Stabilization Analyses for Characters

Chris Bentz

03/01/2023

Load libraries

If the libraries are not installed yet, you need to install them using, for example, the command: install.packages("ggplot2"). For the Hrate package this is different, since it comes from github. The devtools library needs to be installed, and then the install_github() function is used.

```
library(stringr)
library(ggplot2)
library(plyr)
library(entropy)
library(ggExtra)
library(gsubfn)
```

```
## Loading required package: proto
```

```
# library(devtools)
# install_github("dimalik/Hrate")
library(Hrate)
```

List files

Create list with all the files in the directory "corpus".

```
# give file paths to the files to be processed
file.list <- list.files(path = "~/Github/NaLaFi/data/",
                        recursive = T, full.names = T)
head(file.list)
```

```
## [1] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_10"
## [2] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_11"
## [3] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_12"
## [4] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_13"
## [5] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_14"
## [6] "/home/chris/Github/NaLaFi/data//generated/random/random_ran_15"
```

```
length(file.list)
```

```
## [1] 330
```

Stabilization analysis per file

```
# set counter
counter = 0
# set the maximal number of units (n), and the stepsize for stabilization analysis
# (i.e. in steps of how many units are values calculated?)
n = 100
stepsize = 10
# initialize dataframe to append results to
stabilization.df <- data.frame(filename = character(0), subcorpus = character(0),
                               code = character(0), huni.chars = numeric (0),
                               hrate.chars = numeric(0), ttr.chars = numeric(0),
                               rm.chars = numeric(0), units = numeric(0))
# start time
start_time <- Sys.time()
for (file in file.list)
{
  try({ # if the processing failes for a certain file, there will be no output for this file,
  # but the try() function allows the loop to keep running

  # basic processing
  # loading textfile
  textfile <- scan(file, what = "char", quote = "",
                   comment.char = "", encoding = "UTF-8", sep = "\n" , skip = 7, nmax = 20)
  # skip 7 first lines, nmax gives the maximum number of lines to be read,
  # note that reading more lines will considerably increase processing time.
  # remove annotations marked by '<>'
  textfile <- gsub("<.*>","",textfile)
  # print(head(textfile))
  # get filename
  filename <- basename(file)
  #print(filename) # for visual inspection
  # get subcorpus category
  subcorpus <- sub("_.*", "", filename)
