# Turbulent code in water models

R code

Arnald Puy

## Contents

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

# Load the packages
sensobol::load_packages(c("data.table", "tidyverse", "openxlsx", "tm", "stringr",
                          "pdftools", "tidytext", "scales", "cowplot",
                          "ggrepel", "tidyquant", "text2vec"))
```

```
## Loading required package: text2vec
```

```
##
## Attaching package: 'text2vec'
```

```
## The following objects are masked from 'package:LSAfun':
##
##     coherence, normalize
```

```
# Create custom theme
# 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 = 6),
          axis.title = element_text(size = 10),
          axis.text.x = element_text(size = 7),
          axis.text.y = element_text(size = 7),
          axis.title.x = element_text(size = 7.3),
          axis.title.y = element_text(size = 7.3),
          strip.text.x = element_text(size = 7.4),
```

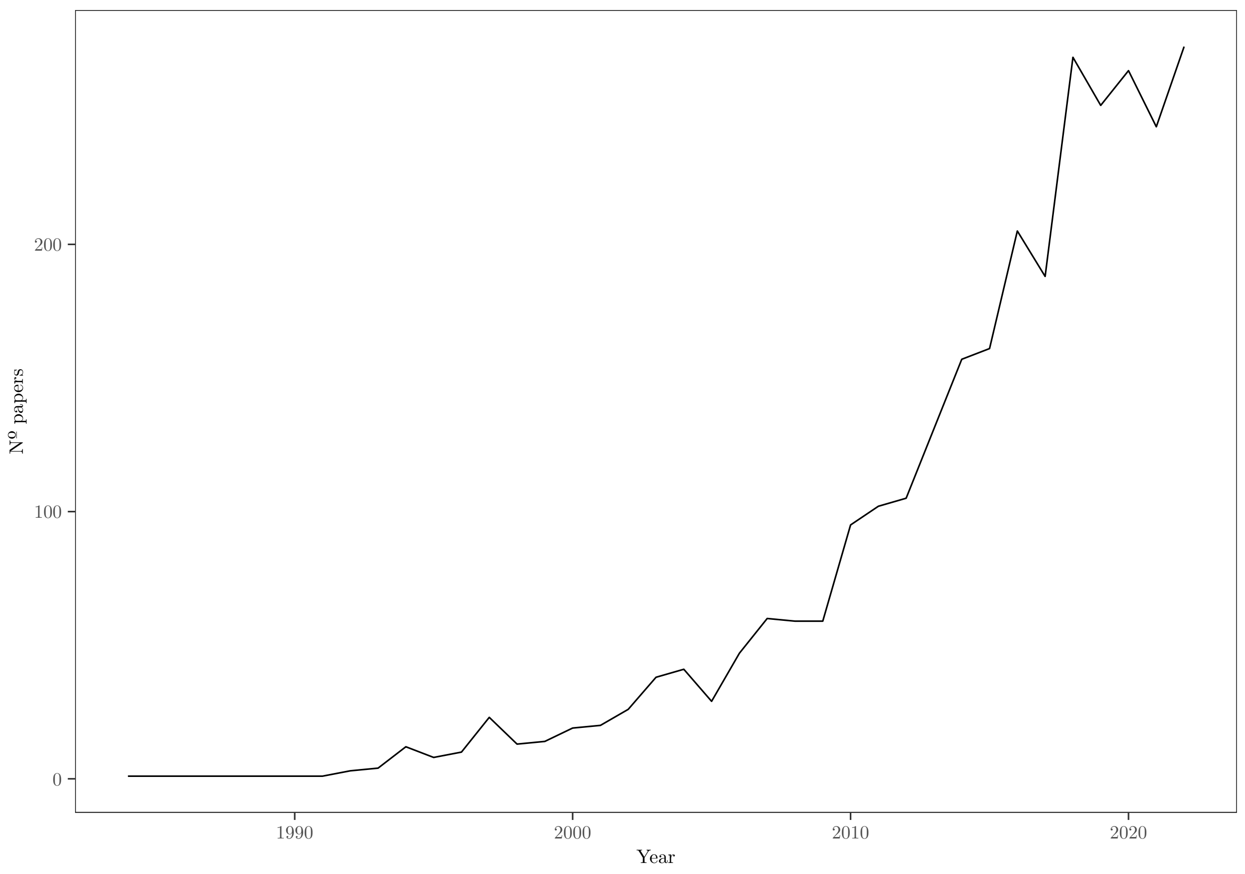
```r
          strip.text.y = element_text(size = 7.4),
          legend.key.width = unit(0.4, "cm"),
          legend.key.height = unit(0.5, "lines"),
          legend.title = element_text(size = 7.5))
}
```

```r
# FUNCTIONS TO CLEAN THE TEXT ##############################################

# Function to remove words 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)
}

# READ DATA ################################################################

dt <- data.table(read.xlsx("final.dt.xlsx"))
dt[, keywords.large:= tolower(keywords.large)]
dt[, abstract.cleaned:= clear_text(abstract.large, stem = FALSE)]

# TERMS TO SEARCH ##########################################################

keywords <- c("validation", "verification", "calibration", "confirmation", "evaluation",
              "benchmarking")
keywords.stemmed <- stemDocument(keywords)

# keywords.stemmed <- c(
#  "calibration", "verification", "validation", "evaluation",
#  "calibrations", "verifications", "validations", "evaluations",
#  "calibrated", "verified", "validated", "evaluated",
#  "calibrating", "verifying", "validating", "evaluating",
#  "calibrative", "verificative", "validative", "evaluative",
#  "calibratively", "verificatively", "validatively", "evaluatively"
# )


# SEARCHING FOR TERMS ##########################################################

# Check which papers include keywords in abstract, keywords or title -----------
selected_cols <- c("title", "abstract", "keywords")
#selected_cols <- c("title.large", "abstract.large", "keywords.large")
out <- list()

for(i in 1:length(keywords.stemmed)) {

  out[[i]] <- dt[, lapply(.SD, function(x)
    str_detect(x, keywords.stemmed[i])), .SDcols = (selected_cols)]

}

# ARRANGE DATA #################################################################

names(out) <- keywords.stemmed

valid.dt <- lapply(out, function(x) rowSums(x, na.rm = TRUE) > 0L) %>%
  do.call(cbind, .) %>%
  data.table() %>%
  .[, any.column:= rowSums(.SD) > 0]

full.dt <- cbind(dt, valid.dt)
full.dt.cols <- data.frame(full.dt[, .SD, .SDcols = keywords.stemmed])

# DESCRIPTIVE STATISTICS #######################################################

full.dt[, lapply(.SD, sum), .SDcols = keywords.stemmed]
```

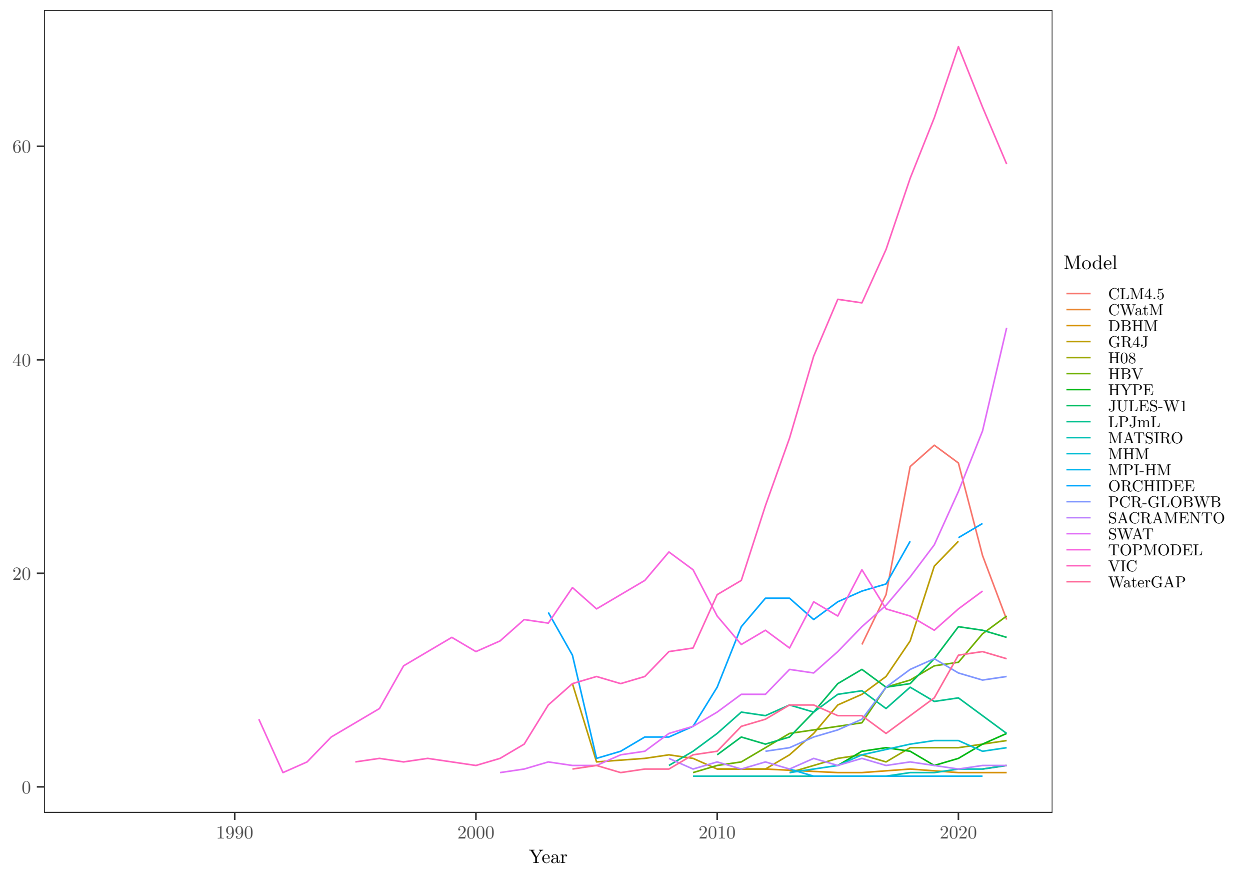
```
##    valid verif calibr confirm evalu benchmark
##    <int> <int>  <int>   <int> <int>     <int>
## 1:   536    76    743      74  1065        86
```

