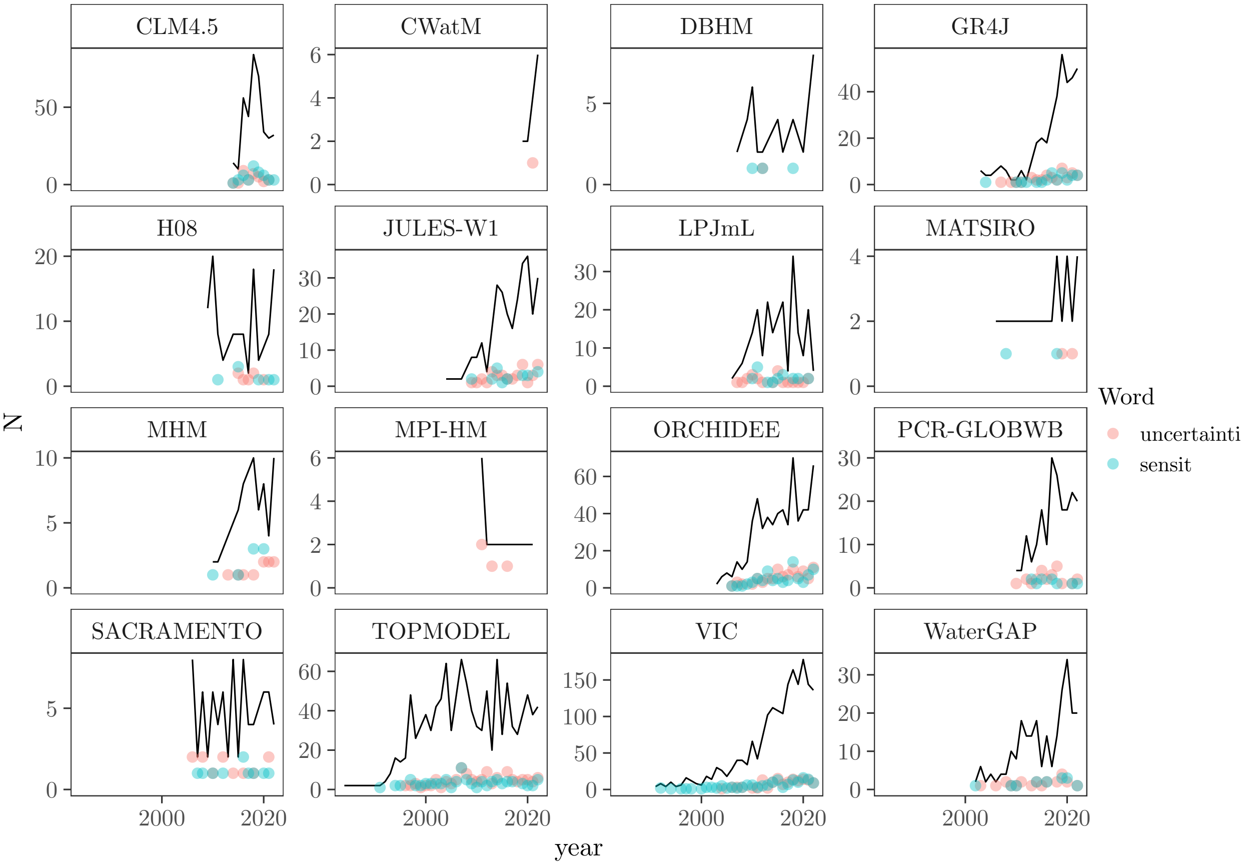


```
# FRACTION OF STUDIES WITH WORDS UNCERTAINTI AND SENSIT IN THE
# ABSTRACT, THROUGH TIME AND BY MODEL #########################################

final.dt[, .N, .(year, Model)]
```

```
##      year    Model  N
##   1: 2002 WaterGAP  1
##   2: 2003 WaterGAP  3
##   3: 2004 WaterGAP  1
##   4: 2005 WaterGAP  2
##   5: 2006 WaterGAP  1
##  ---
## 256: 2018   CLM4.5 42
## 257: 2019   CLM4.5 35
## 258: 2020   CLM4.5 17
## 259: 2021   CLM4.5 15
## 260: 2022   CLM4.5 16
```

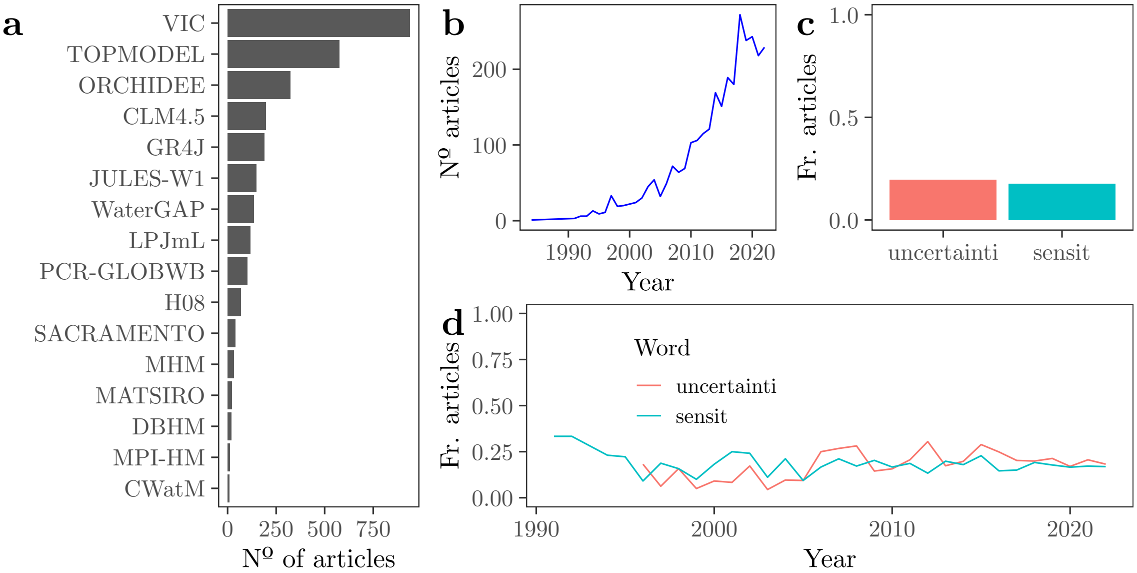
```
da <- final.dt[, .(WOS, uncertainti, sensit, year, Model)] %>%
  melt(., measure.var = keywords_vec_stemmed) %>%
  .[, .N, .(year, Model)] %>%
  ggplot(., aes(year, N, group = Model)) +
  geom_line() +
  scale_y_continuous(limits = c(0, NA),
                     breaks = pretty_breaks(n = 3)) +
  facet_wrap(~Model, scales = "free_y") +
  theme_AP()

years.sa.ua <- final.dt[, .(WOS, uncertainti, sensit, year, Model)] %>%
  melt(., measure.var = keywords_vec_stemmed) %>%
  .[value == TRUE, .N, .(year, variable, Model)]

plot.fraction.years.model <- da +
  geom_point(data = years.sa.ua, aes(year, color = variable), alpha = 0.4) +
  scale_color_discrete(name = "Word") +
  scale_x_continuous(breaks = pretty_breaks(n = 2))

plot.fraction.years.model
```

```
## Warning: Removed 5 rows containing missing values (`geom_line()`).

## Warning: Removed 6 rows containing missing values (`geom_point()`).
```



