Mind the hubris in mathematical modeling

R code

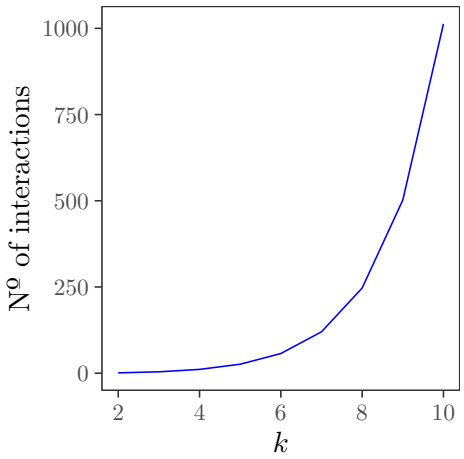
Arnald Puy

Contents

```r
# PRELIMINARY ----------------------------------------------------------------

# 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)
    }
  }
}

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.margin=margin(0, 0, 0, 0),
          legend.box.margin=margin(-7,-7,-7,-7),
          legend.key = element_rect(fill = "transparent",
                                    color = NA),
          strip.background = element_rect(fill = "white"))
}

# Load the packages
loadPackages(c("data.table", "tidyverse", "cowplot", "scales", "patchwork",
               "ggpubr", "benchmarkme", "parallel"))

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

checkpoint("2022-05-20",
           R.version ="4.2.0",
           checkpointLocation = getwd())
```

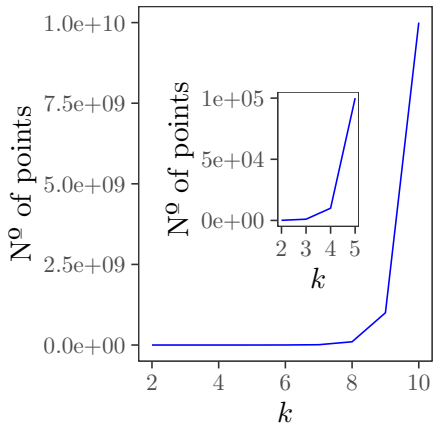
1 Explosion of the uncertainty space

```
# EXPLOSION OF THE UNCERTAINTY SPACE ------------------------------------------

# Define dimensions -------------------------------
k <- 2:10

# Compute number of interactions-------------------
x <- sapply(k, function(k) 2^k - k - 1)

a <- data.table(cbind(x, k)) %>%
  ggplot(., aes(k, x)) +
  geom_line(color = "blue") +
  labs(x = "$k$", y = "Nº of interactions") +
  theme_AP()

a
```

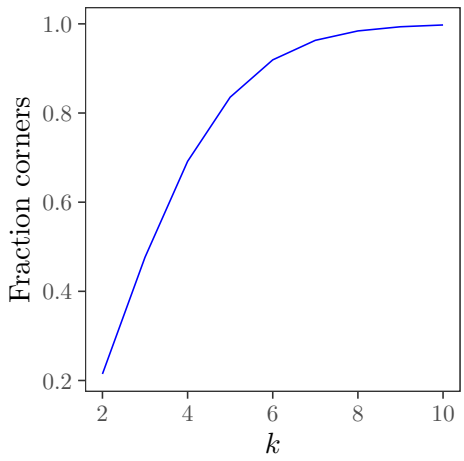



```
# Curse of dimensionality -----------------------------------------------------
N <- 10 # Sample density

out <- N^k

b <- cbind(k, out) %>%
  data.table() %>%
  ggplot(., aes(k, out)) +
  geom_line(color = "blue") +
  labs(x = "$k$", y = "Nº of points") +
  theme_AP()

inset.plot <- b +
  scale_x_continuous(limits = c(2, 5),
                     breaks = pretty_breaks(n = 3)) +
  scale_y_continuous(limits = c(1e+02, 1e+05),
                     breaks = pretty_breaks(n = 3)) +
```

