Appendix 4: Correlations Between Complexity Measures

25 March, 2021

Session Info

Give the session info (reduced).

```
## [1] "R version 3.6.3 (2020-02-29)"
```

```
## [1] "x86_64-pc-linux-gnu"
```

Load Libraries

If the libraries are not installed yet, you need to install them using, for example, the command: install.packages("ggplot2").

```
library(readr)
library(ggplot2)
library(gridExtra)
library(GGally)
library(ggrepel)
library(psych)
library(ggcorrplot)
```

Give the package versions.

```
## ggcorrplot      psych    ggrepel     GGally  gridExtra    ggplot2      readr
##    "0.1.3"   "2.0.12"    "0.9.0"    "2.0.0"      "2.3"    "3.3.3"    "1.4.0"
```

Load the Data

The participants' results are loaded as csv files directly from the github repository into separate data frames. We only use the name of the first author (lower case) to name the data frame.

```
#Track A (Parallel Bible Corpus, PBC)
gutierrez.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/maste
# remove the parentheses in column names
colnames(gutierrez.results) <- sub("\\(", "", colnames(gutierrez.results))
colnames(gutierrez.results) <- sub("\\)", "", colnames(gutierrez.results))
# replace "+" by "."
colnames(gutierrez.results) <- gsub("\\+", ".", colnames(gutierrez.results))
oh.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/master/PBCt

#TRACK B (Universal Dependencies, UD)
brunato.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/master/
coltekin.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/maste
semenuks.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/maste
```

```
sinnemaki.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/maste
sozinova.results <- read_csv("https://raw.githubusercontent.com/IWMLC/language-complexity-metrics/maste
```

Sanity check, look at the number of rows and columns of the data frames.

```
#Track A (should be 49 rows)
track.a.rows <- c(nrow(gutierrez.results), nrow(oh.results))
print(track.a.rows) # this corresponds to the number of languages
```

```
## [1] 49 49
```

```
track.a.cols <- c(ncol(gutierrez.results)-2, ncol(oh.results)-2)
print(track.a.cols) # this is the number of measures per team
```

```
## [1] 12  3
```

```
#Track B (should be 63 rows)
track.b.rows <- c(nrow(brunato.results), nrow(coltekin.results),
                  nrow(semenuks.results), nrow(sinnemaki.results),
                  nrow(sozinova.results))
print(track.b.rows) # this corresponds to the number of languages
```

```
## [1] 63 63 63 63 63
```

```
track.b.cols <- c(ncol(brunato.results)-2, ncol(coltekin.results)-2,
                  ncol(semenuks.results)-2, ncol(sinnemaki.results)-2,
                  ncol(sozinova.results)-2)
print(track.b.cols) # this is the number of measures per team
```

```
## [1] 11  6  2  6  2
```

Preprocessing

Put data into a single data frame.

```
track.a <- cbind(gutierrez.results, oh.results[, 3:ncol(oh.results)])
track.b <- cbind(brunato.results, coltekin.results[, 3:ncol(coltekin.results)],
                 semenuks.results[, 3:ncol(semenuks.results)],
                 sinnemaki.results[, 3:ncol(sinnemaki.results)],
                 sozinova.results[, 3:ncol(sozinova.results)])
```

Remove certain measures. To include all measures, this code can just be commented out. Note, however, that there are certain measures in Track A which are redundant in the sense that they only differ in whether the Bible texts are fully parallelized or not. In Track B, some measures given by the same team have strong positive correlations, e.g. the number of tokens in a sentence (BV_n_tokens) and the average number of tokens per clause (BV_avg_token_per_clause). We hence just keep one of the strongly correlated measures to not inflate the number of correlated data points. Also there are measures with many NAs in Track B, i.e. "SI_double_dl", "SI_head_dl", "SI_zero_dl", which are removed here.

```
# Remove measures in Track A
track.a <- track.a[ , -which(names(track.a) %in% c("GM_H1gram", "GM_H3gram", "GM_TTR",
                                                   "GM_TTR.H1", "GM_TTR.H3", "GM_TTR.H1.H3",
                                                   "GM_TTR.H1_fullyparallelised",
                                                   "GM_TTR.H3_fullyparallelised",
                                                   "GM_TTR.H1.H3_fullyparallelised"))]
# Remove measures in Track B
```

```
track.b <- track.b[ , -which(names(track.b) %in% c("BV_avg_max_depth",
                                                   "BV_avg_token_per_clause",
                                                   "SI_double_dl", "SI_head_dl",
                                                   "SI_zero_dl"))]
```

Invert the values (by substracting them from 1) for the measure "CR_inflection_accurracy" in Track B. Note that higher values in the original measure mean *lower* rather than higher complexity.

```
track.b$CR_inflection_accuracy <- 1-track.b$CR_inflection_accuracy
```

Center and scale all numerical columns to make them more comparable.

```
# keep meta-information columns again
track.a.scaled <- cbind(track.a[1:2], scale(track.a[3:ncol(track.a)]))
track.b.scaled <- cbind(track.b[1:2], scale(track.b[3:ncol(track.b)]))
```

Check the first 6 rows of the data.

```
#head(track.a.scaled)
#head(track.b.scaled)
```

Remove the first two columns of data frames (useful for plotting).