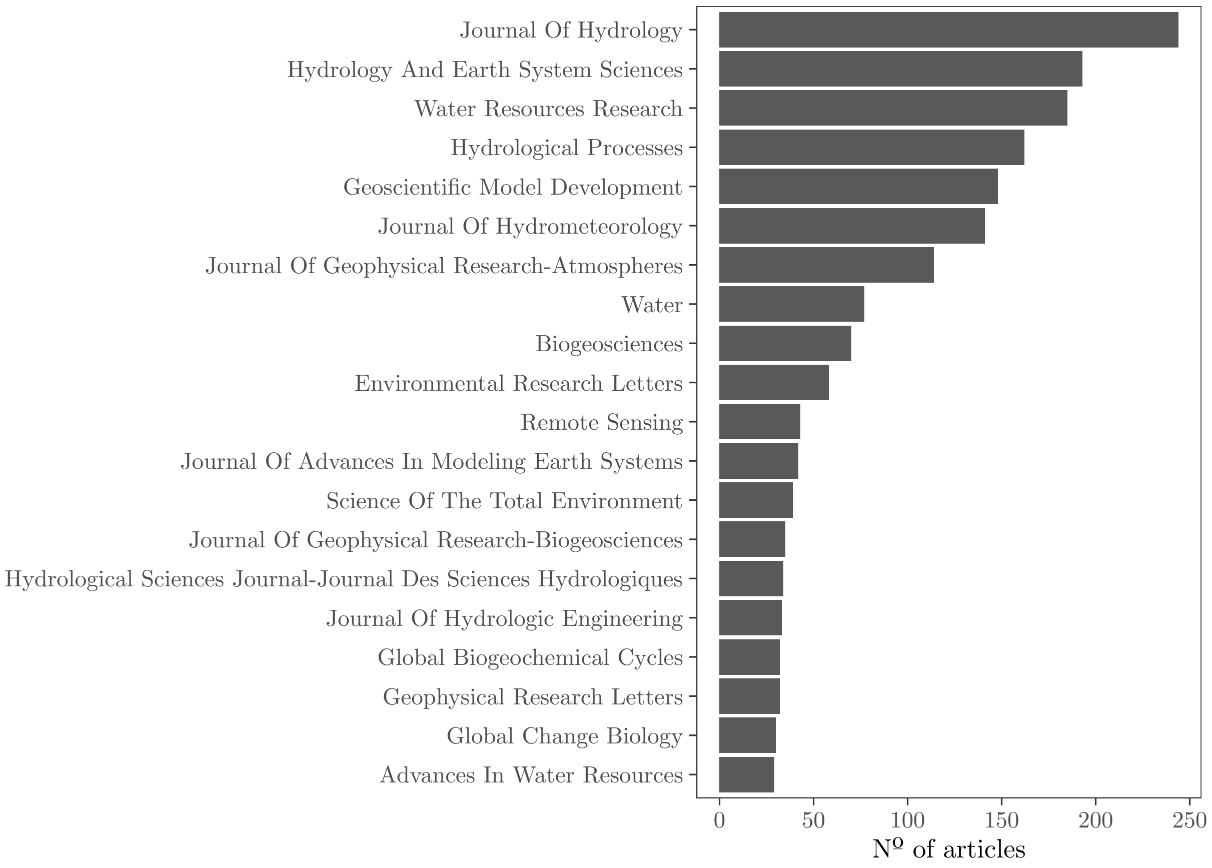
```
# MERGE PLOTS ################################################################

top <- plot_grid(plot.time, plot.n.keywords.bar, ncol = 2, labels = c("b", "c"),
                 rel_widths = c(0.5, 0.5))
right <- plot_grid(top, plot.fraction.years, ncol = 1, labels = c("", "d"),
                   rel_heights = c(0.52, 0.48))
```

```r
plot_grid(total.articles, right, ncol = 2, rel_widths = c(0.38, 0.62),
          labels = c("a", ""))
```



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

tmp <- rbindlist(journals, idcol = "Model")

tmp[, 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()
```

```
## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.
## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.