```r
# Count number of papers with mentions to validation, validation + verification,
# validation + calibration, etc ------------------------------------------------

# Function to count the number of TRUE values shared between columns
count_shared_true <- function(data, cols) {
  sum(rowSums(data[, cols]) == length(cols))
}

# Create an empty list to store results
results_list <- list()

# Loop through all combinations of columns and count shared TRUE values
for (size in 2:length(keywords.stemmed)) {

  for (cols_combination in combn(ncol(full.dt.cols), size, simplify = FALSE)) {

    shared_true_count <- count_shared_true(data = full.dt.cols, cols = cols_combination)
    col_names <- colnames(full.dt.cols)[cols_combination]

    # Append results to the list
    results_list <- c(results_list, list(data.table(combination = paste(col_names, collapse = 
  }
}

# Combine the list of data.tables into a single data.table
comb_dt <- rbindlist(results_list)
```

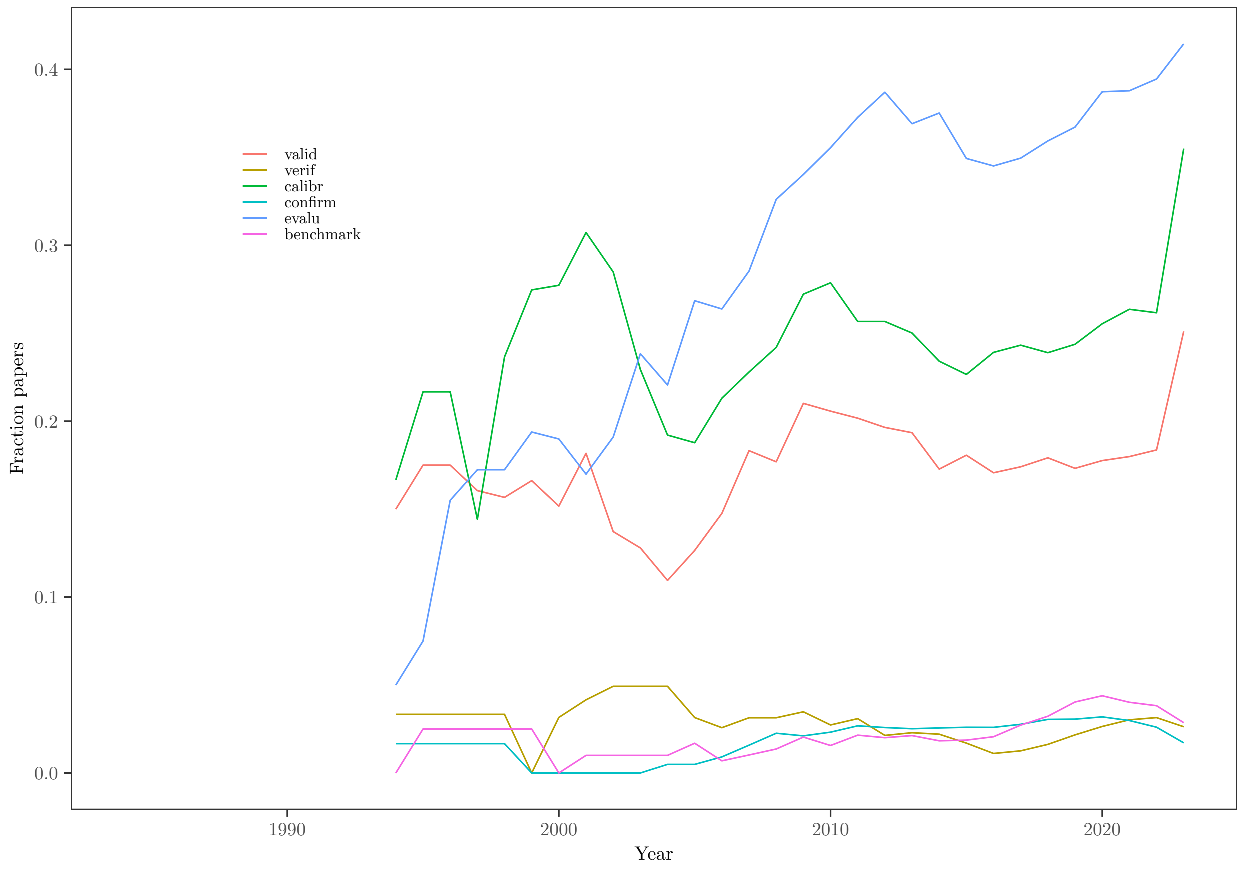
```
# PLOTS ########################################################################

plot.year <- full.dt[, .N, year] %>%
  .[!year == 2023] %>%
  ggplot(., aes(year, N)) +
  geom_line() +
  labs(x = "Year", y = "Nº papers") +
  theme_AP()

plot.year
```



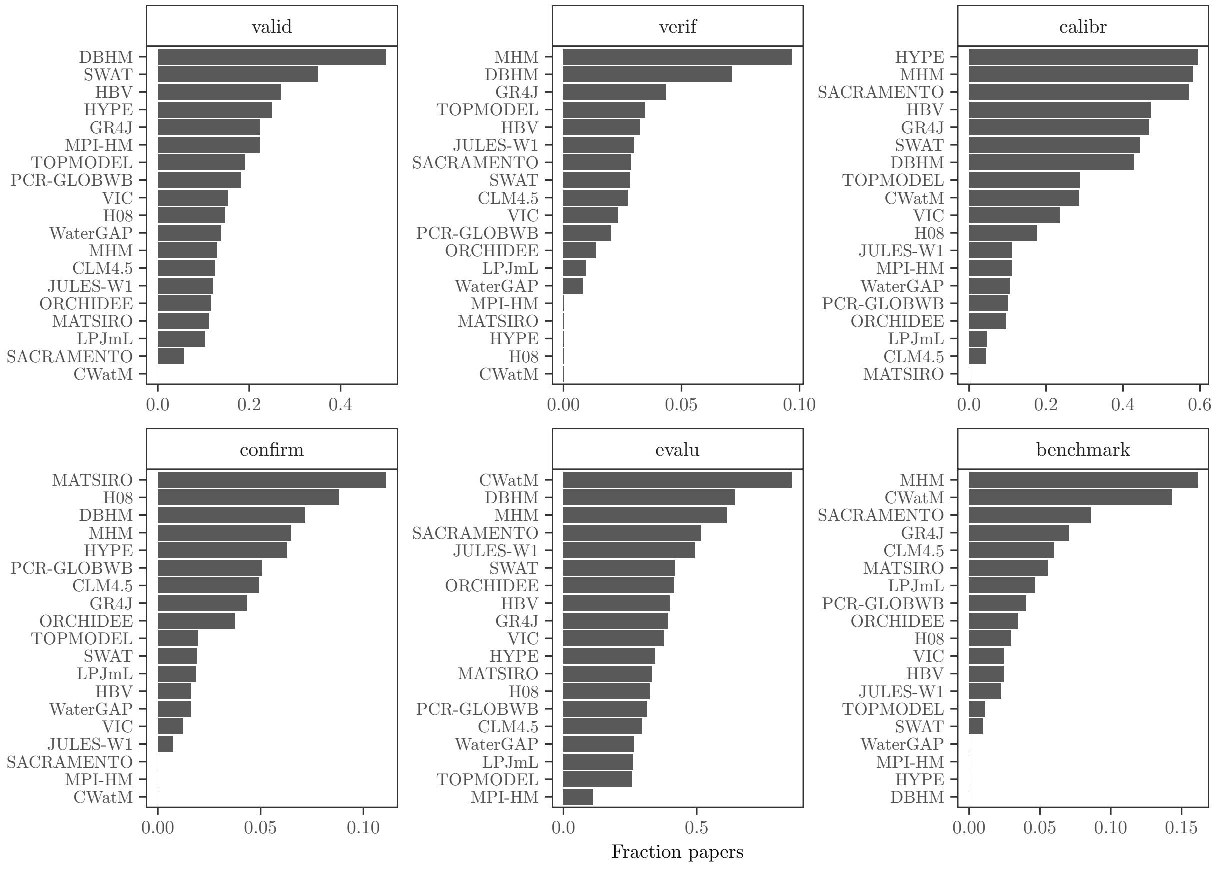
```
plot.model.year <- full.dt[, .N, .(Model, year)] %>%
  .[!year == 2023] %>%
  ggplot(., aes(year, N, color = Model)) +
  geom_ma(ma_fun = SMA, n = 3, lty = 1) +
  labs(x = "Year", y = "") +
  theme_AP()