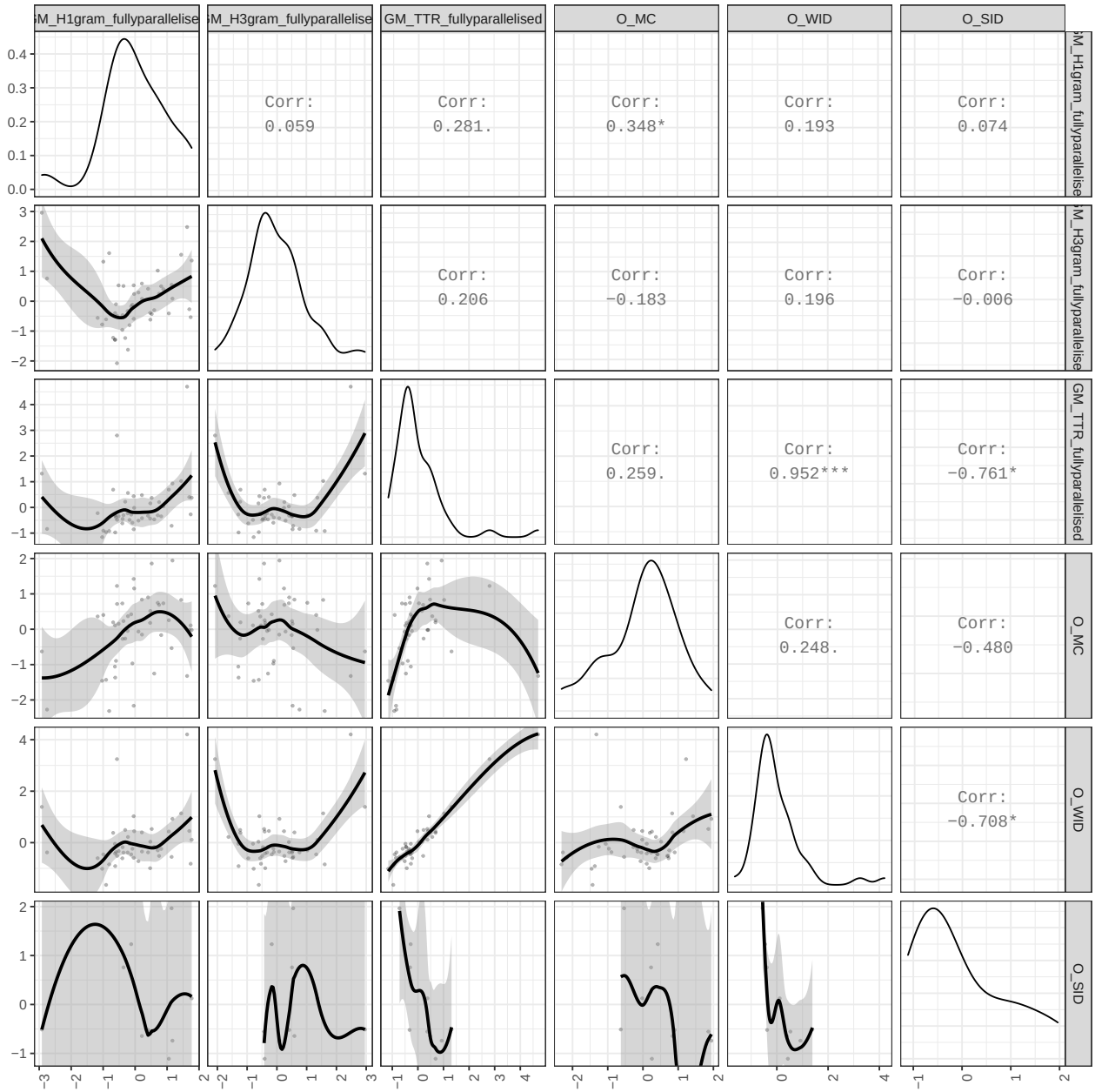
```
track.a.short <- track.a.scaled[, 3:ncol(track.a)]
track.b.short <- track.b.scaled[, 3:ncol(track.b)]
```

Scatterplots by Track

TRACK A

For visual reference, we here firstly give scatterplots between selected measures of the respective track. The Spearman correlation coefficient is reported instead of the Pearson correlation coefficient. This is because we are only interested whether there is a correlation between the rankings of complexities, regardless of whether this is a linear relationship. We therefore also use the local regression smoothers in the plots (loess) rather than linear models (lm). Note: warning messages are disabled here as there are datasets with NAs, and for each plot this throws a warning message using the ggpairs() plotting function. NAs are dealt with by removing the entire row containing an NA value.

```
track.a.scatterplot <- ggpairs(track.a.short,
                        lower = list(continuous = wrap("smooth_loess", alpha = 0.3,
                                                       lwd = 0.5, size = 2))) +
                        #upper = list(continuous = wrap('cor', method = "spearman"))) +
                        theme_bw() +
                        theme(axis.text.x = element_text(angle = 90, hjust = 1))
print(track.a.scatterplot)
```