  # print(subcorpus) # for visual inspection
  # get the three letter identification code + the running number
  code <- substring(substring(filename, regexpr("_", filename) + 1), 1, 8)

  # Split into individual characters/signs
  # remove tabs and parentheses, as well as star signs `*' and plus signs `+´
  # note that this might have to be tuned according to the text files included
  textfile <- str_replace_all(textfile, c("\\\t" = "", "\\(" = "", "\\)" = "",
                                       "\\]" = "", "\\[" = "",  "\\}" = "",
                                       "\\{" = "", "\\*" = "", "\\+" = ""))
  # split the textfile into individual utf-8 characters. Note that white spaces are
  # counted as utf-8 characters here.
  chars <- unlist(strsplit(textfile, ""))
  chars <- chars[1:n] # use only maximally n units
  chars <- chars[!is.na(chars)] # remove NAs for vectors which are already shorter
  # than n
  # chars <- chars[chars != " "] # remove white spaces from character vector

  # run loop with stepsizes
  # define the number of units (i.e. characters) used for analyses (note that k is
  # always either equal to or smaller than n)
  k = length(chars)
  for (i in 1:(k/stepsize))
  {
    # unigram entropy estimation
    # calculate unigram entropy for characters
    chars.df <- as.data.frame(table(chars[1:(i*stepsize)]))
    # print(chars.df)
    huni.chars <- entropy(chars.df$Freq, method = "ML", unit = "log2")
    # entropy rate estimation
    # note: the values chosen for max.length and every.word will crucially
    # impact processing time. max.length = NULL means all units in the file are
    # considered.
    hrate.chars <- get.estimate(text = chars[1:(i*stepsize)], every.word = 1,
                                max.length = NULL)
    # calculate type-token ratio (ttr)
    ttr.chars <- nrow(chars.df)/sum(chars.df$Freq)

    # calculate repetition measure according to Sproat (2014)
    # the overall number of repetitions is the sum of frequency counts minus 1.
    R <- sum(chars.df$Freq-1)
    # calculate the number of adjacent repetitions
    r = 0
    if (length(chars) > 1){
      for (j in 1:(length(chars)-1)){
        if (chars[j] == chars[j+1]){
          r = r + 1
        } else {
          r = r + 0
        }
      }
      # calculate the repetition measure
      rm.chars <- r/R
    } else {
      rm.chars <- "NA"
    }

    # append results to dataframe
    local.df <- data.frame(filename, subcorpus, code, huni.chars, hrate.chars,
                           ttr.chars, rm.chars, units = i*stepsize)
    stabilization.df <- rbind(stabilization.df, local.df)
  }
  # counter
  counter <- counter + 1
  # print(counter)
  })
}
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 23.62261 secs
```