## Warning in (function (texString, cex = 1, face = 1, engine =
## getOption("tikzDefaultEngine"), : Attempting to calculate the width of a
## Unicode stringusing the pdftex engine. This may fail! See the Unicodesection
## of ?tikzDevice for more information.
```



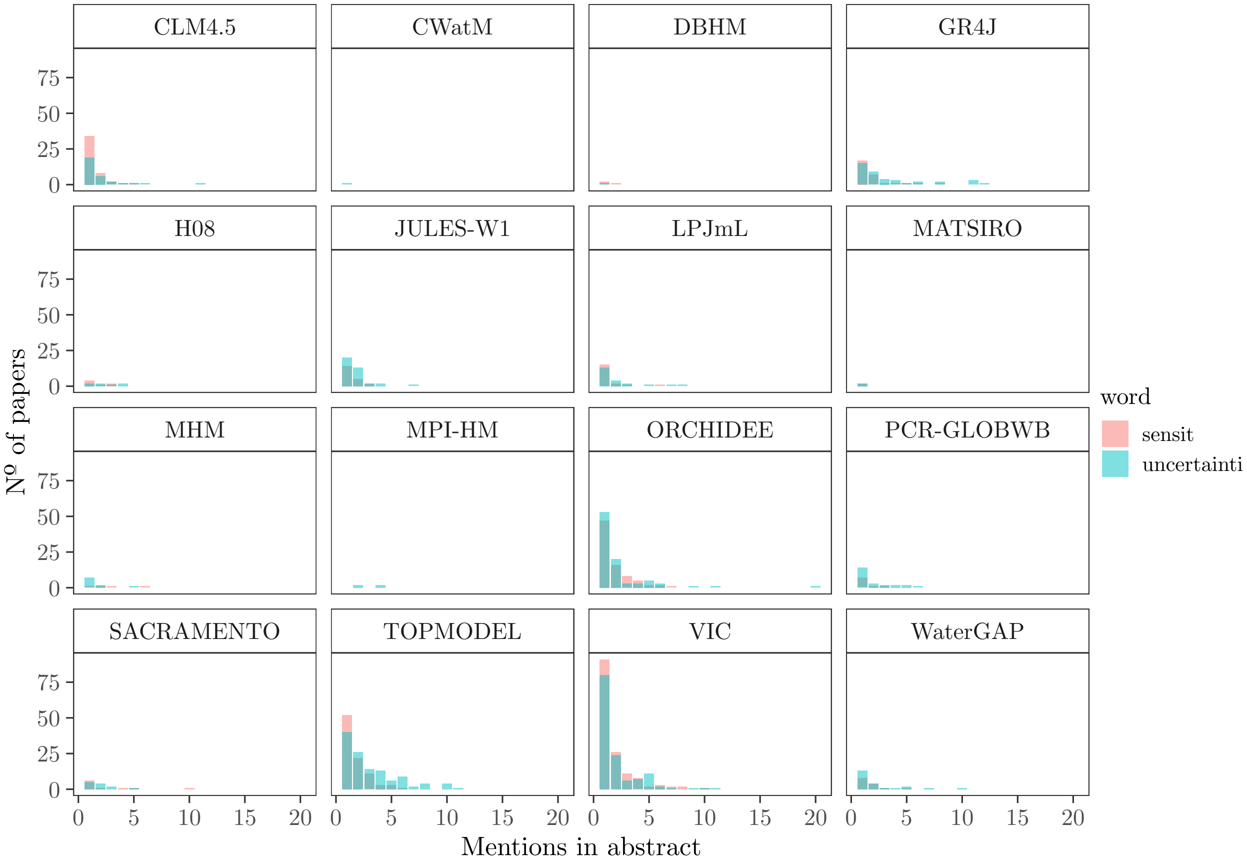
```
# RANK OF WORDS IN ABSTRACT ####################################################

first.n.words <- 100 # Check the most common n words (first n words)

# Check rank overall
rank.dt <- final.dt %>%
  unnest_tokens(word, abstract) %>%
  .[, .N, word] %>%
  .[order(-N), .SD] %>%
  .[, rank := frank(-N, ties.method = "first")]