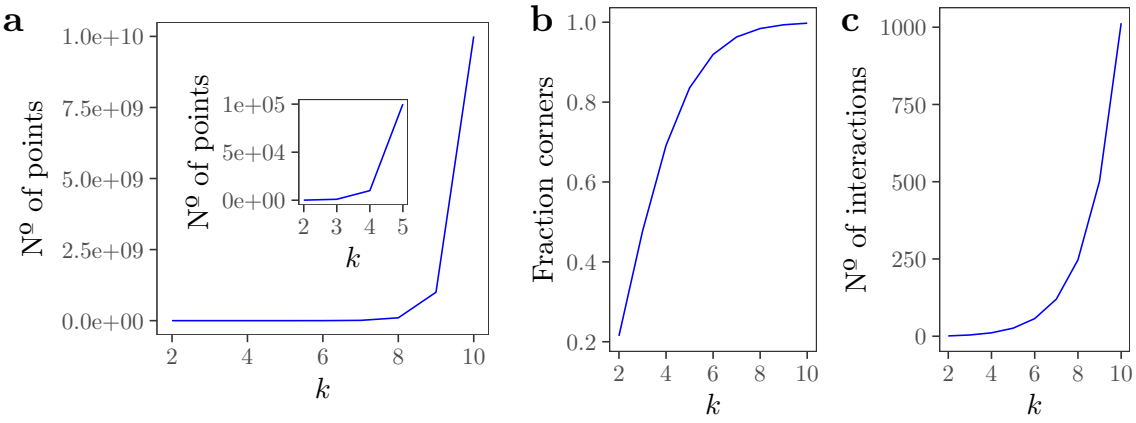
```
  labs(x = "", y = "") +
  labs(x = "$k$", y = "Nº of points")

b <- b +
  inset_element(inset.plot, 0.05, 0.15, 0.8, 0.8)

b
```




```
# Ratio of the hypersphere to the hypercube -----------------------------------

sphere_to_cube <- function(x) pi ^ ((x) / 2) * (0.5) ^ (x) / gamma(1 + x / 2)

out <- sapply(k, function(x) sphere_to_cube(x))

c <- data.table(cbind(k, out)) %>%
  .[, corner:= 1 - out] %>%
  ggplot(., aes(k, corner)) +
  geom_line(color = "blue") +
  labs(x = "$k$", y = "Fraction corners") +
  theme_AP()

c
```



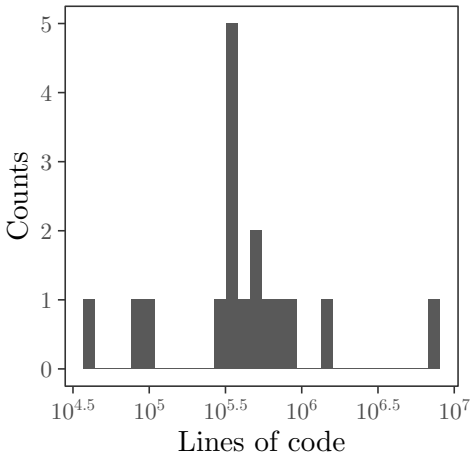
```
# MERGE PLOTS ------------------------------------------------------------------

plot_grid(b, c, a, ncol = 3, labels = "auto",
          rel_widths = c(0.47, 0.28, 0.28), align = "tb")
```



2 Blackboxing

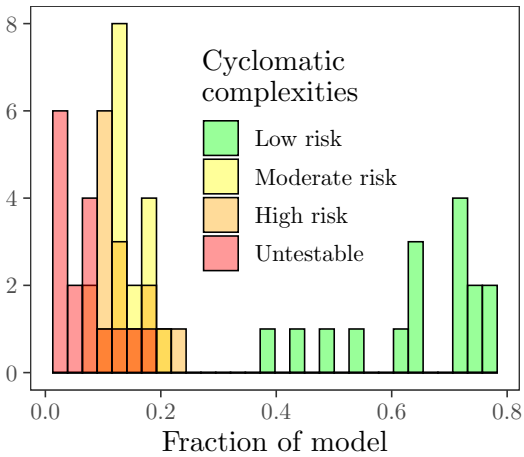
```
# LINES OF CODE ----------------------------------------------------------------

code <- fread("lines_code.csv")
colNames <- colnames(code)[-1]
code[, (colNames):= lapply(.SD, function(x) x * 1000), .SDcols = (colNames)]

# Plot -------------------------------------------------------------------------
code.plot <- code %>%
  ggplot(., aes(KLOC)) +
  geom_histogram() +
  scale_x_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", scales::math_format(10^.x))) +
  coord_cartesian(clip = "off") +
  labs(x = "Lines of code", y = "Counts") +
  theme_AP()

code.plot
```