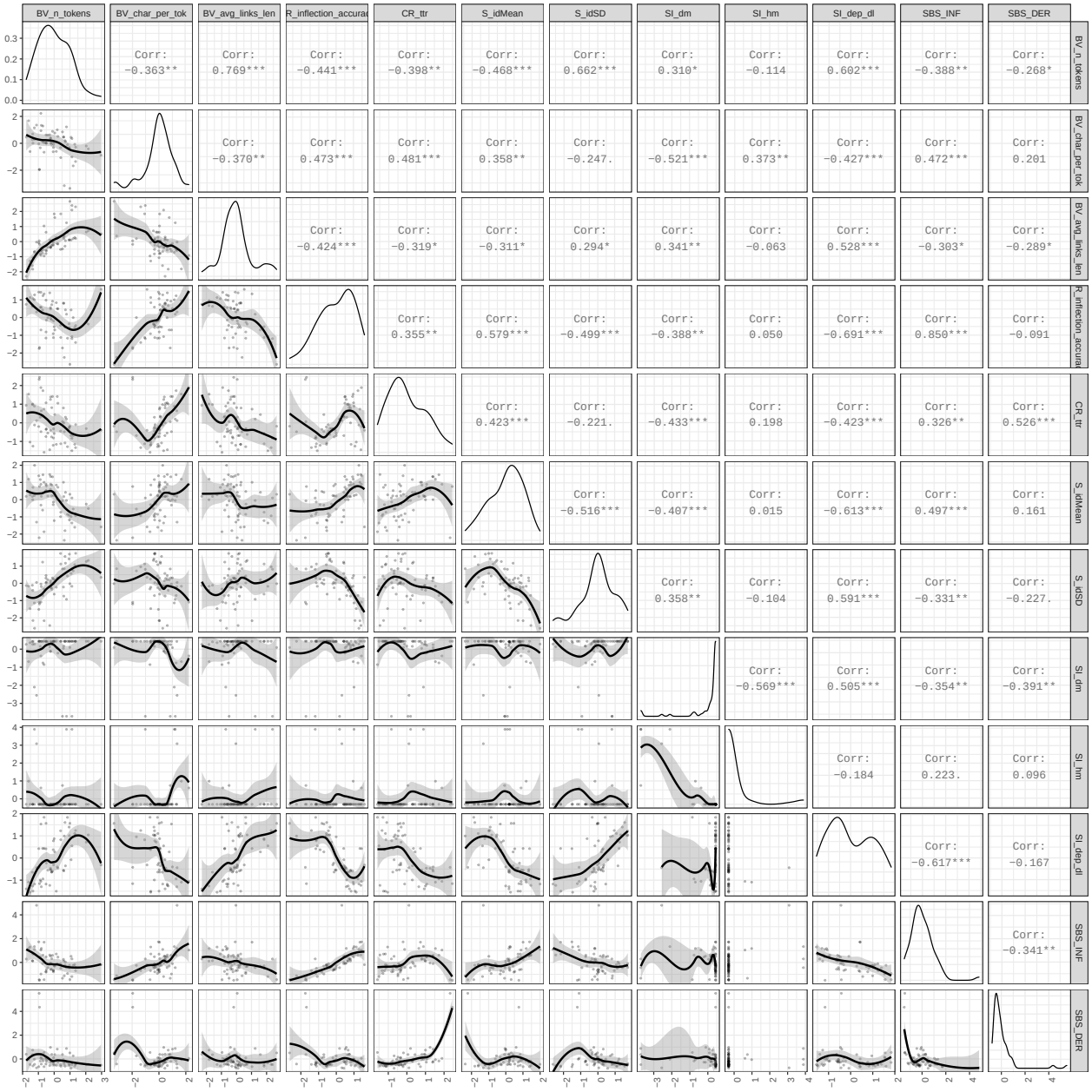


TRACK B

Same for the Track B data. Not all measures are included here (there would be 24). To include them all, the "columns" argument in the code below might be removed.

```
track.b.scatterplot <- ggpairs(track.b.short, progress = TRUE,
                        lower = list(continuous = wrap("smooth_loess", alpha = 0.3,
                                                       lwd = 0.5, size = 2)),
                        upper = list(continuous = wrap('cor', method = "spearman")),
                        columns = c("BV_n_tokens", "BV_char_per_tok", "BV_avg_links_len",
                                    "CR_inflection_accuracy", "CR_ttr",
                              "S_idMean", "S_idSD", "SI_dm", "SI_hm", "SI_dep_dl",
                              "SBS_INF", "SBS_DER")) +
```

```
                    theme_bw() +
                    theme(axis.text.x = element_text(angle = 90, hjust = 1))
print(track.b.scatterplot)
```



Significant Correlations after Correction for Multiple Testing

Not all of the correlations displayed above are going to be significant after correcting for multiple testing. We therefore use the corr.test() function here, since it enables us to choose a correction method, i.e. Holm-Bonferroni. The Bonferroni method would be more conservative, however, it is pointed out in MacDonald (2014, p. 254-260) that it is appropriate only when tests are independent of one another. Since we here run pairwise tests by complexity measures, our tests are not independent (the same measure is tested against others multiple times). We therefore apply the Holm-Bonferroni method (see also the descriptions in the

vignette invoked by the command "?p.adjust()"). Note that NAs are here deleted in pairs of columns, rather than across a whole row.