rank.dt[, head(.SD, first.n.words)] %>%
  print(n = Inf)
```

```
##              word     N rank
##   1:        model 13345    1
##   2:          use  5755    2
##   3:        water  4775    3
##   4:        simul  4339    4
##   5:     hydrolog  4253    5
##   6:         soil  4229    6
##   7:       climat  3722    7
##   8:        chang  3680    8
##  9:        data 3069  9
## 10:      result 3003 10
## 11:        land 2915 11
## 12:       studi 2867 12
## 13:     variabl 2607 13
## 14:      surfac 2546 14
## 15:      runoff 2541 15
## 16:      observ 2412 16
## 17:       basin 2382 17
## 18:      region 2337 18
## 19:        base 2317 19
## 20:     increas 2268 20
## 21:      global 2231 21
## 22:      differ 2098 22
## 23:    precipit 2050 23
## 24:     paramet 2015 24
## 25:       estim 1995 25
## 26:       river 1924 26
## 27:       scale 1916 27
## 28:        show 1908 28
## 29:     moistur 1901 29
## 30:        area 1822 30
## 31:        flow 1684 31
## 32:     spatial 1677 32
## 33:   catchment 1604 33
## 34:     perform 1579 34
## 35:     predict 1560 35
## 36:      impact 1542 36
## 37:     process 1541 37
## 38:         can 1521 38
## 39:      improv 1514 39
## 40:      effect 1503 40
## 41:        time 1480 41
## 42:     drought 1440 42
## 43:      carbon 1433 43
## 44:      compar 1415 44
## 45:  temperatur 1372 45
## 46:     product 1369 46
## 47:         vic 1364 47
## 48:      season 1364 48
## 49:      period 1358 49
## 50:     infiltr 1353 50
## 51:        high 1350 51
## 52:         two 1349 52
## 53: uncertainti 1332 53
## 54:       evalu 1298 54
## 55:  streamflow 1297 55
## 56:   distribut 1285 56
##  57:      rainfal  1241   57
##  58:       method  1240   58
##  59:        capac  1224   59
##  60:       calibr  1190   60
##  61:         flux  1187   61
##  62:        flood  1160   62
##  63:         year  1137   63
##  64:     signific  1121   64
##  65:         larg  1120   65
##  66:        futur  1120   66
##  67:       assess  1092   67
##  68:        veget  1080   68
##  69:       system  1080   69
##  70:      develop  1079   70
##  71:     approach  1077   71
##  72:        relat  1055   72
##  73:        indic  1048   73
##  74:      project  1028   74
##  75:         also  1025   75
##  76:        howev   995   76
##  77:       provid   994   77
##  78:         mean   992   78
##  79:      analysi   979   79
##  80:       measur   973   80
##  81:       annual   960   81
##  82:       condit   958   82
##  83:        three   951   83
##  84:     scenario   936   84
##  85:      resolut   927   85
##  86:         well   919   86
##  87:    groundwat   918   87
##  88:     forecast   916   88
##  89:    atmospher   914   89
##  90:       storag   901   90
##  91:      decreas   898   91
##  92:        dynam   893   92
##  93:       includ   883   93
##  94:       import   882   94
##  95:       sensit   863   95
##  96:         valu   851   96
##  97:        degre   836   97
##  98:      respons   817   98
##  99:        manag   809   99
## 100:       variat   808  100
##               word     N rank
```

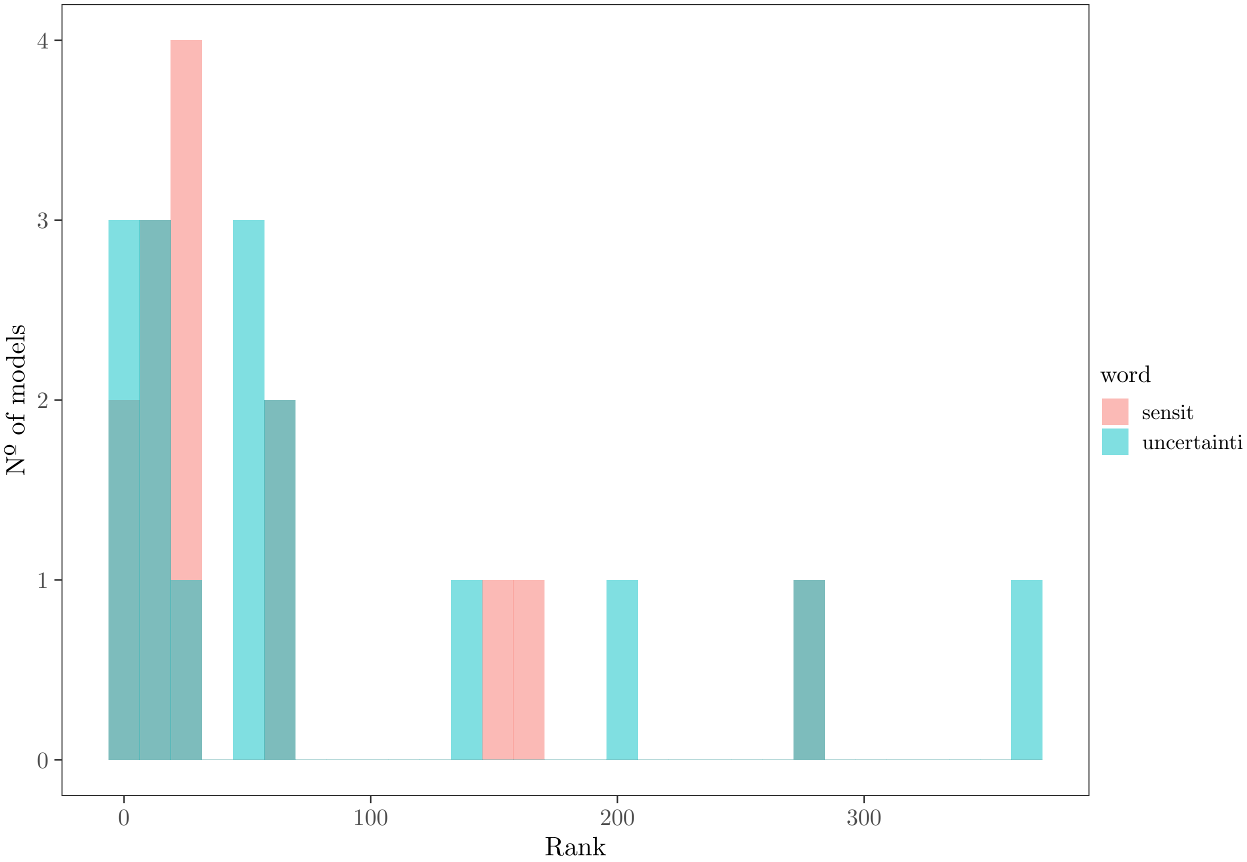
```
# Check rank for uncertainti and sensit
rank.dt[word %in% keywords_vec_stemmed]
```

```
##           word    N rank
## 1: uncertainti 1332   53
## 2:      sensit  863   95
```

```
# Check number of mentions of uncertainti and sensit in abstract per article
tmp <- final.dt %>%
  unnest_tokens(word, abstract) %>%
  .[, .N, .(Model, word, WOS)] %>%
  .[order(-N), .SD, .(Model, WOS)]

out <- tmp[word %in% keywords_vec_stemmed] %>%
  ggplot(., aes(N, fill = word)) +
  geom_bar(position = "identity", alpha = 0.5) +
  facet_wrap(~Model, ncol = 4) +
  scale_x_continuous(breaks = pretty_breaks(n = 4)) +
  labs(x = "Mentions in abstract", y = "Nº of papers") +
  theme_AP()

out
```



```
# Calculate ranks of words in abstract per model
freq.dt <- final.dt %>%
  unnest_tokens(word, abstract) %>%
  .[, .N, .(Model, word)] %>%
  .[order(-N), .SD, Model] %>%
  .[, rank := frank(-N, ties.method = "first"), Model]

# Plot the rank of uncertainti and sensit per model
freq.dt[word %in% keywords_vec_stemmed] %>%
  ggplot(., aes(N, fill = word)) +
  geom_histogram(position = "identity", alpha = 0.5) +
  scale_color_discrete(name = "Word") +
  labs(x = "Rank", y = "Nº of models") +
  theme_AP()
```



```
# Print
dt <- freq.dt[word %in% keywords_vec_stemmed]
setorderv(dt, c("word", "N"))
dt
```