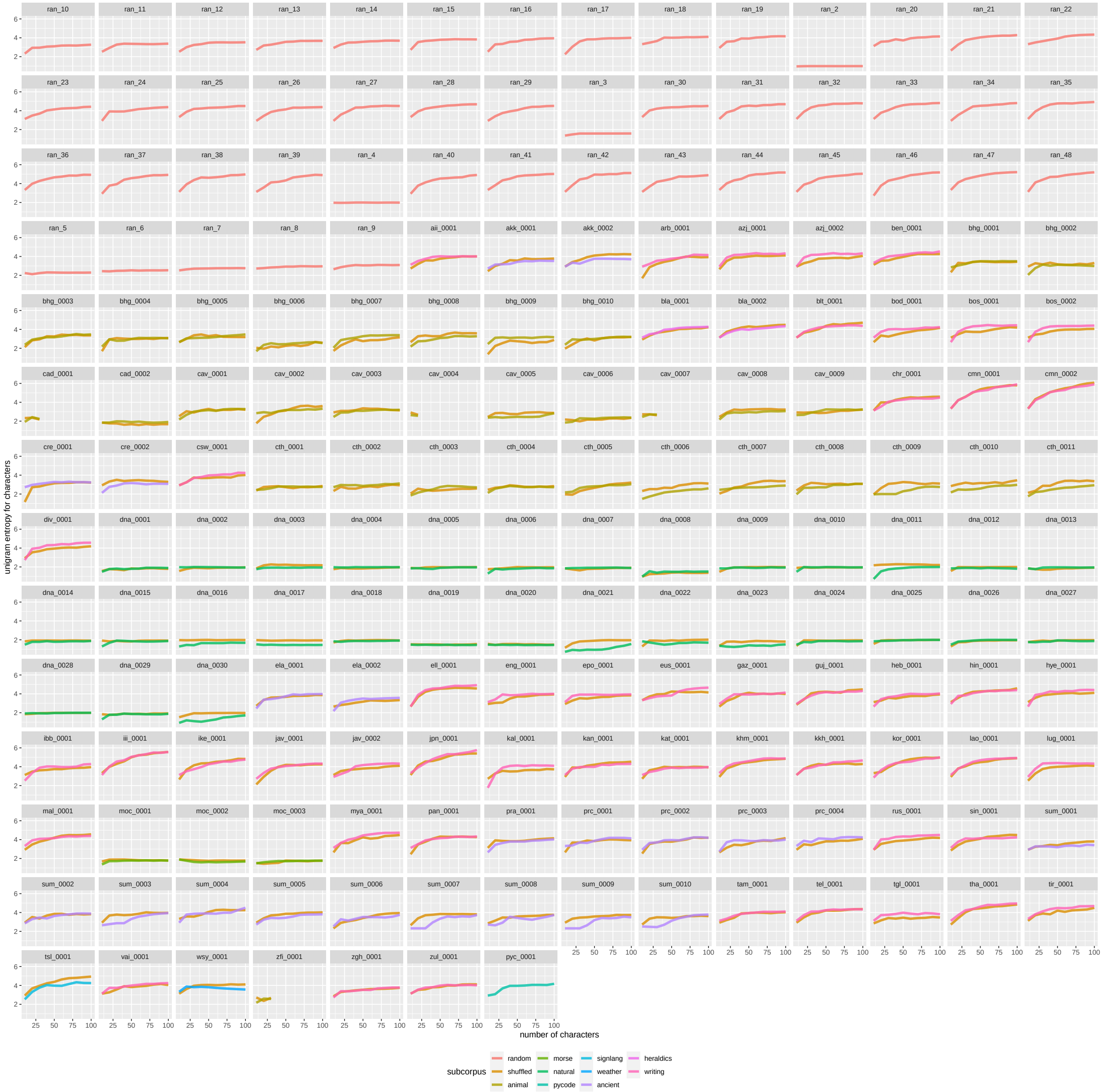
```
head(stabilization.df)
```

```
##        filename subcorpus   code huni.chars hrate.chars ttr.chars  rm.chars
## 1 random_ran_10    random ran_10   2.321928    1.562114 0.6000000 3.5000000
## 2 random_ran_10    random ran_10   2.941446    2.034298 0.4500000 1.2727273
## 3 random_ran_10    random ran_10   2.952584    2.106405 0.3000000 0.6666667
## 4 random_ran_10    random ran_10   3.064911    2.225446 0.2500000 0.4666667
## 5 random_ran_10    random ran_10   3.099987    2.293921 0.2000000 0.3500000
## 6 random_ran_10    random ran_10   3.175431    2.378248 0.1666667 0.2800000
##   units
## 1    10
## 2    20
## 3    30
## 4    40
## 5    50
## 6    60
```