plot.model.year
```



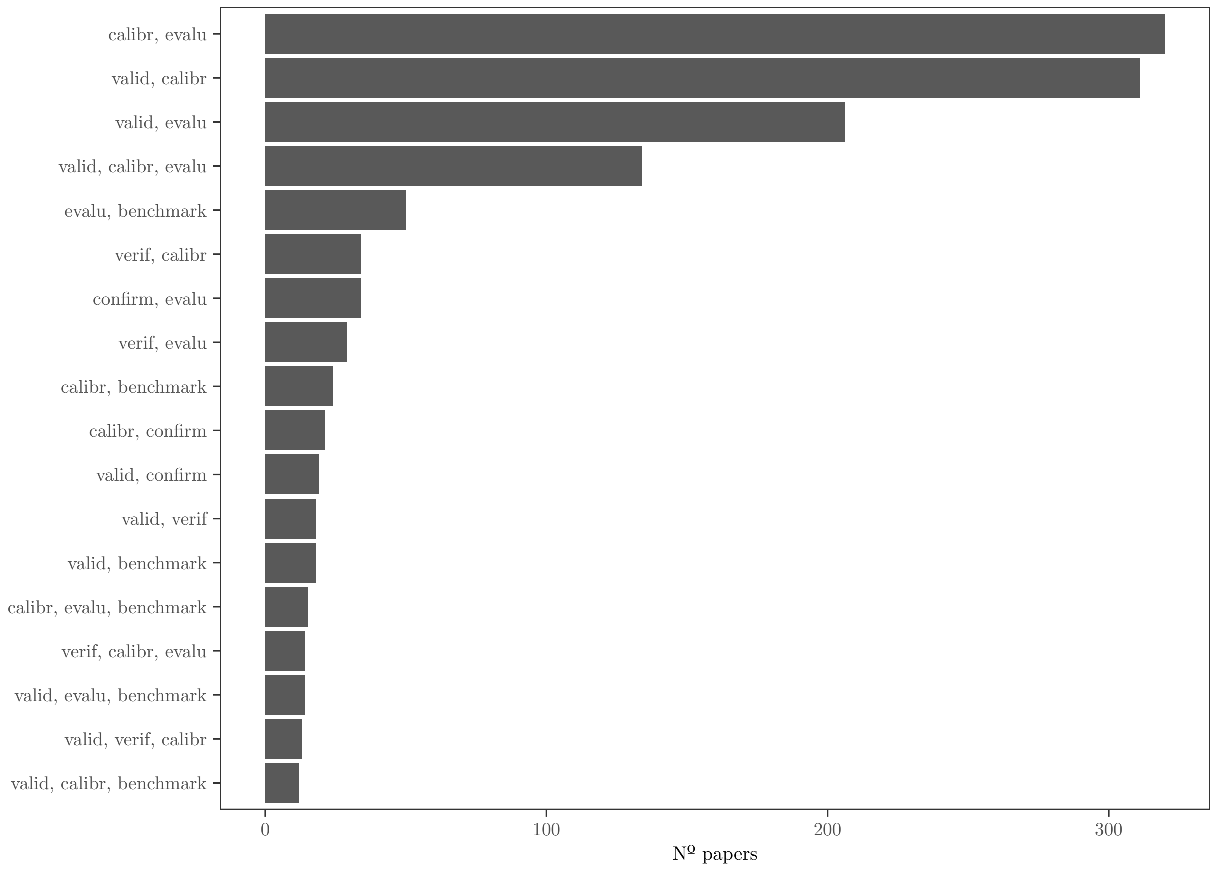
```
plot.keywords.time <- merge(full.dt, full.dt[, .(total.papers = .N), year], by = "year") %>%
  melt(., measure.vars = keywords.stemmed) %>%
  .[, sum(value, na.rm = TRUE), .(variable, year, total.papers)] %>%
  .[, fraction:= V1 / total.papers] %>%
  ggplot(., aes(year, fraction, color = variable)) +
  scale_color_discrete(name = "") +
  geom_ma(ma_fun = SMA, n = 5, lty = 1) +
  theme_AP() +
  theme(legend.position = c(0.2, 0.8)) +
  labs(x = "Year", y = "Fraction papers")

plot.keywords.time
```



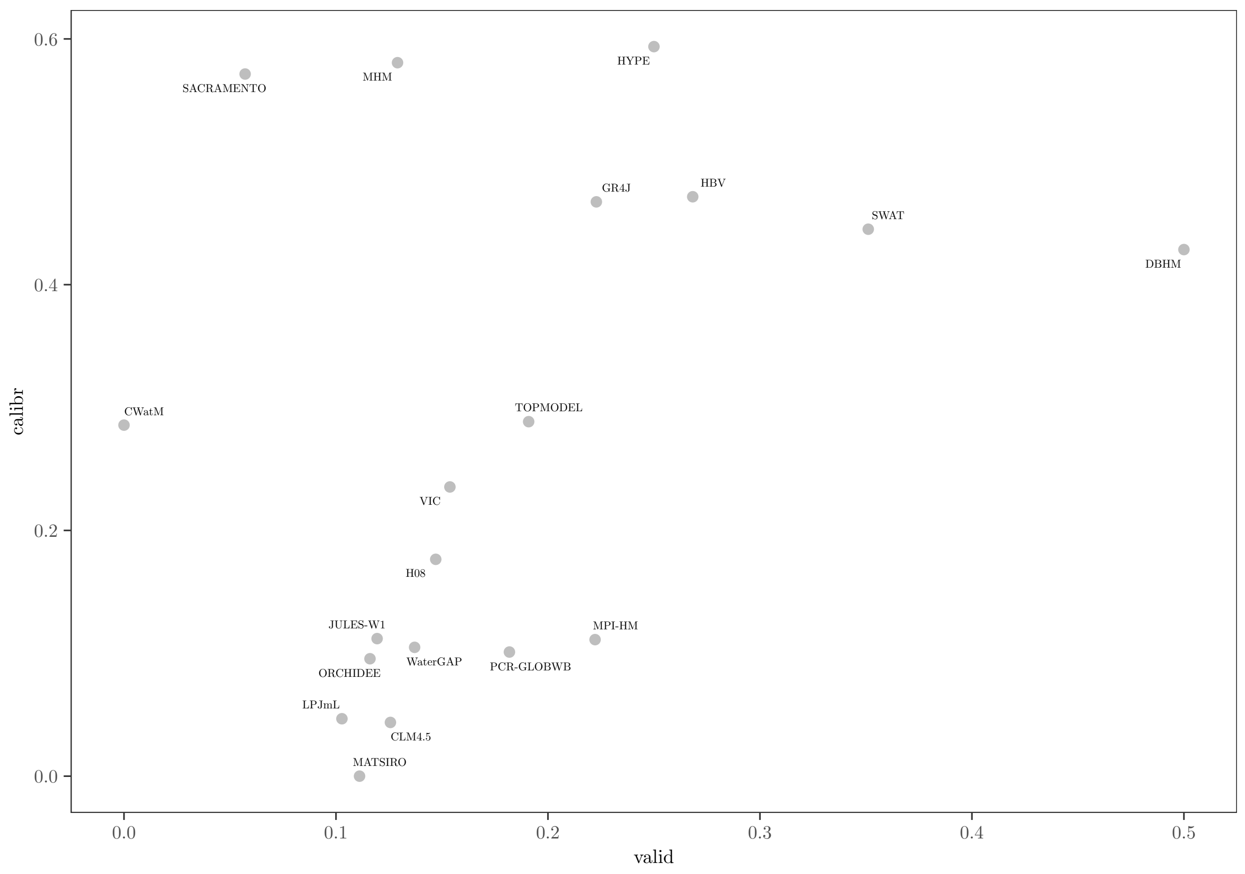
```
tmp <- merge(full.dt, full.dt[, .(total.papers = .N), Model], by = "Model") %>%
  melt(., measure.vars = keywords.stemmed) %>%
  .[, sum(value, na.rm = TRUE), .(variable, Model, total.papers)] %>%
  .[, fraction:= V1 / total.papers] %>%
  mutate(variable = as.factor(variable),
         name = reorder_within(Model, fraction, variable))

plot.keyword.per.model <- tmp %>%
  ggplot(., aes(name, fraction)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  facet_wrap(~variable, scales = "free") +
  scale_x_reordered() +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  theme_AP() +
  labs(x = "", y = "Fraction papers") +
  theme(axis.text.y = element_text(size = 6.5))

