Mind the hubris in mathematical modeling

R code

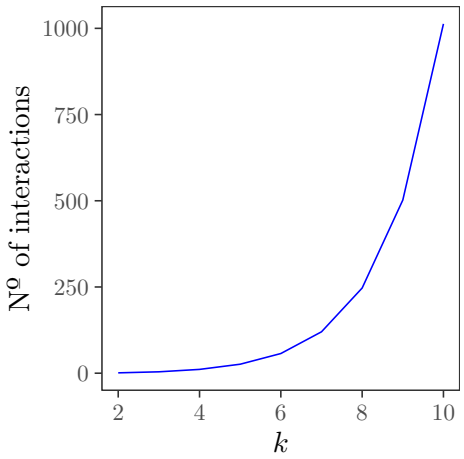
Arnald Puy

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"))

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

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

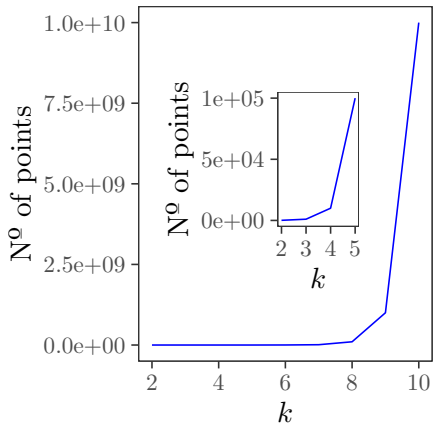
1 Explosion of the uncertainty space

```
# 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
```

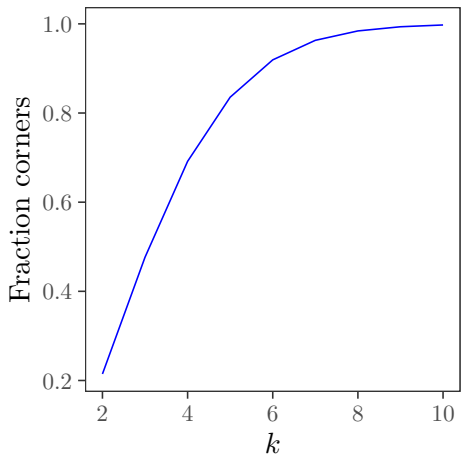



```
# Curse of dimensionality -----------------------------------------------------
N <- 10 # Sample density

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)) +
```

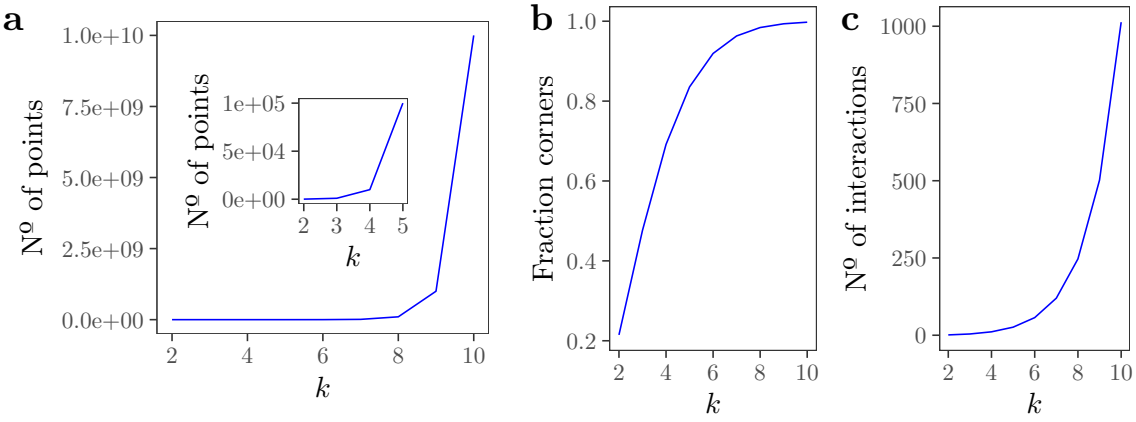
```
  labs(x = "", y = "") +
  labs(x = "$k$", y = "Nº of points")

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
```



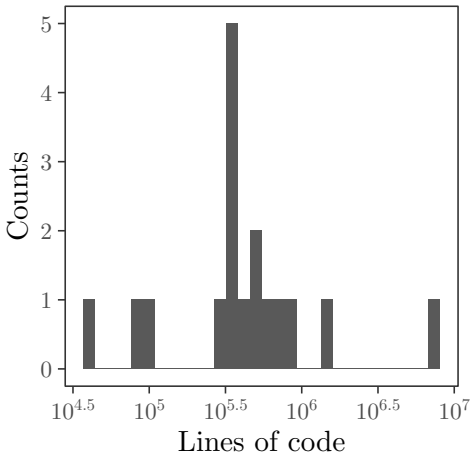
```
# MERGE PLOTS ------------------------------------------------------------------

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



2 Black boxing processes

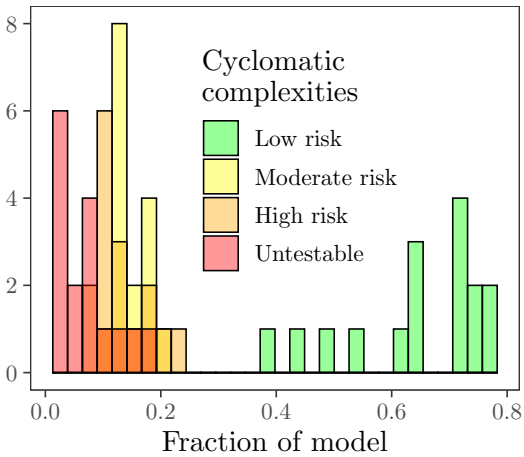
```
# LINES OF CODE ----------------------------------------------------------------