Stabilization plots

Unigram entropy characters

```
huni.chars.plot <- ggplot(stabilization.df, aes(x = units, y = huni.chars,
                                                colour = subcorpus)) +
  geom_line(alpha = 0.8, size  = 1.5) +
  theme(legend.position = "bottom") +
  labs(x = "number of characters", y = "unigram entropy for characters") +
  facet_wrap(~code)
huni.chars.plot
```



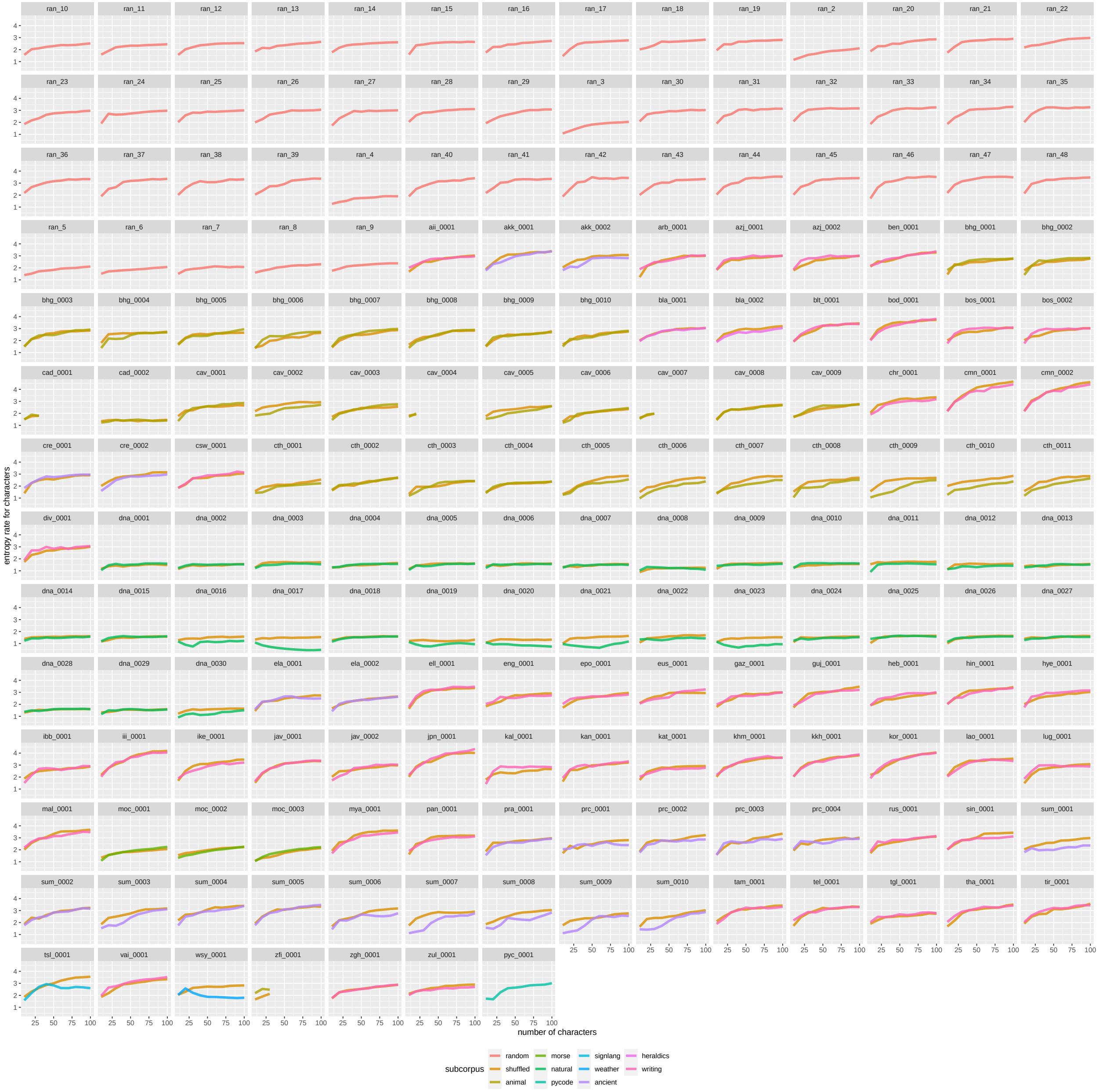
Safe figure to file

```r
ggsave("~/Github/NaLaFi/figures/stabilization_huni_chars.pdf", huni.chars.plot, dpi = 300,
       scale = 1, device = cairo_pdf)
```

```
## Saving 20 x 20 in image
```

Entropy rate characters

```r
hrate.chars.plot <- ggplot(stabilization.df, aes(x = units, y = hrate.chars,
                                                  colour = subcorpus)) +
  geom_line(alpha = 0.8, size  = 1.5) +
  theme(legend.position = "bottom") +
  labs(x = "number of characters", y = "entropy rate for characters") +
  facet_wrap(~code)
hrate.chars.plot
```