plot.keyword.per.model
```



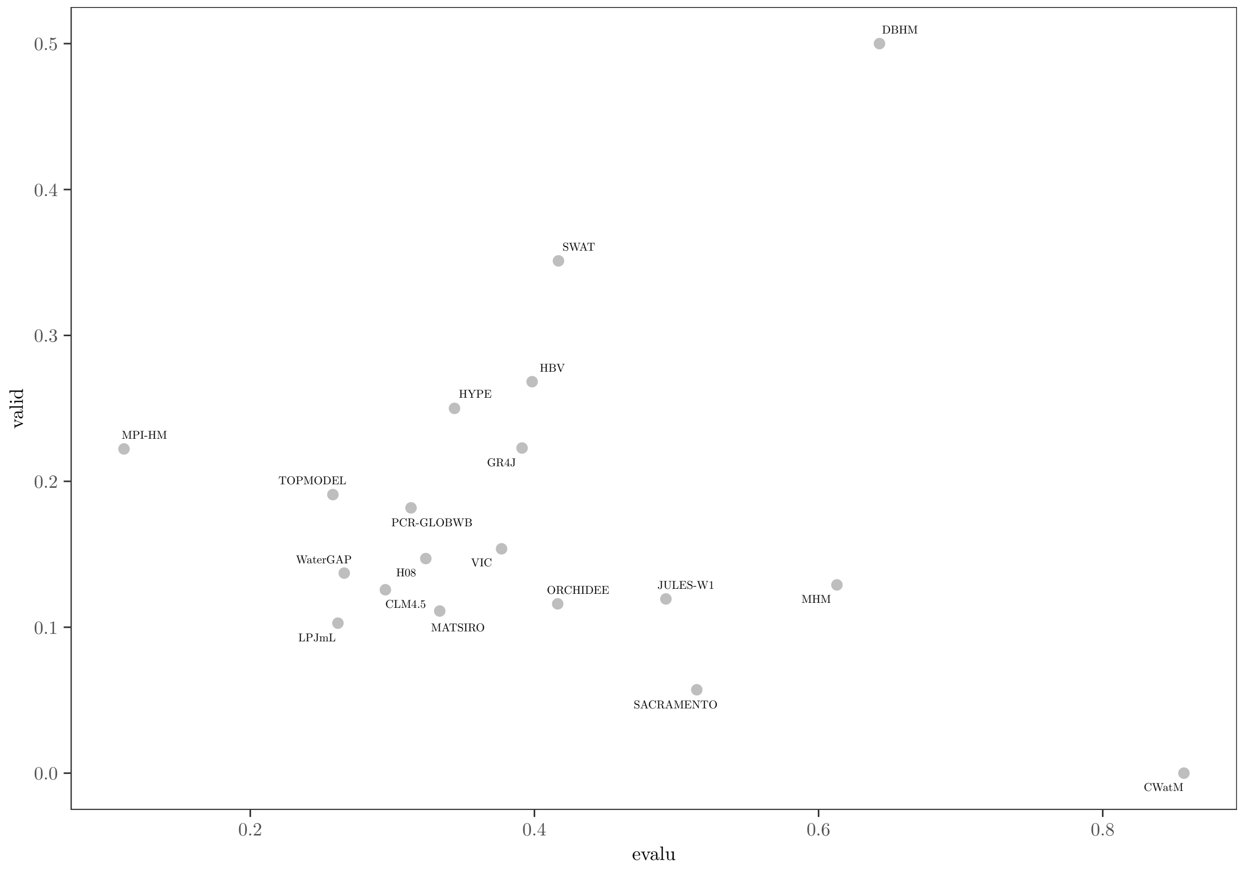
```r
plot.keyword.comb <- comb_dt[N > 10] %>%
  ggplot(., aes(reorder(combination, N), N)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_AP() +
  theme(axis.text.y = element_text(size = 7)) +
  labs(x = "", y = "Nº papers")

plot.keyword.comb
```



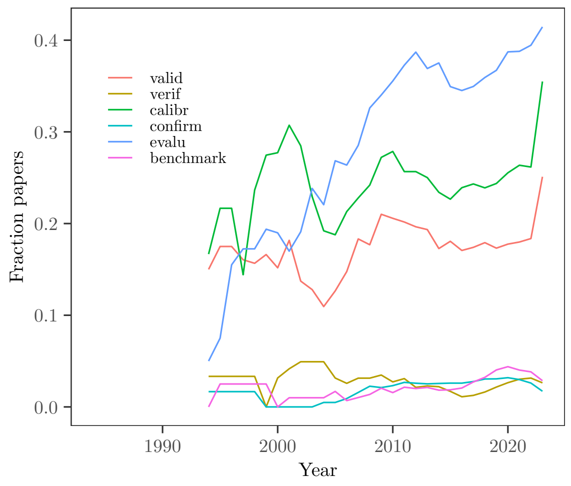
```r
total.papers.model <- full.dt[, .N, Model]

plot.valid.calibr <- tmp %>%
  dcast(., Model ~ variable, value.var = "fraction") %>%
  merge(., total.papers.model, by = "Model") %>%
  ggplot(., aes(valid, calibr, label = Model)) +
  geom_point(color = "grey") +
  geom_text_repel(size = 1.5) +
  theme_AP() +
  theme(legend.position = "none")

plot.valid.calibr
```



```r
plot.evalu.valid <- tmp %>%
  dcast(., Model ~ variable, value.var = "fraction") %>%
  merge(., total.papers.model, by = "Model") %>%
  ggplot(., aes(evalu, valid, label = Model)) +
  geom_point(color = "grey") +
  geom_text_repel(size = 1.5) +
  theme_AP() +
  theme(legend.position = "right")