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



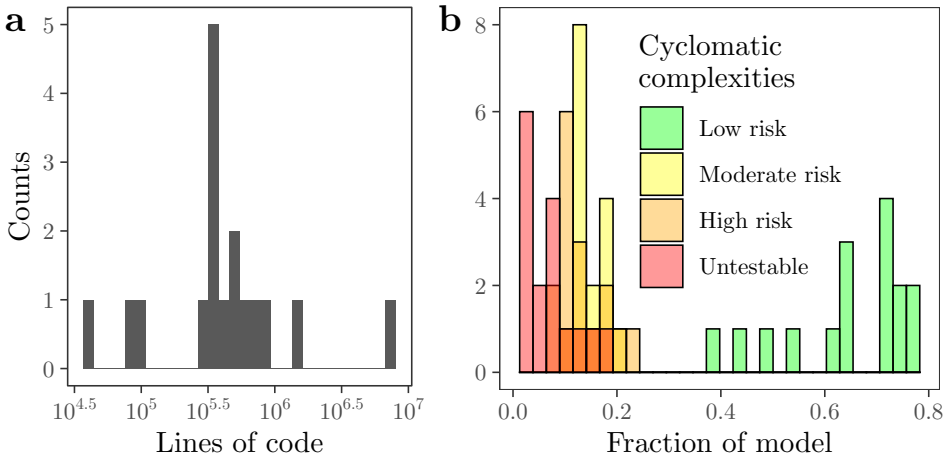

```
# CYCLOMATIC COMPLEXITIES ----------------------------------------------------

cyclomatic <- fread("cyclomatic_complexity.csv")
colNames <- colnames(cyclomatic)[-1]
new_colNames <- c("Low risk", "Moderate risk", "High risk", "Untestable")
cyclomatic[, total:= rowSums(.SD), .SDcols = colNames]
fraction <- cyclomatic[, lapply(.SD, function(x) x / total), .SDcols = colNames]
colnames(fraction) <- new_colNames

# Plot ---------------------------------
cyclomatic.plot <- melt(fraction, measure.vars = new_colNames,
     variable.name = "Cyclomatic \n complexities") %>%
  ggplot(., aes(value, fill = `Cyclomatic \n complexities`)) +
  scale_fill_manual(values = c("green", "yellow", "orange", "red")) +
  labs(x = "Fraction of model", y = "") +
  geom_histogram(alpha = 0.4, position = "identity", color = "black") +
  theme_AP() +
  theme(legend.position = c(0.55, 0.6))

cyclomatic.plot
```

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



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

plot_grid(code.plot, cyclomatic.plot, ncol = 2, labels = "auto",
          rel_widths = c(0.45, 0.55))
```

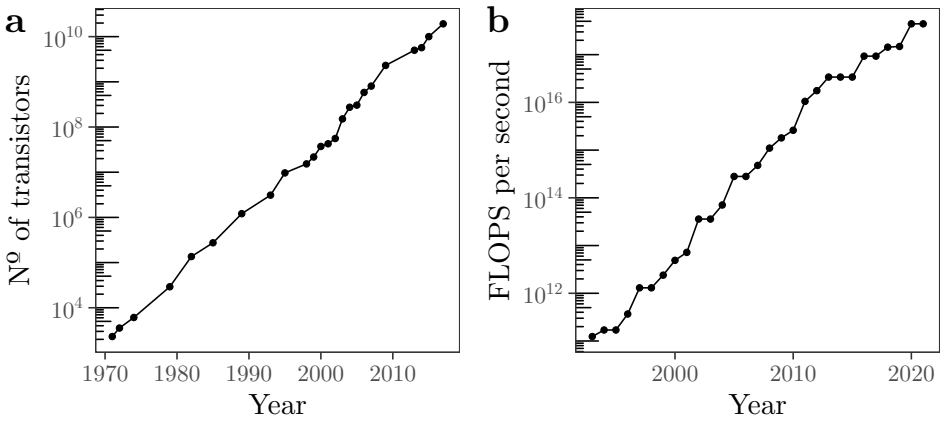
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



3 Computational exhaustion

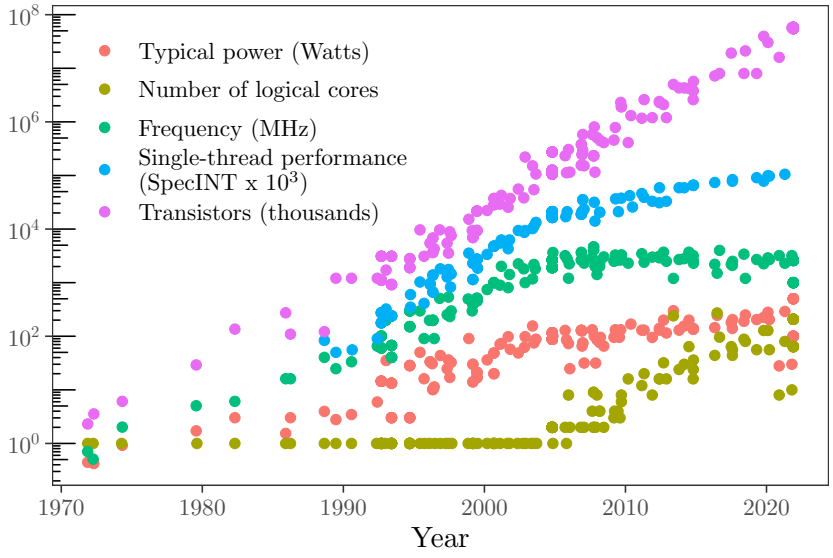
```
# MOORE'S LAW AND COMPUTATIONAL CAPACITY ---------------------------------------

transistors <- fread("transistors-per-microprocessor.csv")
supercomputers <- fread("supercomputer-power-flops.csv")

a <- transistors %>%
  ggplot(., aes(Year, `Transistors per microprocessor`)) +
  geom_line() +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", math_format(10^.x))) +
  annotation_logticks(sides = "l") +
  labs(x = "Year", y = "Nº of transistors") +
  geom_point(size = 0.8) +
  theme_AP()

b <- supercomputers %>%
  ggplot(., aes(Year, `Floating-Point Operations per Second`)) +
  geom_line() +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", math_format(10^.x))) +
  annotation_logticks(sides = "l") +
  labs(x = "Year", y = "FLOPS per second") +
  geom_point(size = 0.8) +
```