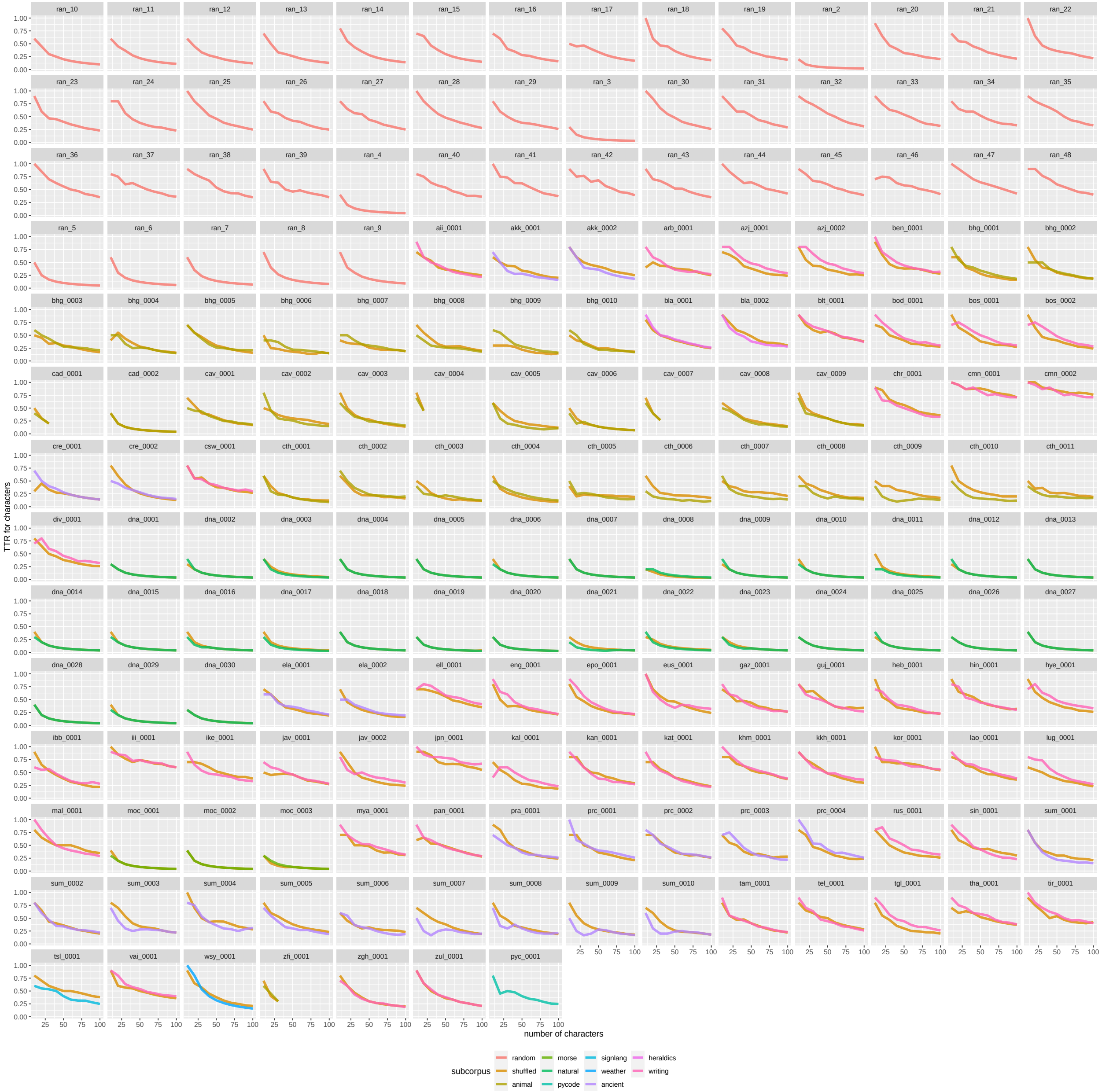
Safe figure to file

```
ggsave("~/Github/NaLaFi/figures/stabilization_hrate_chars.pdf", hrate.chars.plot, dpi = 300,
       scale = 1, device = cairo_pdf)
```

```
## Saving 20 x 20 in image
```

TTR characters

```
ttr.chars.plot <- ggplot(stabilization.df, aes(x = units, y = ttr.chars,
                                               colour = subcorpus)) +
  geom_line(alpha = 0.8, size  = 1.5) +
  theme(legend.position = "bottom") +
  labs(x = "number of characters", y = "TTR for characters") +
  facet_wrap(~code)
ttr.chars.plot
```