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`.
```



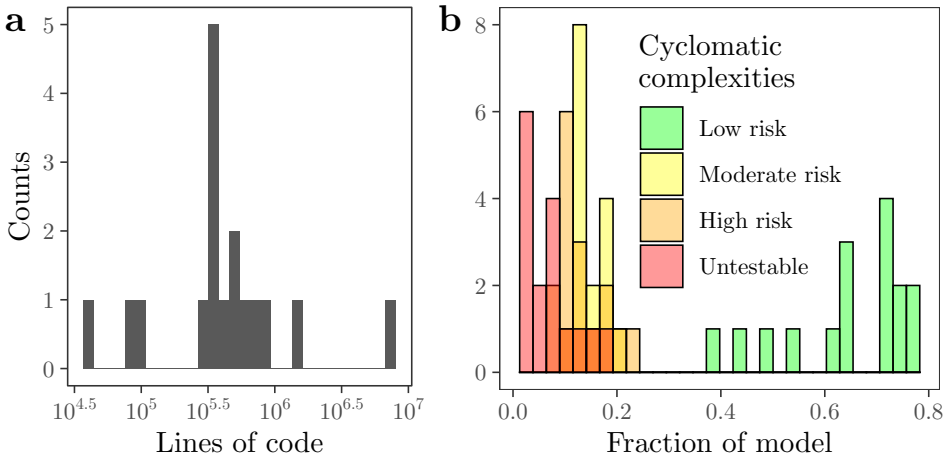

```
# CYCLOMATIC COMPLEXITIES ----------------------------------------------------

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))
```

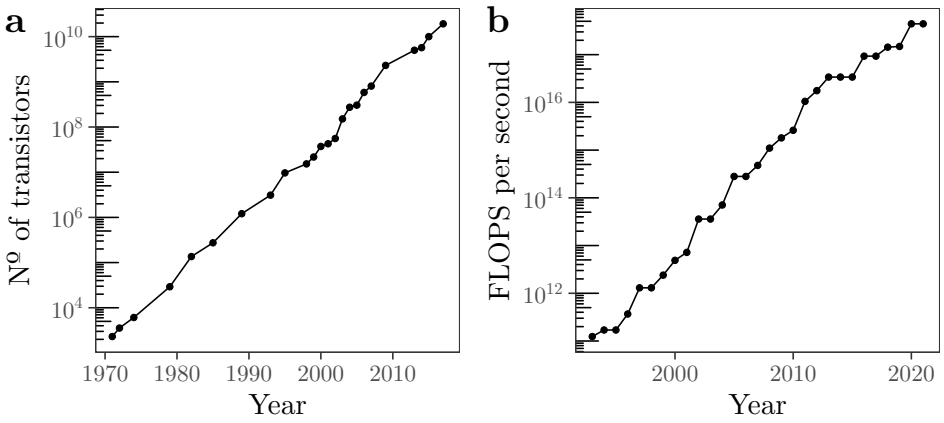
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



3 Physical limits to computation

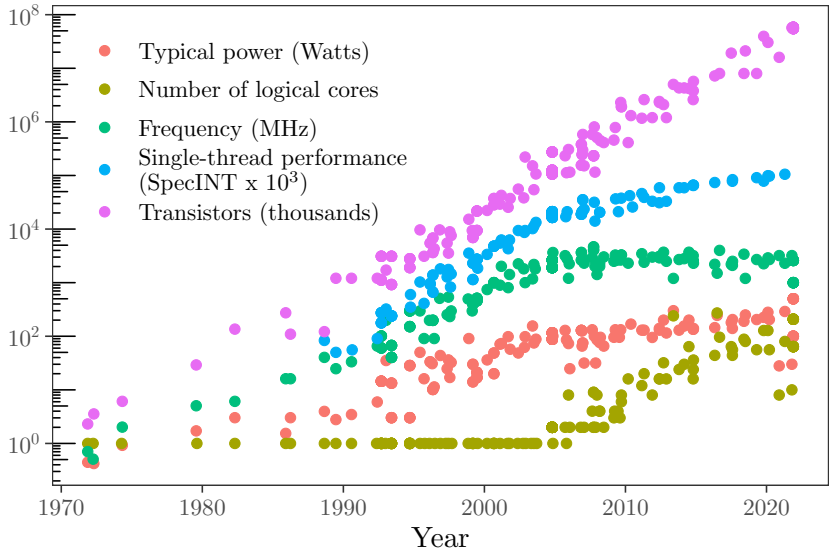
```
# MOORE'S LAW AND COMPUTATIONAL CAPACITY ---------------------------------------

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")
```

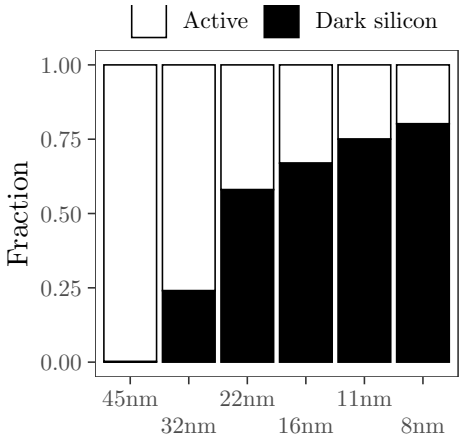



```
# 50 YEARS OF MICROPROCESSOR TREND DATA ----------------------------------------

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).
```