```
  theme_AP()

plot_grid(a, b, ncol = 2, labels = "auto")
```

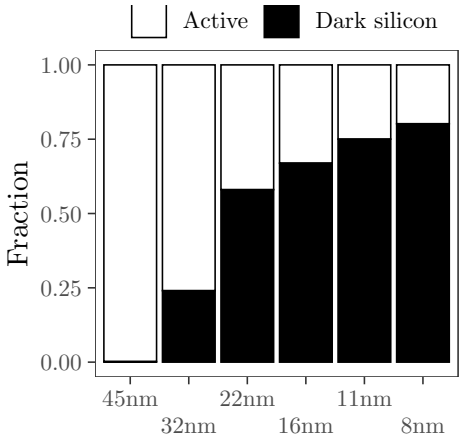



```
# 50 YEARS OF MICROPROCESSOR TREND DATA ----------------------------------------

watts <- fread("watts.txt", col.names = c("Year", "Typical power (Watts)"),
               colClasses = c("numeric", "numeric"))
cores <- fread("cores.txt", col.names = c("Year", "Number of logical cores"),
               colClasses = c("numeric", "numeric"))
frequency <- fread("frequency.txt", col.names = c("Year", "Frequency (MHz)"),
                   colClasses = c("numeric", "numeric"))
specint <- fread("specint.txt",
                 col.names = c("Year", "Single-thread performance \n (SpecINT x $10^3$)"),
                 colClasses = c("numeric", "numeric"))
transistors <- fread("transistors.txt", col.names = c("Year", "Transistors (thousands)"),
                     colClasses = c("numeric", "numeric"))

list_dt <- list(watts, cores, frequency, specint, transistors)

all <- Reduce(function(...) merge(..., all = TRUE), list_dt)

colNames_dt <- colnames(all)[-1]

# Plot
microprocessor.data <- melt(all, measure.vars = colNames_dt) %>%
  ggplot(., aes(Year, value, color = variable)) +
  geom_point() +
  scale_y_log10(breaks = trans_breaks("log10", function(x) 10^x),
                labels = trans_format("log10", math_format(10^.x))) +
  annotation_logticks(sides = "l") +
  labs(x = "Year", y  = "") +
  scale_color_discrete(name = "") +
  theme_AP() +
  theme(legend.position = c(0.25, 0.78))
```

```
microprocessor.data
```

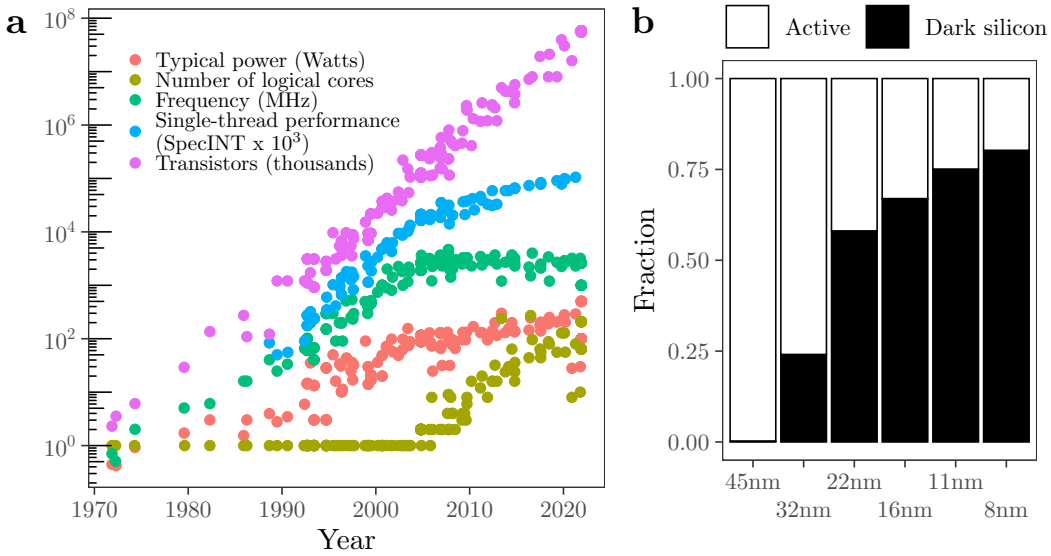



```
# FRACTION OF DARK SILICON AS A FUNCTION OF TECHNOLOGY -------------------------

dark_silicon <- fread("dark_silicon_percentage.csv")
colNames <- c("Size", "Active")
setnames(dark_silicon, c("V1", "V2"), colNames)

dark_silicon <- dark_silicon[, `Dark silicon`:= 1 - Active]