plot.evalu.valid
```

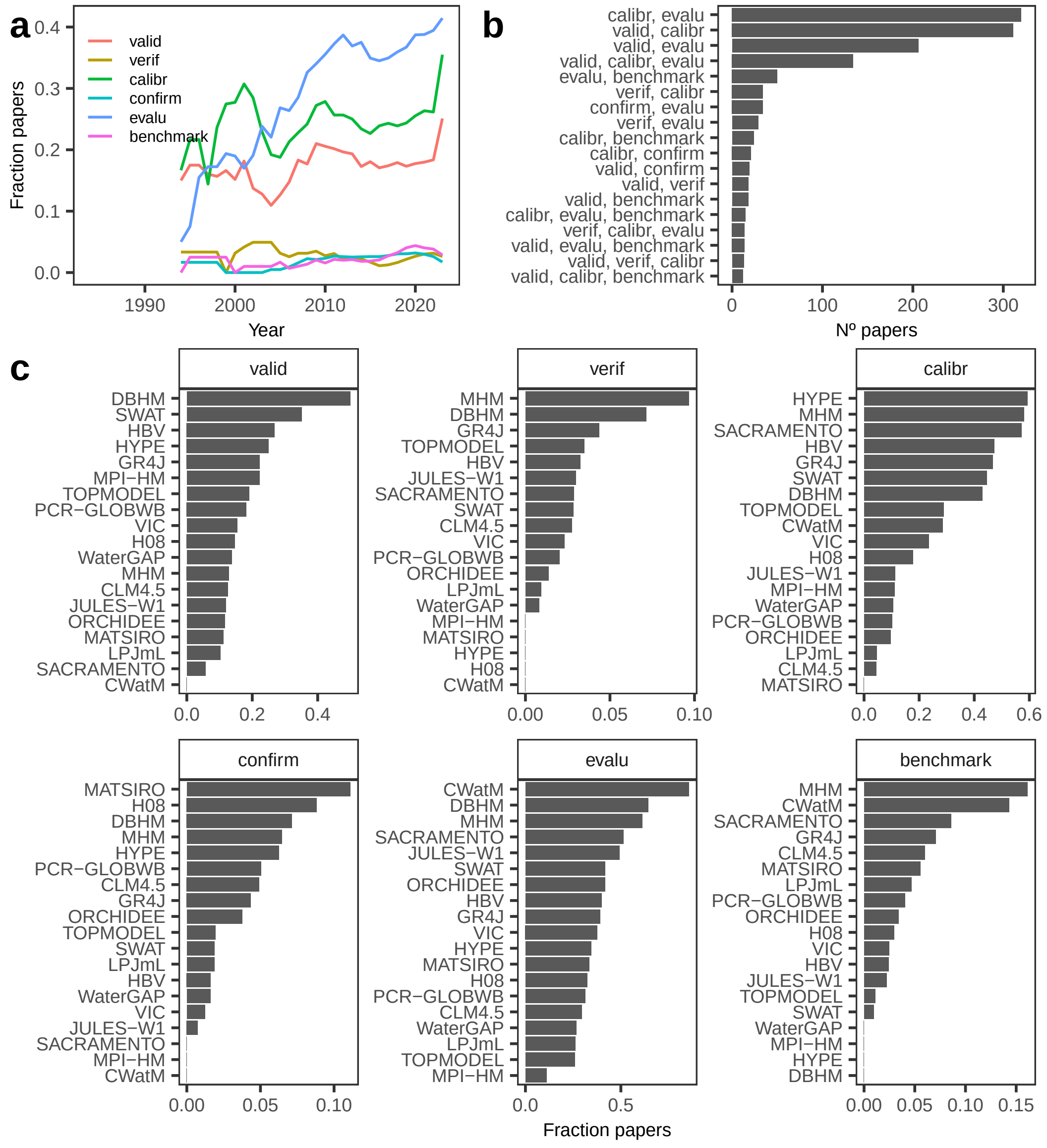


```
plot.keywords.time
```

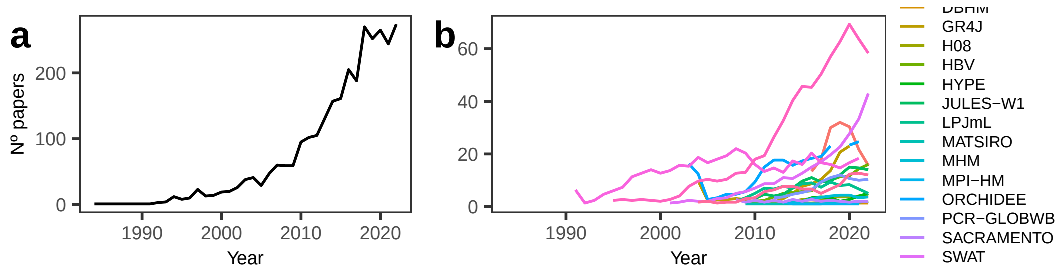


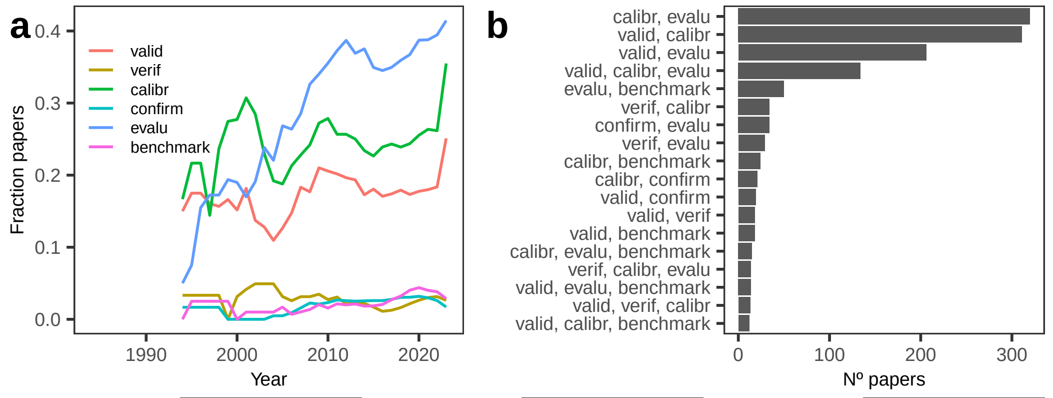
```
# MERGE DESCRIPTIVE PLOTS ########################################################

top <- plot_grid(plot.keywords.time, plot.keyword.comb, ncol = 2, labels = "auto",
                 rel_widths = c(0.45, 0.55))

plot.merged <- plot_grid(top, plot.keyword.per.model, ncol = 1,
                         labels = c("", "c"), rel_heights = c(0.3, 0.7))
```

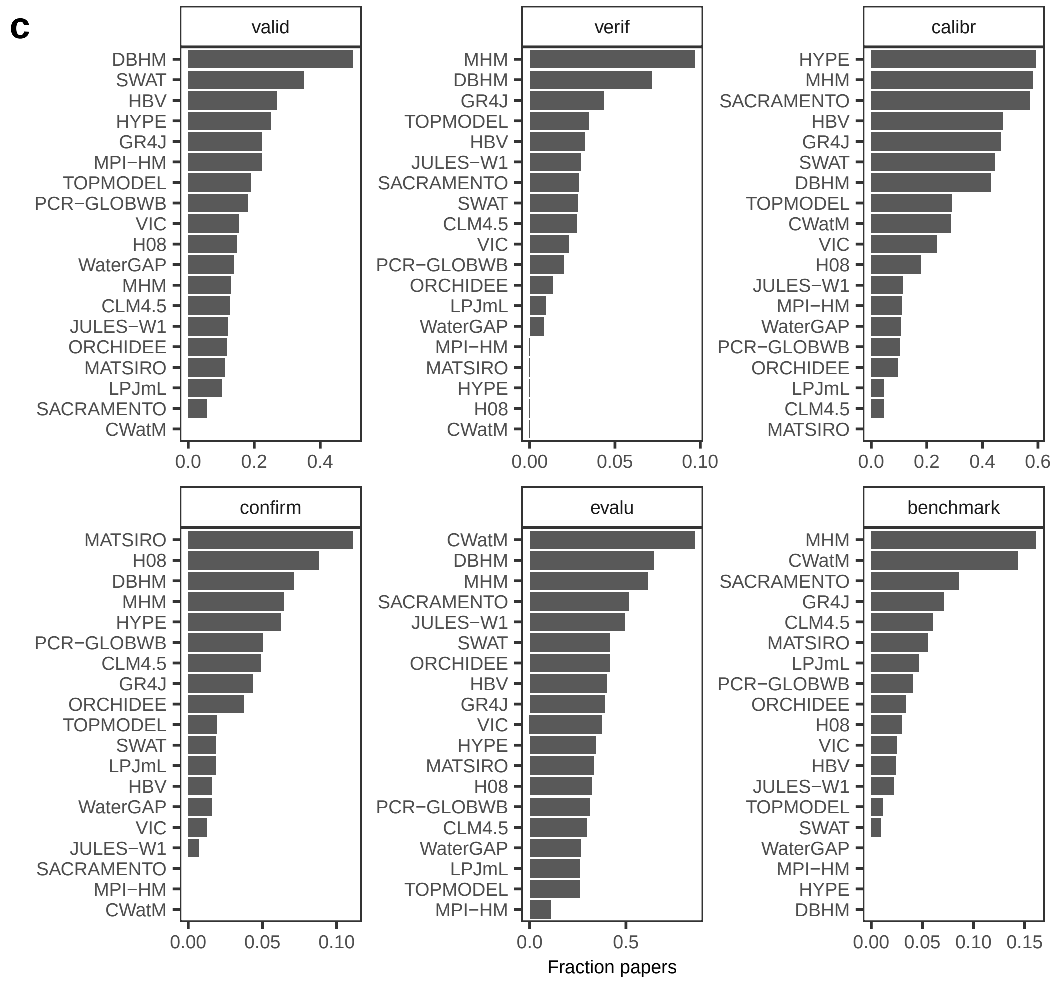
```
plot.merged
```



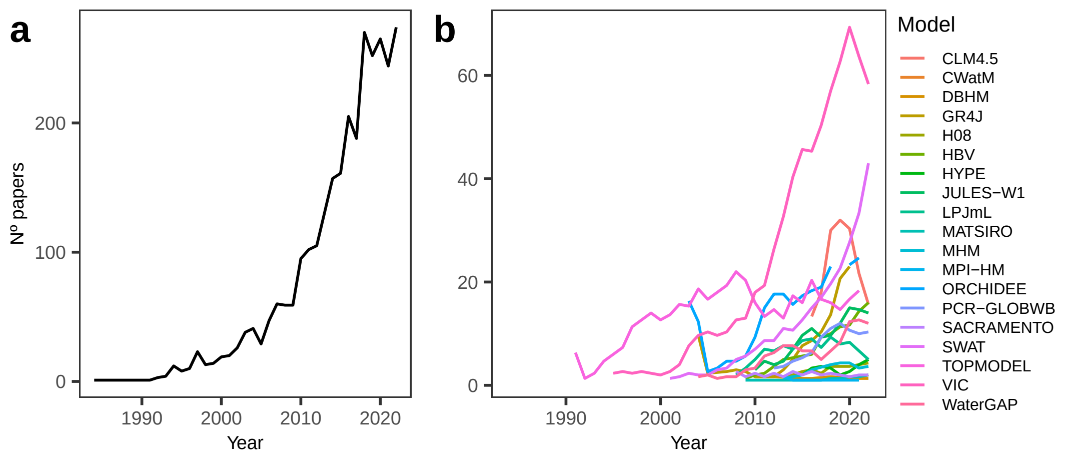
```
toppest <- plot_grid(plot.year, plot.model.year, ncol = 2, labels = "auto",
                     rel_widths = c(0.4, 0.6))

top <- plot_grid(plot.keywords.time, plot.keyword.comb, ncol = 2, labels = "auto",
                 rel_widths = c(0.45, 0.55))

both.top <- plot_grid(toppest, top, ncol = 1, rel_heights = c(0.4, 0.6))

plot.merged <- plot_grid(both.top, plot.keyword.per.model, ncol = 1,
                         labels = c("", "c"), rel_heights = c(0.4, 0.6))