```
##          Model        word   N rank
##  1:    MATSIRO      sensit   2  301
##  2:       DBHM      sensit   4  183
##  3: PCR-GLOBWB      sensit  12  334
##  4:        MHM      sensit  12   79
##  5:        H08      sensit  12  162
##  6:   WaterGAP      sensit  24  201
##  7: SACRAMENTO      sensit  27   33
##  8:      LPJmL      sensit  28  139
##  9:   JULES-W1      sensit  30  166
## 10:       GR4J      sensit  57   87
## 11:     CLM4.5      sensit  65   90
## 12:   ORCHIDEE      sensit 152   56
## 13:   TOPMODEL      sensit 162   85
## 14:        VIC      sensit 276   95
## 15:       DBHM uncertainti   1  580
## 16:      CWatM uncertainti   1  213
## 17:    MATSIRO uncertainti   2  408
## 18:     MPI-HM uncertainti  12   17
## 19:        MHM uncertainti  16   62
## 20:        H08 uncertainti  17   92
## 21: SACRAMENTO uncertainti  24   37
## 22:      LPJmL uncertainti  47   61
## 23: PCR-GLOBWB uncertainti  50   53
## 24:   WaterGAP uncertainti  55   72
## 25:     CLM4.5 uncertainti  63   96
## 26:   JULES-W1 uncertainti  67   62
## 27:       GR4J uncertainti 135   30
## 28:   ORCHIDEE uncertainti 197   39
## 29:        VIC uncertainti 278   94
## 30:   TOPMODEL uncertainti 367   25
##          Model        word   N rank
```

7 Co-occurrence analysis

```r
# Create function --------------------------------------------------------------
tokenize_fun <- function(dt, word, keywords, N.tokens) {

  # Create long dataset
  dt <- melt(dt, measure.vars = keywords)
  output <- dt[variable == word & value == TRUE]

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

  # Vector to retrieve only the bigrams with uncertainti or sensit
  vec <- token.analysis[, str_detect(bigram, word)]
```

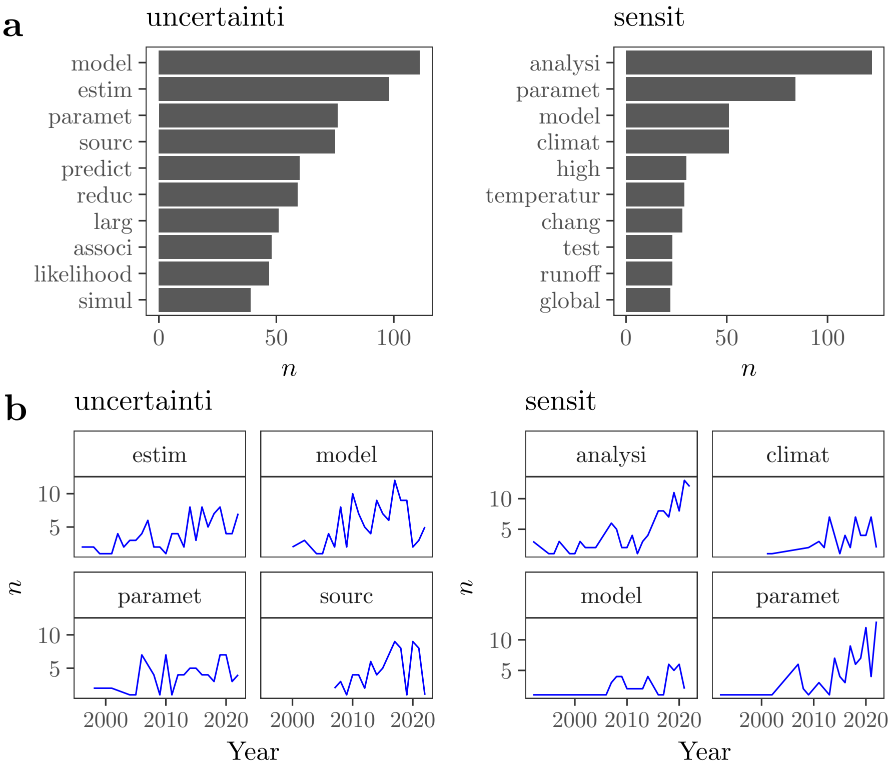
```r
# Final dataset
output.dt <- token.analysis[vec]

# Plot the q0 words most commonly
# associated with uncertainti and sensit ------
plot.token <-  output.dt %>%
  .[, sum(n), bigram] %>%
  .[order(-V1)] %>%
  .[, head(.SD, 10)] %>%
  .[, bigram:= str_squish(str_remove(bigram, word))] %>%
  ggplot(., aes(reorder(bigram, V1, sum), V1)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP() +
  labs(y = "$n$", x = "") +
  ggtitle(word) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11))

# Plot the 4 words most commonly associated with uncertainti and sensit
# and see their evolution through time --------
vec.words <- output.dt[, sum(n), bigram] %>%
  .[order(-V1)] %>%
  .[, head(.SD, 4)] %>%
  .[, bigram:= str_squish(str_remove(bigram, word))] %>%
  .[, bigram]

plot.token.year <- output.dt[, sum(n), .(year, bigram)] %>%
  .[, bigram:= str_squish(str_remove(bigram, word))] %>%
  .[bigram %in% vec.words] %>%
  ggplot(., aes(year, V1)) +
  geom_line(color = "blue") +
  facet_wrap(~bigram) +
  scale_x_continuous(breaks = pretty_breaks(n = 3),
                     guide = guide_axis(check.overlap = TRUE)) +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP() +
  labs(x = "Year", y = "$n$") +
  ggtitle(word) +
  theme(plot.title = element_text(size = 11),
        axis.text.x = element_text(size = 8.5))

# Plot the 4 words most commonly associated with uncertainti and sensit
# in each model -------------------------------

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

  # Arrange and output --------------------------

  out <- list(token.analysis, plot.token, plot.token.year, plot.token.model)
  names(out) <- c("data", "token", "year", "model")

  return(out)

}
```

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

N.tokens <- 2
token.dt <- list()

for (j in keywords_vec_stemmed) {

  token.dt[[j]] <- tokenize_fun(dt = final.dt, word = j,
                                keywords = keywords_vec_stemmed,
                                N.tokens = N.tokens)

}
```

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

top <- plot_grid(token.dt$uncertainti$token, token.dt$sensit$token, ncol = 2,
                 labels = c("a", ""))
bottom <- plot_grid(token.dt$uncertainti$year, token.dt$sensit$year, ncol = 2,
                    labels = c("b", ""))
```

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