# PLOT
dark.silicon.plot <- melt(dark_silicon, measure.vars = c("Active", "Dark silicon")) %>%
  .[, Size:= factor(Size, levels = c("45nm", "32nm", "22nm",
                                     "16nm", "11nm", "8nm"))] %>%
  ggplot(., aes(Size, value, fill = variable)) +
  scale_fill_manual(values = c("white", "black"), name = "") +
  geom_bar(stat = "identity", position = "fill", color = "black") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  labs(x = "", y = "Fraction") +
  theme_AP() +
  theme(legend.position = "top")

dark.silicon.plot
```




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

ggarrange(microprocessor.data +
            theme(legend.text = element_text(size = 8),
                  legend.position = c(0.36, 0.83),
                  legend.key.size = unit(0.2, "lines")),
          dark.silicon.plot +
            theme(legend.box.margin=margin(-2,-7,-7,-7)), ncol = 2,
                  labels = "auto", widths = c(0.65, 0.45))
```

```
## Warning: Removed 312 rows containing missing values (geom_point).
```



4 Model attachment

```
# READ IN DATASET --------------------------------------------------------------

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

file <- "full.dt.csv"
```

```r
full.dt <- fread(file = file)

# Create vector with name of models (file "full.dt.csv" is already organized
# following the order of this vector) ----------------------------------------

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

# Analyse dataset ------------------------------------------------------------

dt.use <- full.dt[, .N, .(Model, university.first)] %>%
  dcast(., university.first~ Model, value.var = "N")

for(j in seq_along(dt.use)){
  set(dt.use, i = which(is.na(dt.use[[j]]) & is.numeric(dt.use[[j]])), j = j, value = 0)
}

# Total number each institute uses a model
dt.use[, total:= rowSums(.SD), .SDcols = models]

# Turn lowercase of institutions except acronyms
exceptions <- c("USA", "UK", "CNRS", "IIASA", "DOE", "PCSHE", "IIT", "NCAR",
                "NOAA", "KICT", "CSIRO", "INRAE")

pattern <- sprintf("(?:%s)(*SKIP)(*FAIL)|\\b([A-Z])(\\w+)",
                   paste0(exceptions, collapse = "|"))

dt.use <- dt.use[, university.first:= gsub(pattern, "\\1\\L\\2", university.first, perl = TRUE)

# Calculate fraction of studies with attachment
tmp <- dt.use[, lapply(.SD, function(x) x / total), .SDcols = models] %>%
  .[, lapply(.SD, round, 2), .SDcols = models]

# RETRIEVE MAX VALUES PER INSTITUTE ############################################

matrix.values <- as.matrix(tmp)
colIndex <- apply(matrix.values, 1, which.max)

# Add column for totals
total.studies <- dt.use$total
total.dt <- cbind(matrix.values, total.studies)

out <- vector()
for(i in 1:length(colIndex)) {
  out[i] <- matrix.values[[i, colIndex[i]]]
}
```

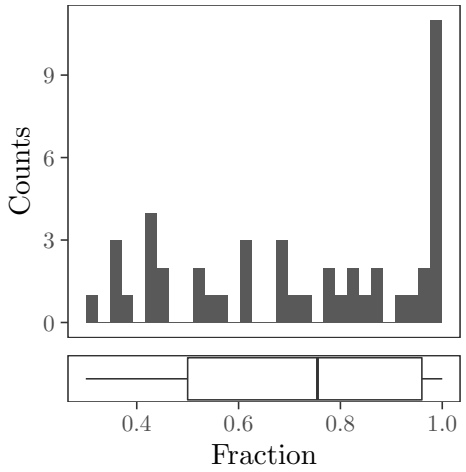
```r
list_institutes <- dt.use$university.first

dt.complete <- cbind(matrix.values, out, total.studies) %>%
  data.table()  %>%
  cbind(list_institutes, .)

# Compute some statistics on the vector for institutes with more than 5 studies
f <- c(mean, median, min, max)
sapply(f, function(f) f(dt.complete[, out][total.studies >= 5], na.rm = TRUE))
```

```
## [1] 0.7143478 0.7550000 0.3000000 1.0000000
```