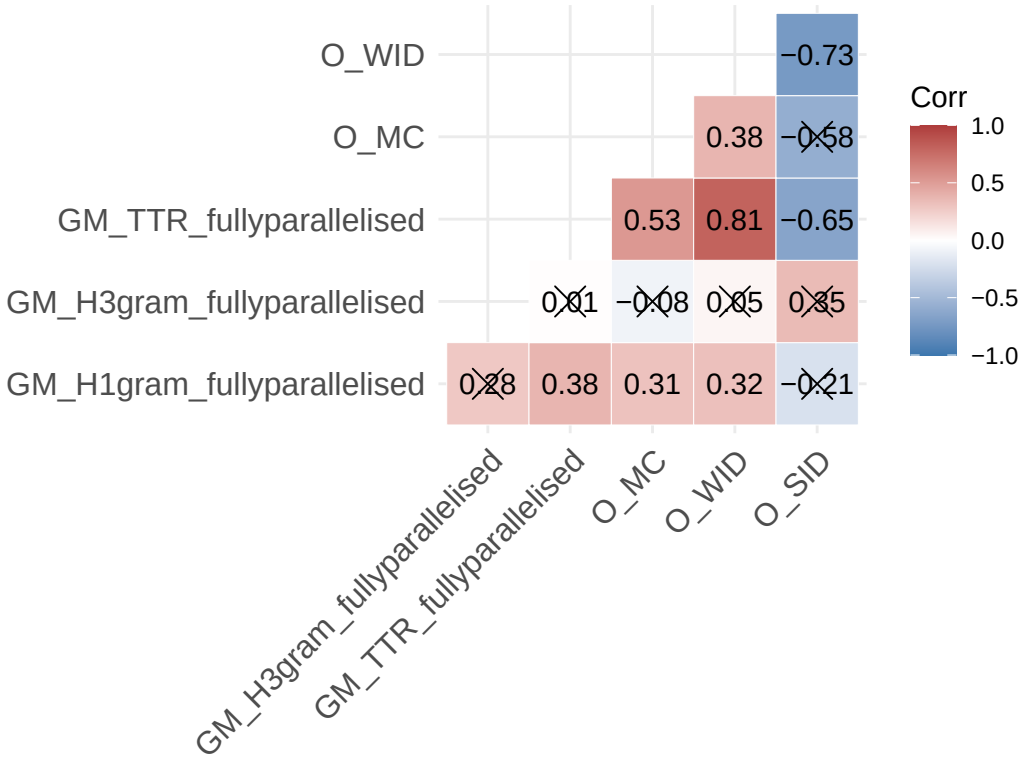
TRACK A

Calculate Spearman rank correlations with p-values adjusted by the Holm-Bonferroni method.

```
cor.results.a <- corr.test(track.a.short, method = "spearman",
                           use = "pairwise.complete.obs", adjust = "holm")
```

Give correlogram of selected measures.

```
correlogram.TrackA <- ggcorrplot(cor.results.a$r, p.mat = cor.results.a$p, type = "lower",
     outline.col = "white", colors = c("#3C77AE", "white", "#AE3C3C"),
     lab = T, insig = "pch")
correlogram.TrackA
```



Safe to file.

```
ggsave("Figures/Corrs/correlogram_TrackA.pdf", correlogram.TrackA,
       dpi = 300, scale = 1, width = 5.5, height = 5.5, device = cairo_pdf)
```