```
plot_grid(top, bottom, ncol = 1)
```



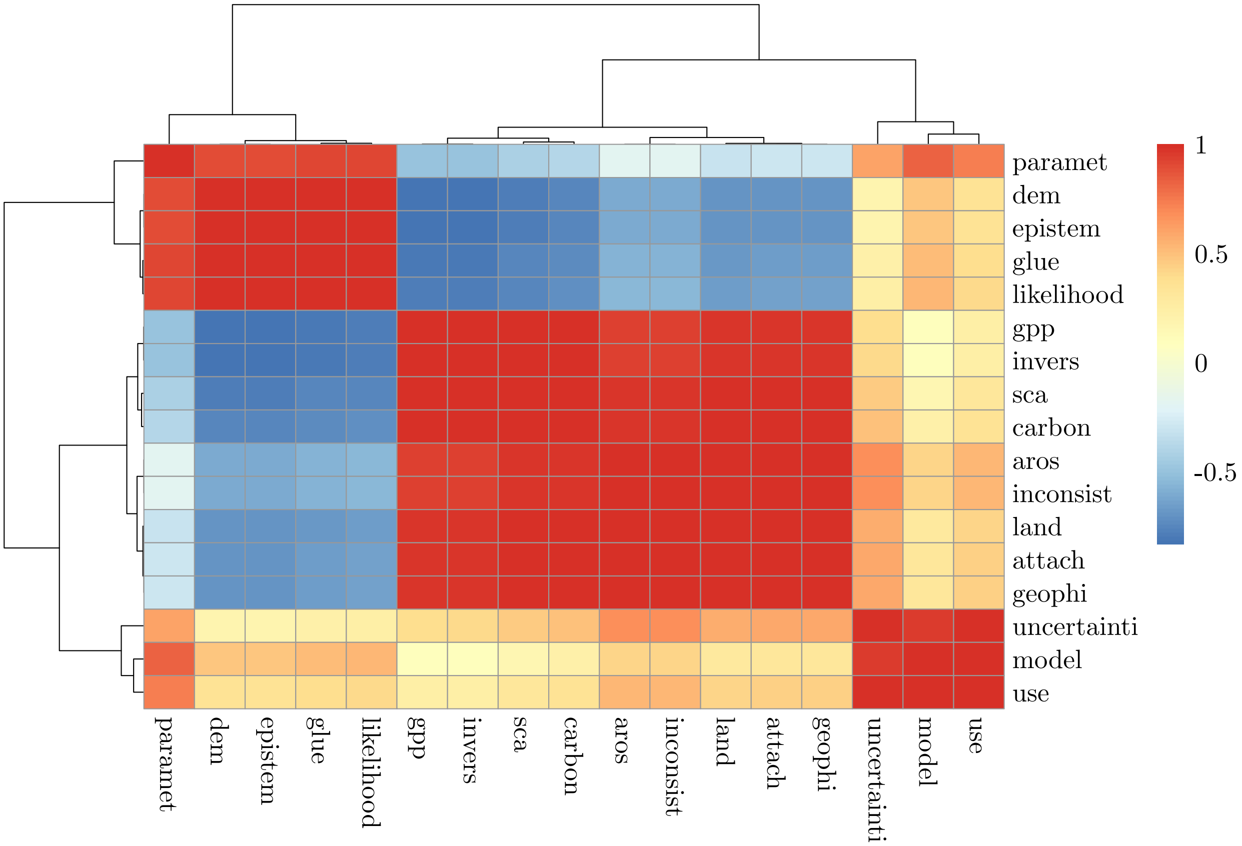
8 Latent Semantic Analysis (LSA)

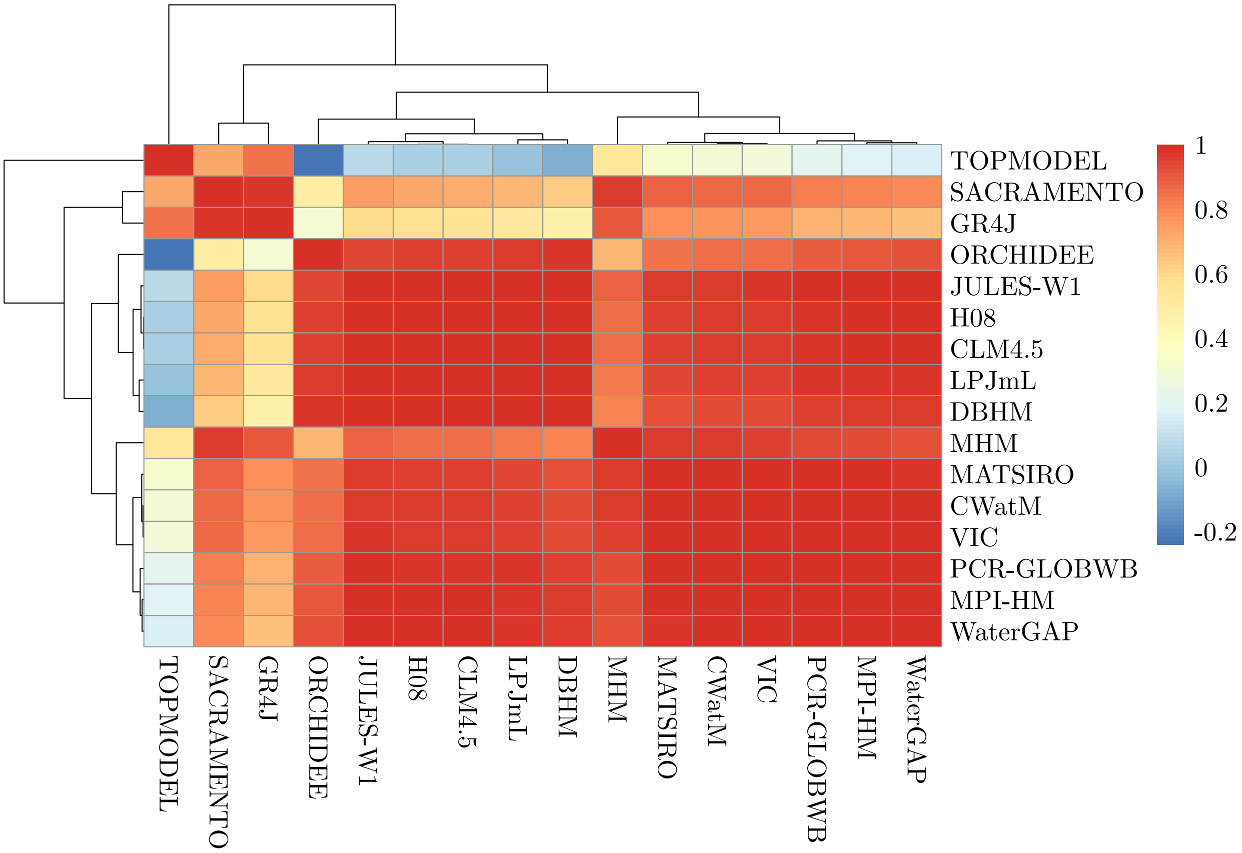
```
# FUNCTION FOR LATENT SEMANTIC ANALYSIS ########################################