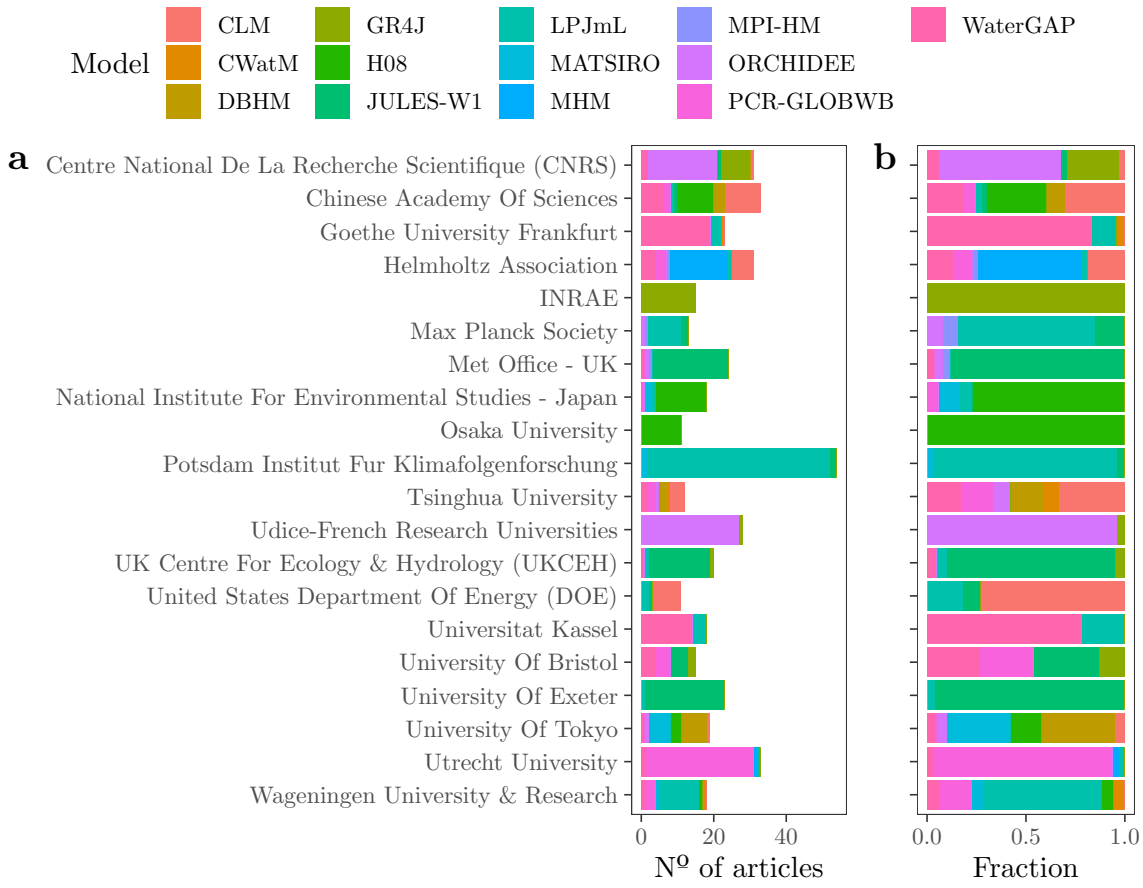
```r
# PLOT ------------------------------------------------------------------------

histo.plot <- dt.complete[total.studies >= 5, .(out)] %>%
  ggplot(., aes(out)) +
  geom_histogram() +
  labs(x = "", y = "Counts") +
  theme_AP() +
  theme(plot.margin = margin(b = -0.5, unit = "cm"),
        axis.ticks.x = element_blank(),
        axis.text.x = element_blank())

box.plot <- dt.complete[total.studies >= 5, .(out)] %>%
  ggplot(., aes(out)) +
  geom_boxplot() +
  scale_y_continuous(breaks = pretty_breaks(n = 3)) +
  labs(x = "Fraction", y = "") +
  theme_AP() +
  theme(axis.ticks.y = element_blank(),
        axis.text.y = element_blank())

plot.hist <- plot_grid(histo.plot, box.plot, ncol = 1, rel_heights = c(0.7, 0.3), align = "v")
```