plot.merged
```


a
Nº papers
0
100
200
1990
2000
2010
2020
Year
b
0
20
40
60
1990
2000
2010
2020
Year
DBHM
GR4J
H08
HBV
HYPE
JULES−W1
LPJmL
MATSIRO
MHM
MPI−HM
ORCHIDEE
PCR−GLOBWB
SACRAMENTO
SWAT

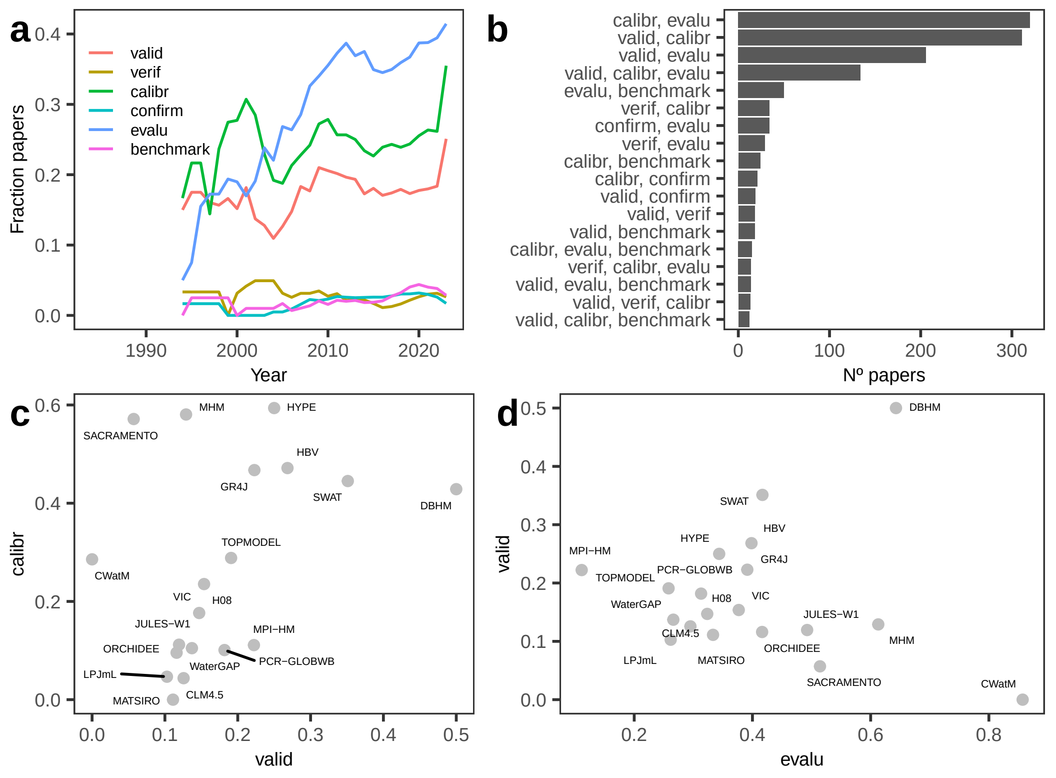

a
Fraction papers
0.0
0.1
0.2
0.3
0.4
valid
verif
calibr
confirm
evalu
benchmark
1990
2000
2010
2020
Year
b
calibr, evalu
valid, calibr
valid, evalu
valid, calibr, evalu
evalu, benchmark
verif, calibr
confirm, evalu
verif, evalu
calibr, benchmark
calibr, confirm
valid, confirm
valid, verif
valid, benchmark
calibr, evalu, benchmark
verif, calibr, evalu
valid, evalu, benchmark
valid, verif, calibr
valid, calibr, benchmark
0
100
200
300
Nº papers


c
valid
DBHM
SWAT
HBV
HYPE
GR4J
MPI−HM
TOPMODEL
PCR−GLOBWB
VIC
H08
WaterGAP
MHM
CLM4.5
JULES−W1
ORCHIDEE
MATSIRO
LPJmL
SACRAMENTO
CWatM
0.0
0.2
0.4
verif
MHM
DBHM
GR4J
TOPMODEL
HBV
JULES−W1
SACRAMENTO
SWAT
CLM4.5
VIC
PCR−GLOBWB
ORCHIDEE
LPJmL
WaterGAP
MPI−HM
MATSIRO
HYPE
H08
CWatM
0.00
0.05
0.10
calibr
HYPE
MHM
SACRAMENTO
HBV
GR4J
SWAT
DBHM
TOPMODEL
CWatM
VIC
H08
JULES−W1
MPI−HM
WaterGAP
PCR−GLOBWB
ORCHIDEE
LPJmL
CLM4.5
MATSIRO
0.0
0.2
0.4
0.6
confirm
MATSIRO
H08
DBHM
MHM
HYPE
PCR−GLOBWB
CLM4.5
GR4J
ORCHIDEE
TOPMODEL
SWAT
LPJmL
HBV
WaterGAP
VIC
JULES−W1
SACRAMENTO
MPI−HM
CWatM
0.00
0.05
0.10
evalu
CWatM
DBHM
MHM
SACRAMENTO
JULES−W1
SWAT
ORCHIDEE
HBV
GR4J
VIC
HYPE
MATSIRO
H08
PCR−GLOBWB
CLM4.5
WaterGAP
LPJmL
TOPMODEL
MPI−HM
0.0
0.5
benchmark
MHM
CWatM
SACRAMENTO
GR4J
CLM4.5
MATSIRO
LPJmL
PCR−GLOBWB
ORCHIDEE
H08
VIC
HBV
JULES−W1
TOPMODEL
SWAT
WaterGAP
MPI−HM
HYPE
DBHM
0.00
0.05
0.10
0.15
Fraction papers

```
toppest
```



```r
scatter.plots <- plot_grid(plot.valid.calibr, plot.evalu.valid, ncol = 2,
                           labels = c("c", "d"), rel_widths = c(0.46, 0.54))

plot_grid(top, scatter.plots, ncol = 1, rel_heights = c(0.5, 0.5))
```

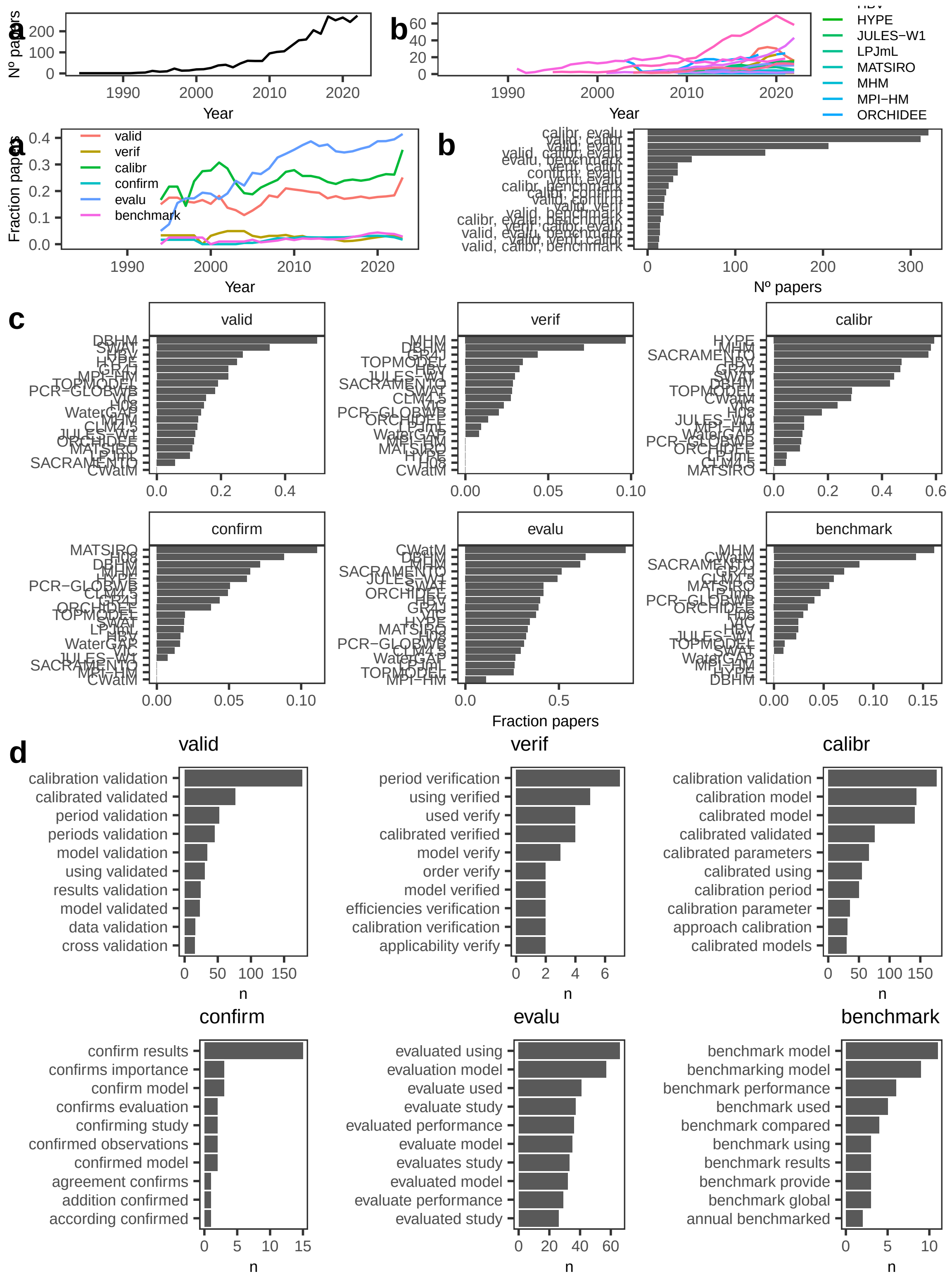