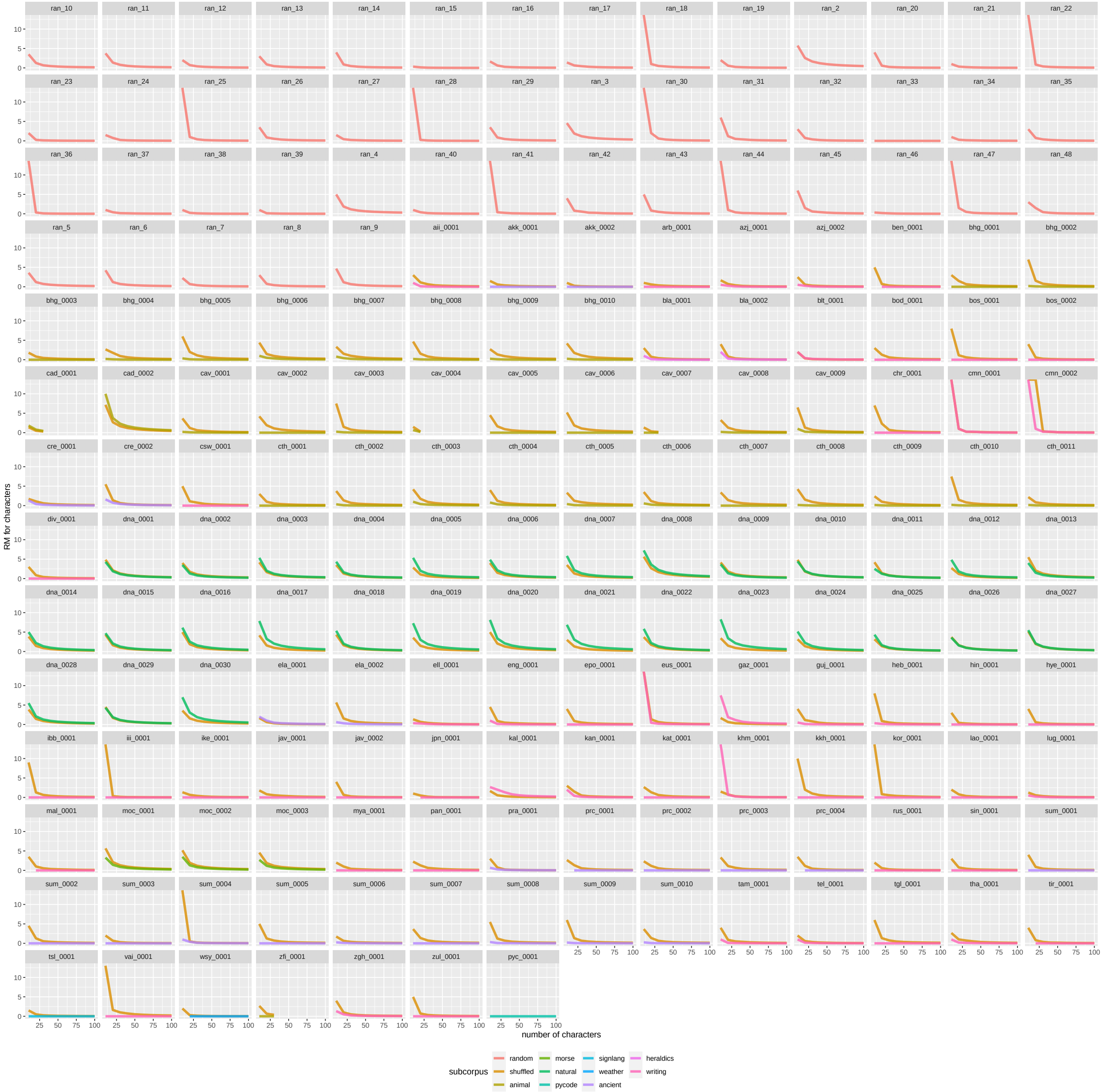
Safe figure to file

```
ggsave("~/Github/NaLaFi/figures/stabilization_ttr_chars.pdf", ttr.chars.plot, dpi = 300,
       scale = 1, device = cairo_pdf)
```

```
## Saving 20 x 20 in image
```

RM characters

```
rm.chars.plot <- ggplot(stabilization.df, aes(x = units, y = rm.chars,
                                                colour = subcorpus)) +
  geom_line(alpha = 0.8, size  = 1.5) +
  theme(legend.position = "bottom") +
  labs(x = "number of characters", y = "RM for characters") +
  facet_wrap(~code)
rm.chars.plot
```



Safe figure to file

```r
ggsave("~/Github/NaLaFi/figures/stabilization_rm_chars.pdf", rm.chars.plot, dpi = 300,
       scale = 1, device = cairo_pdf)
```

```
## Saving 20 x 20 in image
```

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