TRACK B

Reorder columns to have measures of the same domain together.

```
col_order <- c("BV_char_per_tok", "CR_inflection_accuracy", "CR_ttr",
               "CR_msp", "CR_mfe", "CR_cfe_form_feat", "CR_cfe_feat_form",
               "SI_dm", "SI_hm", "SBS_INF", "SBS_DER", "BV_n_tokens",
               "BV_verbal_head_per_sent", "BV_verbal_root_perc", "BV_avg_links_len",
```

```
              "BV_avg_subordinate_chain_len", "BV_subordinate_pre",
              "BV_subordinate_post", "BV_avg_verb_edges", "S_idMean", "S_idSD", "SI_dep_dl")
track.b.short.reorder <- track.b.short[, col_order]
track.b.short.reorder
```

```
##    BV_char_per_tok CR_inflection_accuracy       CR_ttr      CR_msp      CR_mfe
## 1     0.3468879742           -1.075215411 -1.499243910 -0.71957316 -0.34875945
## 2    -0.8885200699            1.590109935 -0.401409097 -0.12174250 -0.15892657
## 3     0.1042120149            0.419644762  0.354494808 -0.03051694  0.65660540
## 4    -0.4142382073           -0.369912715 -0.625341201 -0.60889296 -0.40117200
## 5     0.5082350317            0.490218708  1.052364889  0.14415716  1.28196175
## 6    -0.3376085815            0.589140585  0.298757104  0.24827254  1.46458262
## 7     0.4016707747            0.485597376  1.388236459  0.03066197  1.41331554
## 8    -0.6131451340            1.129441984 -1.035049699  2.38120309  1.12661272
## 9    -3.3479979861           -2.635788677 -0.183372462 -1.23134817 -1.72537388
## 10   -0.0869977777           -0.191174227 -0.322179912 -0.54490365  0.14126474
## 11    0.7679493989           -0.698914957  0.441436055 -0.80112508  0.19824449
## 12    0.4470033087            0.429155126 -0.687586350  0.28228248  0.49322349
## 13    1.2829319931            1.228518845  1.724286400 -0.02361676  0.22836053
## 14    0.0287309618           -1.154975665 -0.740286408 -0.64067327 -0.50420210
## 15   -0.3089793016           -1.131404051 -0.814229199 -0.67037369 -0.61730816
## 16   -0.6311199512           -0.910425456 -1.206954145 -0.51301542 -0.48413367
## 17   -0.1091412557           -1.164288777 -1.282499870 -0.59805434 -0.61115800
## 18    1.3330436636            0.570317109  0.644365188  0.85495590  0.46432644
## 19   -1.0173103201           -1.187987196 -0.832391749  0.15490996 -1.39955023
## 20    1.6750711432            0.955944764  1.538195317  0.46984553  0.59523937
## 21    2.2247350990            1.421234046  1.889085444  0.36740192  0.71399634
## 22   -0.3191925126           -0.562571559 -0.223068264 -0.63233853 -0.50199025
## 23    0.1953329543           -0.539014035 -1.080931489 -0.39694195 -0.52860670
## 24    0.0085336799            1.063785249 -1.254353792  1.45542533  0.90702806
## 25    0.0455391143            1.336331151  0.733078556  1.03541814  0.35225902
## 26   -0.0392760600            1.435083955 -0.201734169  1.45918955  1.11062097
## 27   -1.6779295203           -0.003263521 -0.685107024 -0.23670868 -0.11609345
## 28   -1.0690402341           -1.121879598 -1.125095783 -0.77324029 -0.27164884
## 29    0.4644057078            0.810161448  0.795275849  0.16061670  0.46086095
## 30    1.2183256071            0.189733469  0.200537801 -0.26961309  0.75160009
## 31    0.8446322115           -1.714382329  0.064382731 -0.91751642 -1.66143363
## 32   -0.3279718363           -0.440472573 -0.441943104 -0.44321244 -0.37960412
## 33   -0.0794135226           -0.558091121 -1.154432459 -0.27074769 -0.39933209
## 34    0.0574696959           -0.515625584 -0.573911018 -0.45575742 -0.14327985
## 35   -3.1525011433           -2.245567869 -0.479593184 -1.13528621 -2.57024352
## 36   -1.9525258299           -1.748521014  2.328290138 -1.23520817 -2.57024352
## 37   -1.9531225131           -1.682765652  2.439474542 -1.24991044 -2.57024352
## 38    0.6277629789            1.393055190 -1.445687279  1.83123297  0.75965854
## 39    0.2812856979            1.280100242 -0.118648954  1.36210851  1.05308377
## 40    0.6161738762            0.979375483  1.070309428  0.34774918  1.11238191
## 41    0.0207526594           -0.811700832 -0.354403899 -0.76923253 -0.72820004
## 42    0.9333625096           -0.698957226 -0.475484338 -0.78958033 -0.94245652
## 43   -0.0009643613           -0.144369146 -0.327732172 -0.50121639  0.13493796
## 44    0.0245909669           -0.050237675 -0.403257111 -0.53668417  0.06902399
## 45    0.4596034153            0.744293370  0.933956650  0.15781598  1.43633257
## 46    0.6962893779            0.673818050  1.005483924  0.02092016  1.31067458
## 47   -0.3721524480           -0.351004703 -0.303405870 -0.53380762 -0.55593018
## 48   -0.6034754673            0.838269858 -1.156938372  1.62681968  0.82377947
```

```
## 49    0.1194089858            0.434058247  0.301777348  0.10423478  0.46847471
## 50    1.0047235545            0.504547657  1.410063615 -0.19664095  0.21355456
## 51    0.8870587935            0.922876875  1.411235845  0.17691145  0.68340020
## 52    0.2675726824            0.631324334  0.977480930  0.10360021  1.21488325
## 53    0.0694179250            0.725441715  0.811523676  0.07612858  0.68550663
## 54   -0.1992476846           -0.266439953 -0.310408794 -0.52025763 -0.34005391
## 55   -0.2899258253           -0.449884311 -0.100324951 -0.55826244 -0.22711421
## 56    0.4634435246            0.805709189 -0.004067259  0.45479417 -0.14017450
## 57   -0.0812082554           -0.120882069 -0.755740986 -0.41111057  0.14929230
## 58    0.6058761054           -0.007870764 -0.549516074 -0.23987404  0.12284433
## 59    1.1465378083            1.002834381  0.891415110  2.12243396  0.71414432
## 60    1.3545187241            0.316566504  0.115149698  