```r
# TOKEN ANALYSIS ##############################################################

# 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.cleaned, token = "ngrams", n = N.tokens) %>%
    separate(bigram, into = c("word1", "word2"), sep = " ") %>%
    data.table() %>%
    .[, `:=`(word1= pmin(word1, word2), word2 = pmax(word1, word2))] %>%
    count(word1, word2, sort = TRUE) %>%
    unite(., col = "bigram", c("word1", "word2"), sep = " ") %>%
    data.table()

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

  # 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)] %>%
    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 = 9),
          axis.text.y = element_text(size = 7))

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

  out <- list(output.dt, plot.token)
  names(out) <- c("data", "token")

  return(out)

}
```

```
# RUN TOKEN ANALYSIS ############################################################

N.tokens <- 2
```

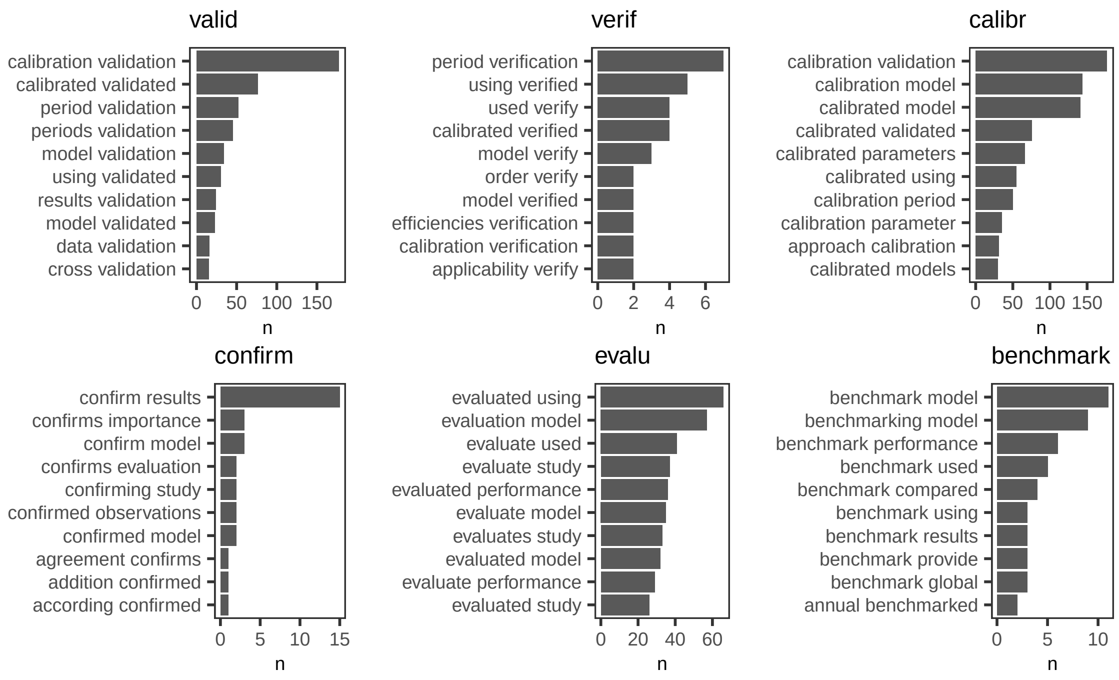
```r
token.dt <- list()

for (j in keywords.stemmed) {

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

}
```

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

plot.tokens <- plot_grid(token.dt$valid[[2]], token.dt$verif[[2]], token.dt$calibr[[2]],
                         token.dt$confirm[[2]], token.dt$evalu[[2]], token.dt$benchmark[[2]],
                         ncol = 3)

plot_grid(plot.merged, plot.tokens, ncol = 1, labels = c("", "d"),
          rel_heights = c(0.57, 0.43))
```



a
Nº papers
Year
b
HYPE
JULES−W1
LPJmL
MATSIRO
MHM
MPI−HM
ORCHIDEE
Fraction papers
valid
verif
calibr
confirm
evalu
benchmark
calibr, evalu
valid, calibr
valid, evalu
valid, calibr, evalu
evalu, benchmark
Nº papers
c
valid
verif
calibr
confirm
evalu
benchmark
Fraction papers
d
valid
calibration validation
calibrated validated
period validation
periods validation
model validation
using validated
results validation
model validated
data validation
cross validation
verif
period verification
using verified
used verify
calibrated verified
model verify
order verify
model verified
efficiencies verification
calibration verification
applicability verify
calibr
calibration validation
calibration model
calibrated model
calibrated validated
calibrated parameters
calibrated using
calibration period
calibration parameter
approach calibration
calibrated models
confirm
confirm results
confirms importance
confirm model
confirms evaluation
confirming study
confirmed observations
confirmed model
agreement confirms
addition confirmed
according confirmed
evalu
evaluated using
evaluation model
evaluate used
evaluate study
evaluated performance
evaluate model
evaluates study
evaluated model
evaluate performance
evaluated study
benchmark
benchmark model
benchmarking model
benchmark performance
benchmark used
benchmark compared
benchmark using
benchmark results
benchmark provide
benchmark global
annual benchmarked
n

```
plot.tokens
```



## 1 Close reading

```
# KEYWORDS TO LOOK FOR #########################################################

keywords_selected <- c(
  "benchmark", "calibrate", "confirm", "evaluate", "validate", "verify",
  "benchmarks", "calibrates", "confirms", "evaluates", "validates", "verifies",
  "benchmarked", "calibrated", "confirmed", "evaluated", "validated", "verified",
  "benchmarking", "calibrating", "confirming", "evaluating", "validating", "verifying",
  "calibration", "confirmation", "evaluation", "validation", "verification",
  "calibrations", "confirmations", "evaluations", "validations", "verifications",
  "benchmarkable", "calibrative", "confirmative", "evaluative", "validative", "verificative",
  "calibratively", "confirmatively", "evaluatively", "validatively", "verificatively"
)

keywords_selected_stemmed <- unique(stemDocument(keywords_selected))[-c(1, 3, 6)]

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

list.close.reading <- data.table(read.xlsx("dt.papers.close.reading.xlsx"))

dt.students <- data.table(read.xlsx("validation_work_students.xlsx")) %>%
  .[, title.large:= tolower(title)]

dt.close.reading <- merge(list.close.reading[, .(Model, title.large)],
      dt.students[, .(doi, title.large, paragraph)], by = "title.large") %>%
  .[, title.large:= tolower(title.large)] %>%
```

```r
  .[, paragraph.clean:= clear_text(paragraph)]

total.models <- dt.close.reading[, .N, .(Model, title.large)] %>%
  .[, .(Model)] %>%
  .[, .(total.papers = .N), Model]

# COUNT KEYWORDS ###############################################################

out <- out.stemmed <- list()

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

  pattern <- paste0("\\b", keywords_selected[i], "\\b")

  out[[i]] <- dt.close.reading[, lapply(.SD, function(x)
    str_count(x, pattern)), .SDcols = "paragraph"]

}

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


  out.stemmed[[i]] <- dt.close.reading[, lapply(.SD, function(x)
    str_count(x, keywords_selected_stemmed[i])), .SDcols = "paragraph"]

}

# ARRANGE DATA #################################################################

dt.keywords <- do.call(cbind, out) %>%
  data.table()

dt.keywords.stemmed <- do.call(cbind, out.stemmed) %>%
  data.table()

colnames(dt.keywords) <- keywords_selected
colnames(dt.keywords.stemmed) <- keywords_selected_stemmed

full.dt.keywords <- cbind(dt.keywords, dt.keywords.stemmed)

vec.columns <- colSums(full.dt.keywords, na.rm = TRUE)

colnames.keywords <- names(vec.columns[!vec.columns == 0])

full.dt.close.reading <- cbind(dt.close.reading, full.dt.keywords) %>%
  merge(., total.models, by = "Model")
```