# Create function
lsa_fun <- function(dt, max.words) {

  document.names <- dt[[1]]

  # Create the vector space ----------------------------------------------------

  import_corpus <- Corpus(VectorSource(dt$text))

  # Create the textmatrix ------------------------------------------------------

  TDM <- as.matrix(TermDocumentMatrix(import_corpus))

  # Create a weighted matrix ---------------------------------------------------

  TDM2 <- lw_tf(TDM) * gw_idf(TDM) # term frequency times inverse document frequency

  # Run the LSA ----------------------------------------------------------------
```

```r
miniLSAspace <- lsa(TDM2, dims = dimcalc_share())
sk <- miniLSAspace$sk # singular value matrix (SVD)
tk <- miniLSAspace$tk # term matrix
dk <- miniLSAspace$dk # document matrix
words.names <- rownames(tk)
rownames(dk) <- document.names

# Weight the semantic space ---------------------------------------------------

tk2 <- t(sk * t(tk)) # value weighted matrix of terms

# Plotting --------------------------------------------------------------------

# Plot PCA of observations --------------------

dt <- tk2 %>%
  data.frame() %>%
  rownames_to_column(., "words") %>%
  data.table()

a <- dt[order(-X1)][1:max.words][, words]
b <- dt[order(X1)][1:max.words][, words]
c <- dt[order(-X2)][1:max.words][, words]
d <- dt[order(X2)][1:max.words][, words]

selected.words <- unique(c(a, b, c, d))

pca.words <- dt[words %in% selected.words] %>%
  ggplot(., aes(X1, X2, label = words)) +
  geom_point() +
  geom_text_repel(size = 3, max.overlaps = 30) +
  theme_AP()

# Plot PCA of models --------------------------

pca.documents <- dk %>%
  data.table() %>%
  .[, Model:= document.names] %>%
  ggplot(., aes(V1, V2)) +
  scale_color_manual() +
  geom_point() +
  geom_text_repel(label = document.names, size = 3, max.overlaps = 25) +
  theme_AP() +
  theme(legend.position = c(0.9, 0.1))

# Plot heatmap of words -----------------------
```

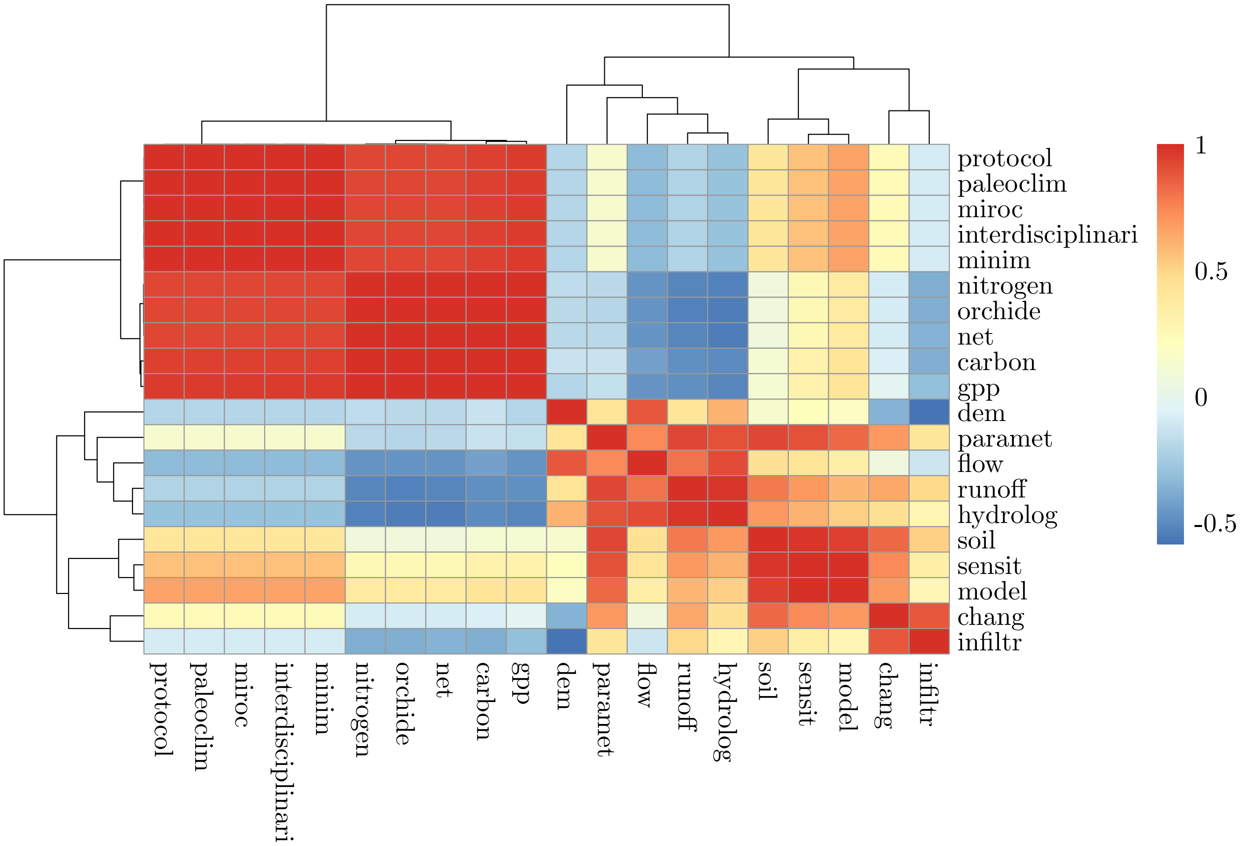
```r
  myCosineSpace2 <- multicos(selected.words, tvectors = tk)
  plot.heatmap.words <- ggplotify::as.ggplot(pheatmap(myCosineSpace2))