4.45210554 -0.22224725
## 61    0.3571339054            0.561032176  1.251170675 -0.13816985  0.83725227
## 62   -1.3549823208           -1.404133117 -1.332374976 -0.93284071 -0.45960421
## 63   -0.6622357055                     NA -1.583122858 -1.23320090 -2.57024352
##    CR_cfe_form_feat CR_cfe_feat_form       SI_dm      SI_hm     SBS_INF
## 1      -0.764887514    -0.2048453481  0.43235739 -0.3052322 -1.18826015
## 2       1.169809712    -0.3078945224  0.43235739 -0.3052322 -0.18969895
## 3      -1.135856695    -0.0883839195  0.43235739 -0.3052322 -0.10384382
## 4      -0.196687731    -0.1213086897  0.43235739 -0.3052322 -0.33499225
## 5       0.077242163    -0.3131366908 -0.21833387 -0.3052322  0.44959155
## 6       0.313024463    -0.1103563021  0.15621893 -0.3052322  0.47865021
## 7      -0.020564926    -0.2687758805 -0.12692327 -0.3052322  0.52884244
## 8       0.337432457     0.3639715322  0.35487202 -0.3052322  1.29625521
## 9       0.331990498    -0.6703426248  0.09221557 -0.3052322 -1.44054215
## 10     -0.089532326    -0.1413100586  0.42800175 -0.3052322 -0.67577107
## 11      2.206341249    -0.2613338735  0.43235739 -0.3052322 -0.30857529
## 12      0.967635549     0.0008109525  0.43235739 -0.3052322  0.51299226
## 13     -0.488061528    -0.2241758442 -0.76128863 -0.3052322  0.61073503
## 14      0.481442316     0.0526657754  0.43235739 -0.3052322 -0.76030535
## 15      0.002967505    -0.1456681312  0.43235739 -0.3052322 -0.79200571
## 16     -0.061177565    -0.0731514677  0.43235739 -0.3052322 -0.69426295
## 17     -0.154364377    -0.1126393497  0.43235739 -0.3052322 -0.92805307
## 18      0.494387174     0.4550594094  0.42570869 -0.3052322  1.18002058
## 19     -0.799020989     0.7589543626 -3.71421692  3.8885877  0.29505232
## 20     -0.478794033    -0.1827949571 -0.69091003  0.9851739  1.16945379
## 21     -1.119490408    -0.3206931102 -0.37484518  0.6096684  1.04265237
## 22     -0.252440870     0.2290637033  0.43235739 -0.3052322 -0.25309966
## 23     -0.851581691     0.1687163595  0.43235739 -0.3052322 -0.23857033
## 24      0.677985938    -0.4893525990  0.43235739 -0.3052322  0.70055270
## 25      0.680720387     0.5322409787  0.43235739 -0.3052322  1.70175560
## 26      0.117733570    -0.0012744737  0.43235739 -0.3052322  1.72685171
## 27     -0.094785703    -0.2048817521  0.43235739 -0.3052322 -0.24649542
## 28      3.445289412    -0.5855786334  0.43235739 -0.3052322 -0.49481488
## 29      0.175210894    -0.2826561856  0.04640087 -0.3052322  0.26203111
## 30     -0.318310125    -0.2909146811 -3.71421692  3.8885877 -0.44066010
## 31     -0.852336022    -0.2802535251 -3.71421692  0.7471311 -1.33091175
## 32     -0.774909339     0.0499926855  0.43235739 -0.3052322 -0.11044806
## 33     -0.948041760    -0.0528276639  0.43235739 -0.3052322 -0.18045302
## 34     -0.579564565     0.3497479898  0.43235739 -0.3052322 -0.28744172
## 35     -0.449078782     0.0566910119  0.43235739 -0.3052322 -1.12618029
## 36     -1.269831259    -0.6458375676  0.43235739 -0.3052322 -1.45507148
## 37     -1.023097689    -0.6367573830  0.43235739 -0.3052322 -1.49998032
## 38      1.152433160    -0.5861298931  0.12876498 -0.3052322  0.62526435
```

```
## 39      0.650466328    -0.4397236174  0.28107919 -0.3052322  1.09416545
## 40     -0.351945203    -0.1978973947  0.20414934 -0.3052322  0.46544173
## 41     -0.551896784     0.0809100381  0.43235739 -0.3052322 -0.81710182
## 42     -0.700473039     0.0545639812  0.43235739 -0.3052322 -0.97296191
## 43     -0.037025505    -0.1612282184  0.42137706 -0.3052322 -0.61237036
## 44      0.237281555    -0.1453092923  0.42445790 -0.3052322 -0.58727425
## 45     -0.233717298     0.0386554562  0.26534049 -0.3052322  0.28580638
## 46      0.076231898    -0.1129097790  0.26068083 -0.3052322  0.11409612
## 47      0.682969910    -0.2151996709  0.43235739 -0.3052322 -0.45651028
## 48      1.268128669     0.9767643811  0.43235739 -0.3052322  0.71111948
## 49     -0.915767172    -0.2077316611  0.43235739 -0.3052322  0.03748693
## 50     -0.526828750    -0.2964896857  0.08305890 -0.3052322 -0.10516467
## 51     -0.358815003    -0.1018585806  0.24718387 -0.3052322  0.59884739
## 52      0.039040689    -0.1343517041  0.40661826 -0.3052322  0.14711732
## 53     -0.148801186    -0.1481383990  0.33220790 -0.3052322  0.31486504
## 54     -0.657839871    -0.2825001687  0.43235739 -0.3052322 -0.35084243
## 55     -0.836993470    -0.2384565928  0.43235739 -0.3052322 -0.41820568
## 56     -0.231885351    -0.3025275404  0.16967473 -0.3052322  0.12202121
## 57      0.090281312    -0.1598968741  0.40167450 -0.3052322 -0.57406576
## 58     -0.120554186    -0.1376176582  0.40858506 -0.3052322 -0.43405586
## 59     -0.967923769    -0.0027046288 -2.10188196  3.8885877  1.33059727
## 60     -0.833827974     7.3787989929 -2.55617967  3.0852644  4.80310705
## 61      0.185448243    -0.2165674192 -0.11179439 -0.3052322 -0.05233075
## 62      4.632323442    -0.4432235969  0.43235739 -0.3052322 -0.69294210
## 63     -0.297108038              NA -0.85654226 -0.3052322 -1.45110894
##         SBS_DER  BV_n_tokens BV_verbal_head_per_sent BV_verbal_root_perc
## 1  -0.340651828  1.125665621             0.353387896          0.12660563
## 2  -0.520427298  2.965365537             1.446980421         -4.44118514
## 3   0.001216280 -0.729022638            -1.087663662          0.51960290
## 4  -0.440854549  2.164380400             0.690108734          0.66341250
## 5  -0.269920495  0.240252566            -0.878909699         -0.51791166
## 6  -0.178559191 -0.879109170            -0.217866612          0.40855499
## 7  -0.281709051 -0.224602382            -1.090812118         -0.66389228
## 8  -0.467378799 -1.527888500             0.562165647          1.88413606
## 9   0.511071302  0.994988485             3.499325867          0.59425897
## 10  0.098471862 -0.040232802            -0.250352841         -0.31643320
## 11  0.366661498  0.041547088            -1.641812677         -0.14087073
## 12 -0.782722656  1.074879333             0.938671237          0.55616006
## 13 -0.172664913 -0.710455831            -1.066160029         -0.52104660
## 14 -0.426118855 -0.516781912            -0.736647153         -1.65031090
## 15 -0.272867634 -0.048205080            -0.233717954         -0.48518350
## 16 -0.440854549 -0.061229785             0.204695887          0.26857027
## 17 -0.296444745  0.846672740             0.416193791          0.08858123
## 18 -0.432013132 -0.812425935             0.133029215          0.97063375
## 19 -0.476220215  1.129002310            -0.045894506         -0.83554387
## 20 -0.057726498 -1.618317206            -0.537698844          0.03513697
## 21  0.051317640 -0.838593033            -0.474298881         -0.51959255
## 22 -0.526321576  0.967851108            -0.213504197         -0.12288819
## 23 -0.608841464  0.688749370            -0.331706512         -1.01474631
## 24 -0.585264353 -1.339839376             0.621690835          1.56065735
## 25 -0.650101408 -0.637721884             1.351012075          1.78702699
## 26 -0.650101408 -0.970189231             0.542556096          0.86566809
## 27  0.260564499  1.204840185             0.488226675          0.46453080
## 28 -0.608841464  0.421355117            -0.074359733          1.08756520
```

```
## 29 -0.054779359  0.589679371           -0.257138382        -0.43848528
## 30  0.879463659  0.781299532           -0.063476781         0.57025254
## 31  0.372555775  0.530125181            0.247792246         0.02592547
## 32 -0.499797326  0.410334910           -0.586849459        -0.47628420
## 33 -0.602947186  1.304343706            0.412192322         0.27497087
## 34 -0.493903048  0.002260029           -0.807855243        -1.81579912
## 35 -0.426118855  0.642767251            0.128222281        -0.61890903
## 36  5.521207357 -0.947260761            1.766141750         0.06158669
## 37  4.354140370 -0.927320787            0.215422267        -0.40560972
## 38 -0.558740103 -0.278602094            0.885464286        -0.35098839
## 39 -0.449695965 -1.239205618            0.306485573         0.60676792
## 40 -0.007625137 -0.512438725            0.446747349         0.37501524
## 41  0.089630445 -0.512082607           -1.090192651        -0.06132369
## 42  0.151520361 -0.835071046           -2.463152247        -2.74465913
## 43 -0.175612052 -0.493240574           -0.834910144        -0.53930933
## 44 -0.157929219 -0.222885493           -0.812592535        -1.16496924
## 45  0.396132886 -1.768977534           -1.695900766         1.58714535
## 46  0.740948132 -1.353980558           -1.527716286         1.02253499
## 47 -0.449695965  0.938756911            0.003953795        -0.22355107
## 48 -0.526321576  0.137254382            2.121553113         0.83064412
## 49 -0.196242024  0.715278575            0.771562412         0.92173834
## 50  0.826415159  0.199956644           -0.828652959         0.02412977
## 51 -0.210977718 -0.104944867           -0.134237147        -0.31239507
## 52  0.357820081 -1.379529346           -1.541560334         0.21650000
## 53  0.381397192 -0.152146712           -0.522981059         0.24132013
## 54 -0.490955909  2.032910728            0.992844442         0.38200978
## 55 -0.396647466  1.364149416            1.038524718        -0.08825126
## 56 -0.264026218  0.610540991           -0.321149167        -0.22503487
## 57 -0.160876358 -0.167793907            0.290818658         0.25499127
## 58 -0.040043664 -0.397652326           -0.855761468        -0.11998446
## 59 -0.022360831 -1.251658425            0.097054719        -0.37437105
## 60 -0.747356990 -1.113759317            0.599398214         0.74711776
## 61  0.973772102 -0.232704737           -0.535035944        -0.21434148
## 62 -0.461484521  1.358210463            0.520898904         1.17679550
## 63  0.546436968 -0.637547751            1.667446564         0.20332383
##    BV_avg_links_len BV_avg_subordinate_chain_len BV_subordinate_pre
## 1        2.09243539                   0.441897979        -0.1403422
## 2        0.51986231                   1.782339005        -1.0079731
## 3       -0.77265563                  -1.185880180        -0.3936576
## 4        0.71161424                   0.573792436        -0.3636057
## 5       -0.15981615                  -0.917835441        -0.6147766
## 6       -0.81634676                  -0.804364366        -0.4387427
## 7       -0.35204100                  -0.958421235        -0.4922567
## 8       -1.28653298                   0.290895687         0.4174365
## 9        2.66874454                   2.592433405         2.5632383
## 10       0.06067893                  -0.208907744        -0.2789602
## 11       1.59654426                  -1.964047576        -0.6359679
## 12       0.10974114                   0.862882381        -0.3830921
## 13      -0.53072829                  -0.755350829        -0.6144441
## 14      -0.79472698                  -0.300891251        -0.6093822
## 15      -0.22855165                   0.072698091        -0.3766580
## 16      -0.05379588                   0.182746830        -0.3894242
## 17       0.62004845                   0.645457513        -0.1956373
## 18      -0.59804270                  -0.018861742         0.7051664
```