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

```r
plot.hist
```

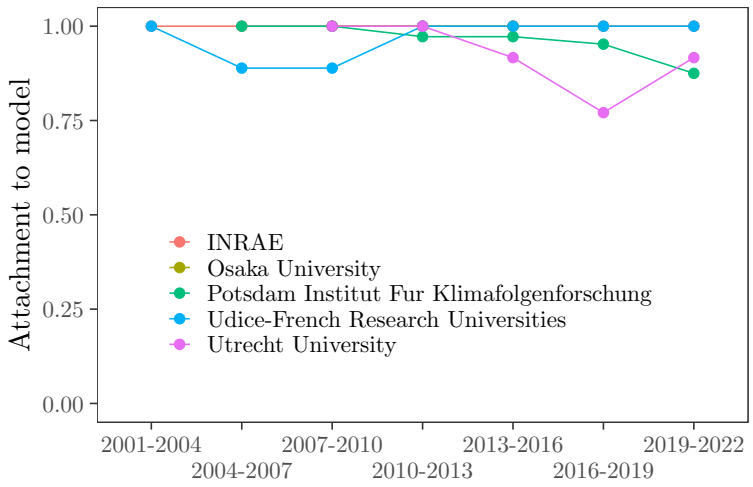



```
# PLOT USE OF MODEL PER INSTITUTE ------------------------------------------

# Plot only the first 20

a <- dt.use[order(-total.studies)][1:20] %>%
  melt(., measure.vars = models) %>%
  na.omit() %>%
  .[, variable:= factor(variable, levels = sort(models))] %>%
  ggplot(., aes(value, university.first, fill = variable)) +
  scale_y_discrete(limits = rev) +
  labs(x = "Nº of articles", y = "") +
  scale_fill_discrete(name = "Model") +
  geom_bar(position = "stack", stat = "identity") +
  theme_AP() +
  theme(legend.position = "none",
        axis.text.y = element_text(size = 9))

b <- dt.complete[order(-total.studies)][1:20] %>%
  melt(., measure.vars = models)%>%
  na.omit() %>%
  .[, variable:= factor(variable, levels = sort(models))] %>%
  ggplot(., aes(value, list_institutes, fill = variable)) +
  scale_y_discrete(limits = rev) +
  labs(x = "Fraction", y = "") +
  scale_fill_discrete(name = "Model") +
  scale_x_continuous(breaks = pretty_breaks(n = 3)) +
  geom_bar(position = "fill", stat = "identity") +
  theme_AP() +
  theme(axis.text.y = element_blank(),
        legend.position = "none")

legend <- get_legend(a + theme(legend.position = "top"))
bottom <- plot_grid(a, b, ncol = 2, labels = "auto", rel_widths = c(0.75 , 0.25))
ggarrange(legend, bottom, nrow = 2, heights = c(0.15, 0.85))
```

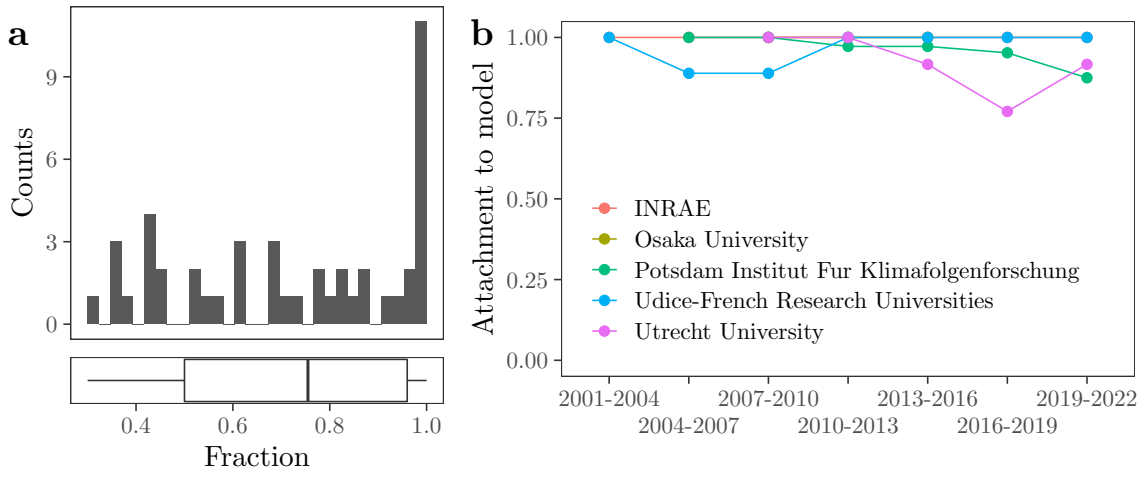



```
# STUDY OF MODEL ATTACHMENT THROUGH YEARS ------------------------------------

vector_institutes <- dt.complete[total.studies >= 11 & out >= 0.9][, list_institutes]
full.dt[, university.first:= gsub(pattern, "\\1\\L\\2", university.first, perl = TRUE)]
prove <- full.dt[university.first %chin% vector_institutes, ] %>%
  .[, .(university.first, year, Model)]

total.per.year <- prove[, .(total = .N), .(university.first, year)]

vec_year <- seq(min(full.dt$year, na.rm = TRUE), max(full.dt$year, na.rm = TRUE), 3)

time_frames <- names_slots <- list()
for (i in 1:(length(vec_year) - 1)) {

  time_frames[[i]] <- prove[year >= vec_year[i] & year <= vec_year[i + 1]] %>%
    .[, .N, .(university.first, year, Model)] %>%
    dcast(., year + university.first ~ Model, value.var = "N") %>%
    merge(total.per.year, ., by = c("university.first", "year")) %>%
    .[, (colnames(.)[-c(1, 2)]):= lapply(.SD, function(x) x / total),
      .SDcols = colnames(.)[-c(1, 2)]] %>%
    melt(., measure.vars = colnames(.)[-c(1:3)])

  names_slots[[i]] <- paste(vec_year[i], vec_year[i + 1], sep = "-")
```

```
}

names(time_frames) <- names_slots
out.dt <- rbindlist(time_frames, idcol = "Years")
out.final <- list()

for (i in vector_institutes) {
  if (i == "INRAE") {

    out.final[[i]] <- out.dt[university.first == i & variable == "GR4J"]
  } else if (i == "Potsdam Institut Fur Klimafolgenforschung") {
    out.final[[i]] <- out.dt[university.first == i & variable == "LPJmL"]
  } else if (i == "Udice-French Research Universities") {
    out.final[[i]] <- out.dt[university.first == i & variable == "ORCHIDEE"]
  } else if(i == "University of Exeter") {
    out.final[[i]] <- out.dt[university.first == i & variable == "JULES-W1"]
  } else if(i == "Utrecht University") {
    out.final[[i]] <- out.dt[university.first == i & variable == "PCR-GLOBWB"]
  } else if ( i == "Osaka University") {
    out.final[[i]] <- out.dt[university.first == i & variable == "H08"]
  }
}

years.dt <- rbindlist(out.final, idcol = "Institution") %>%
  .[, .(mean = mean(value, na.rm = TRUE), sd = sd(value, na.rm = TRUE)), .(university.first, Ye

plot.attachment.years <- ggplot(years.dt, aes(Years, mean, color = university.first, group = un
  geom_line() +
  geom_point() +
  scale_y_continuous(limits = c(0, 1)) +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  scale_color_discrete(name = "") +
  labs(x = "", y = "Attachment to model") +
  theme_AP() +
  theme(legend.position = c(0.48, 0.36),
        legend.key.size = unit(0.8, "lines"))

plot.attachment.years
```




```
# MERGE ------------------------------------------------------------------------

plot_grid(plot.hist, plot.attachment.years, ncol = 2, labels = "auto", rel_widths = c(0.4, 0.6)
```