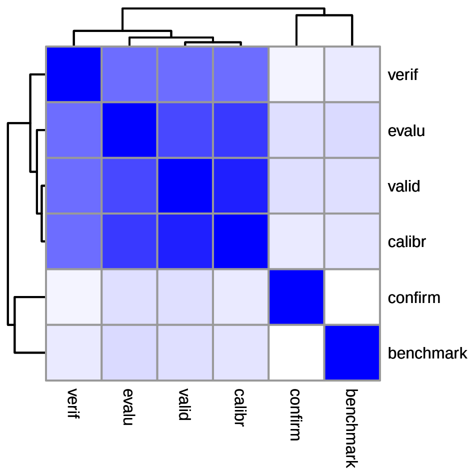
```r
# COSINE ANALYSIS ##################################################################

clean_text <- melt(full.dt.close.reading,
                   measure.vars = keywords.stemmed) %>%
  .[!value == 0] %>%
  unnest_tokens(word, paragraph.clean) %>%
  anti_join(stop_words) %>%
  group_by(variable) %>%
  summarize(paragraph.clean = paste(word, collapse = " "))
```

```
## Joining with `by = join_by(word)`
```

```r
# Iterate over the text -------------------------------------------------------

it <- itoken(clean_text$paragraph.clean, progressbar = FALSE)

# Create vocabulary -----------------------------------------------------------

vocab <- create_vocabulary(it)

# Vectorize text --------------------------------------------------------------

vectorizer <- vocab_vectorizer(vocab)
dtm <- create_dtm(it, vectorizer)

# Apply TF-IDF transformation -------------------------------------------------

tfidf <- TfIdf$new()
dtm_tfidf <- tfidf$fit_transform(dtm)

# Compute cosine similarity matrix --------------------------------------------

similarity_matrix <- sim2(dtm_tfidf, method = "cosine", norm = "l2")

# Convert the similarity matrix into a tidy format_----------------------------

similarity_df <- as.data.frame(as.table(as.matrix(similarity_matrix)))
colnames(similarity_df) <- c("Paragraph1", "Paragraph2", "Similarity")

# Filter for term-specific comparisons if needed ------------------------------

similarity_df <- similarity_df %>%
  filter(Paragraph1 != Paragraph2) %>%
  arrange(desc(Similarity))

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

```
heatmap_data <- similarity_matrix
rownames(heatmap_data) <- clean_text$variable
colnames(heatmap_data) <- clean_text$variable
```

```
plot.heatmap <- pheatmap::pheatmap(as.matrix(heatmap_data),
              color = colorRampPalette(c("white", "blue"))(50),
              fontsize_row = 6,
              fontsize_col = 6,
              treeheight_row = 10,
              treeheight_col = 10,
              legend = FALSE)


plot.heatmap
```

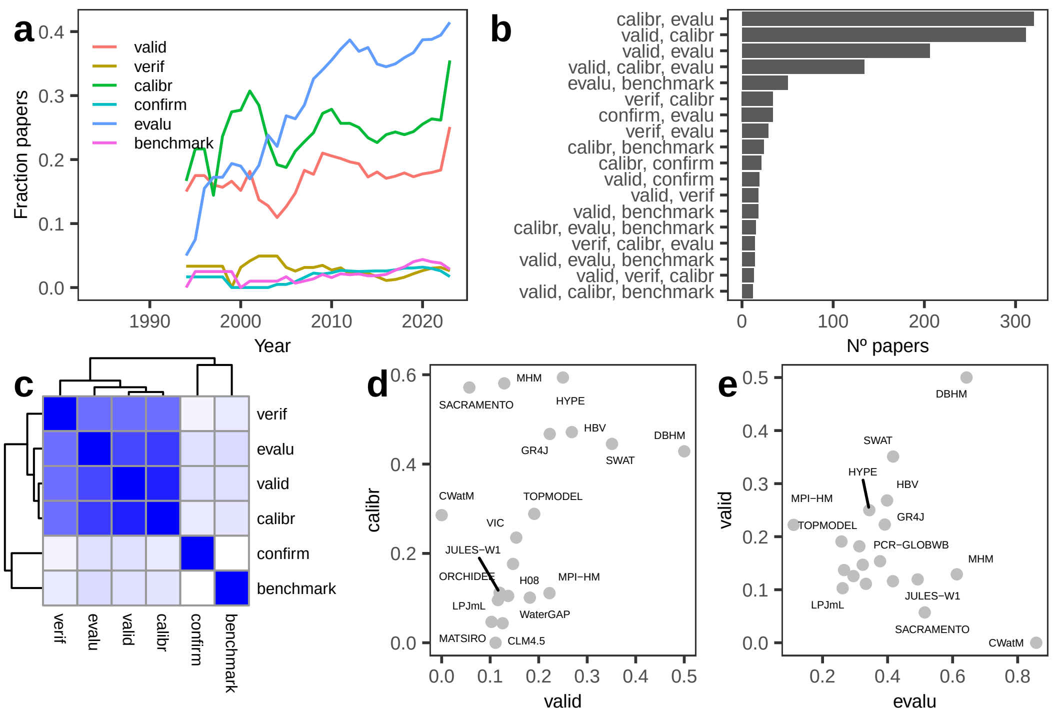


```
grob <- plot.heatmap$gtable
```

```
scatter.plots <- plot_grid(ggdraw(grob), plot.valid.calibr, plot.evalu.valid, ncol = 3,
                           labels = c("c", "d", "e"), rel_widths = c(0.33, 0.33, 0.33))

plot_grid(top, scatter.plots, ncol = 1, rel_heights = c(0.5, 0.5))
```

```
## Warning: ggrepel: 1 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```


a
Fraction papers
Year
valid
verif
calibr
confirm
evalu
benchmark
b
Nº papers
calibr, evalu
valid, calibr
valid, evalu
valid, calibr, evalu
evalu, benchmark
verif, calibr
confirm, evalu
verif, evalu
calibr, benchmark
calibr, confirm
valid, confirm
valid, verif
valid, benchmark
calibr, evalu, benchmark
verif, calibr, evalu
valid, evalu, benchmark
valid, verif, calibr
valid, calibr, benchmark
c
verif
evalu
valid
calibr
confirm
benchmark
d
calibr
valid
MHM
SACRAMENTO
HYPE
HBV
GR4J
SWAT
DBHM
CWatM
TOPMODEL
VIC
JULES−W1
ORCHIDEE
H08
MPI−HM
LPJmL
WaterGAP
MATSIRO
CLM4.5
e
valid
evalu
DBHM
SWAT
HYPE
HBV
MPI−HM
GR4J
TOPMODEL
PCR−GLOBWB
MHM
JULES−W1
LPJmL
SACRAMENTO
CWatM

```r
# SESSION INFORMATION ##########################################################

sessionInfo()
```

```
## R version 4.3.2 (2023-10-31)
## 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] benchmarkme_1.0.8          tidyquant_1.0.7
##  [3] quantmod_0.4.25            TTR_0.24.3
##  [5] PerformanceAnalytics_2.0.4 xts_0.13.1
##  [7] zoo_1.8-12                 ggrepel_0.9.5
##  [9] LSAfun_0.6.3               rgl_1.1.3
## [11] lsa_0.73.3                 SnowballC_0.7.1
## [13] cowplot_1.1.1              scales_1.3.0
## [15] tidytext_0.4.1             pdftools_3.3.3
## [17] tm_0.7-11                  NLP_0.2-1
## [19] openxlsx_4.2.5.2           lubridate_1.9.2
## [21] forcats_1.0.0              stringr_1.5.1
## [23] dplyr_1.1.4                purrr_1.0.2
## [25] readr_2.1.4                tidyr_1.3.0
## [27] tibble_3.2.1               ggplot2_3.4.4
## [29] tidyverse_2.0.0            data.table_1.14.99
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0      filehash_2.4-5        farver_2.1.1
##  [4] fastmap_1.1.1         janeaustenr_1.0.0     digest_0.6.34
##  [7] timechange_0.2.0      lifecycle_1.0.4       qpdf_1.3.2
## [10] tokenizers_0.3.0      magrittr_2.0.3        compiler_4.3.2
## [13] rlang_1.1.3           tools_4.3.2           utf8_1.2.4
## [16] sensobol_1.1.4        yaml_2.3.7            knitr_1.42
## [19] askpass_1.1           labeling_0.4.3        stopwords_2.3
## [22] htmlwidgets_1.6.2     curl_5.0.0            xml2_1.3.3
## [25] withr_3.0.0           grid_4.3.2            fansi_1.0.6
## [28] colorspace_2.1-0      iterators_1.0.14      tinytex_0.45
## [31] cli_3.6.2             rmarkdown_2.21        generics_0.1.3
## [34] tikzDevice_0.12.4     rstudioapi_0.15.0     httr_1.4.5
## [37] tzdb_0.3.0            base64enc_0.1-3       vctrs_0.6.5
## [40] Matrix_1.6-1.1        jsonlite_1.8.4        slam_0.1-50
## [43] hms_1.1.3             foreach_1.5.2         glue_1.7.0
## [46] benchmarkmeData_1.0.4 codetools_0.2-19      Quandl_2.11.0
## [49] stringi_1.8.3         gtable_0.3.4          quadprog_1.5-8
## [52] munsell_0.5.0         pillar_1.9.0          htmltools_0.5.5
## [55] R6_2.5.1              Rdpack_2.6            doParallel_1.0.17
## [58] evaluate_0.20         lattice_0.21-9        highr_0.10
## [61] rbibutils_2.2.16      Rcpp_1.0.12           zip_2.3.0
## [64] xfun_0.39             pkgconfig_2.0.3
```

```
## 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
```