  # Plot heatmap of documents -------------------

  model.names <- rownames(dk)
  myCosineSpace3 <- multicos(model.names, tvectors = dk)
  plot.heatmap.doc <- ggplotify::as.ggplot(pheatmap(myCosineSpace3))

  ##############################

  out <- list(tk, dk, sk, tk2, pca.words, pca.documents, plot.heatmap.words,
              plot.heatmap.doc)
  names(out) <- c("tk", "dk", "sk", "tk2", "words", "documents",
                  "heatmap.words", "heatmap.documents")
  return(out)
}
```

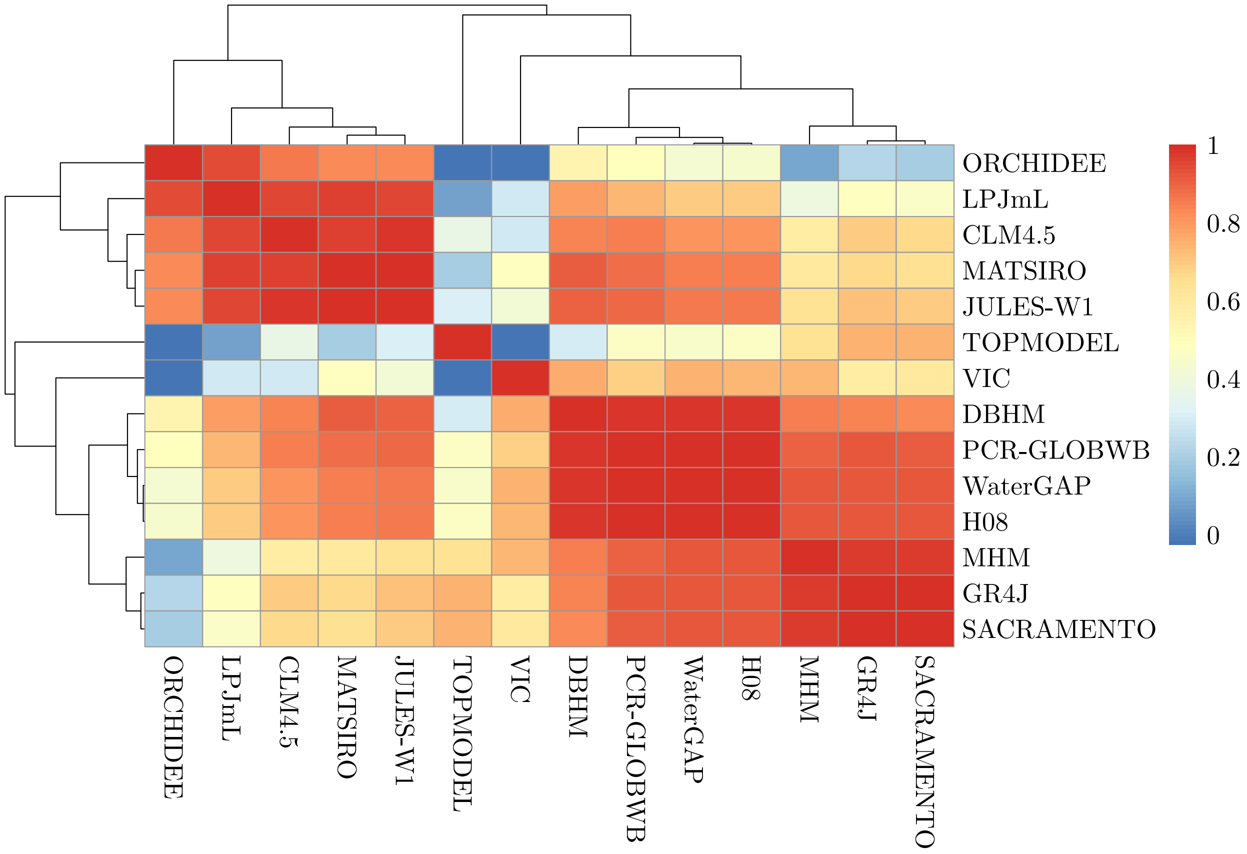
```r
# ARRANGE DATA FOR LATENT SEMANTIC ANALYSIS ####################################

dt.sentences <- final.dt[, .(sentences =  unlist(strsplit(abstract.large, "[.]"))),
                         .(WOS, Model)] %>%
  .[, sentences:= clear_text(sentences)]

dt.sentences[, uncertainti:= str_detect(sentences, keywords_vec_stemmed[1])]
dt.sentences[, sensit:= str_detect(sentences, keywords_vec_stemmed[2])]

# RUN LATENT SEMANTIC ANALYSIS #################################################

# Without removing uncertainti OR sensit
out.unc <- dt.sentences[uncertainti == TRUE] %>%
  .[, .(text = str_squish(paste(sentences, collapse = " "))), Model] %>%
  .[, text:= removeWords(text, models.tolower), Model]

results.unc <- lsa_fun(dt = out.unc, max.words = 5)
```


paramet
dem
epistem
glue
likelihood
gpp
invers
sca
carbon
aros
inconsist
land
attach
geophi
uncertainti
model
use
1
0.5
0
-0.5


TOPMODEL
SACRAMENTO
GR4J
ORCHIDEE
JULES-W1
H08
CLM4.5
LPJmL
DBHM
MHM
MATSIRO
CWatM
VIC
PCR-GLOBWB
MPI-HM
WaterGAP
1
0.8
0.6
0.4
0.2
0
-0.2

```
out.sen <- dt.sentences[sensit == TRUE] %>%
  .[, .(text = str_squish(paste(sentences, collapse = " "))), Model] %>%
  .[, text:= removeWords(text, models.tolower), Model]

results.sen <- lsa_fun(dt = out.sen, max.words = 5)
```




ORCHIDEE
LPJmL
CLM4.5
MATSIRO
JULES-W1
TOPMODEL
VIC
DBHM
PCR-GLOBWB
WaterGAP
H08
MHM
GR4J
SACRAMENTO
1
0.8
0.6
0.4
0.2
0