5 Session information

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

sessionInfo()
```

```
## R version 4.2.0 (2022-04-22)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Monterey 12.3.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-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
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] checkpoint_1.0.2  benchmarkme_1.0.7 ggpubr_0.4.0      patchwork_1.1.1
##  [5] scales_1.2.0      cowplot_1.1.1     forcats_0.5.1     stringr_1.4.0
##  [9] dplyr_1.0.9       purrr_0.3.4       readr_2.1.2       tidyr_1.2.0
## [13] tibble_3.1.7      ggplot2_3.3.6     tidyverse_1.3.1   data.table_1.14.2
##
## loaded via a namespace (and not attached):
##  [1] lattice_0.20-45       lubridate_1.8.0       foreach_1.5.2
##  [4] assertthat_0.2.1      digest_0.6.29         utf8_1.2.2
##  [7] R6_2.5.1              cellranger_1.1.0      backports_1.4.1
## [10] reprex_2.0.1          evaluate_0.15         httr_1.4.3
## [13] pillar_1.7.0          rlang_1.0.2           readxl_1.4.0
## [16] rstudioapi_0.13       car_3.0-13            Matrix_1.4-1
## [19] tikzDevice_0.12.3.1   rmarkdown_2.14        munsell_0.5.0
## [22] broom_0.8.0           compiler_4.2.0        modelr_0.1.8
## [25] xfun_0.31             pkgconfig_2.0.3       htmltools_0.5.2
## [28] tidyselect_1.1.2      codetools_0.2-18      fansi_1.0.3
## [31] crayon_1.5.1          tzdb_0.3.0            dbplyr_2.1.1
## [34] withr_2.5.0           grid_4.2.0            jsonlite_1.8.0
## [37] gtable_0.3.0          lifecycle_1.0.1       DBI_1.1.2
## [40] magrittr_2.0.3        cli_3.3.0             stringi_1.7.6
## [43] carData_3.0-5         ggsignif_0.6.3        fs_1.5.2
## [46] doParallel_1.0.17     benchmarkmeData_1.0.4 xml2_1.3.3
## [49] ellipsis_0.3.2        generics_0.1.2        vctrs_0.4.1
## [52] iterators_1.0.14      tools_4.2.0           glue_1.6.2
## [55] hms_1.1.1             abind_1.4-5           fastmap_1.1.0
## [58] yaml_2.3.5            colorspace_2.0-3      rstatix_0.7.0
```

```
## [61] filehash_2.4-3        rvest_1.0.2           knitr_1.39
## [64] haven_2.5.0
```

```
## 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
```