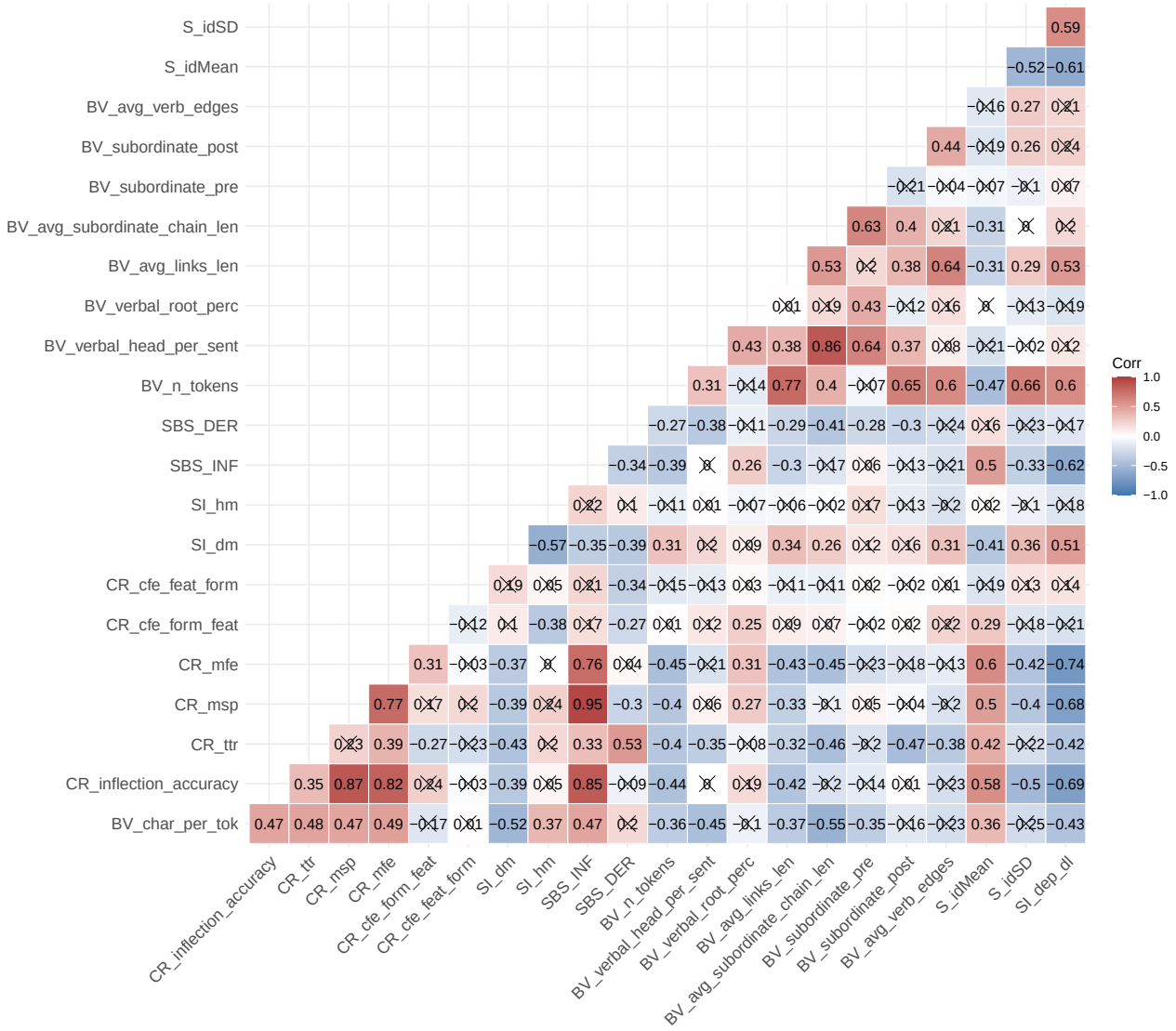
```
## 19       2.39398238                0.708605364         -0.1317135
## 20      -1.81395649               -0.514708733          0.4273918
## 21      -0.91237135               -0.444523218         -0.5117473
## 22       0.11650902               -0.109417680         -0.4031365
## 23      -0.30144782               -0.177693723         -0.4983150
## 24      -1.07030990                0.437495724          0.3635254
## 25       1.03803804                0.540188198          0.5317450
## 26      -0.07737705                0.218648316          0.4447362
## 27       0.19835853                0.203327717         -0.2682047
## 28       1.83925721                0.223861101          0.8966193
## 29       0.30416735                0.289807642         -0.5387394
## 30       1.24877486               -0.380575778         -0.5113796
## 31      -0.48640590                0.020520480         -0.4378801
## 32      -0.51822008               -0.635523227         -0.4712879
## 33       0.27494646                0.594026572         -0.2179310
## 34       1.83499028                0.115089191         -0.7650372
## 35       0.37394837                1.186970417          3.2033683
## 36       0.05474346                1.160826731          2.6943058
## 37      -0.18099471                2.213840564          3.5499824
## 38       0.10945269                1.144847811          0.1170908
## 39      -0.65312795                0.000612195          0.3678733
## 40      -0.56264551               -0.270754870         -0.2770327
## 41       0.62308924               -0.705802058         -0.3038530
## 42      -0.62787816               -2.134895399         -0.7216778
## 43      -0.61402060               -0.398217984         -0.6458221
## 44      -0.36730477               -0.216327508         -0.5778297
## 45      -2.30160889               -2.653774941         -0.9086415
## 46      -1.82948914               -2.260695005         -0.8664700
## 47       0.10609583                0.132276638         -0.2451817
## 48       0.28263626                0.932610599         -0.1161442
## 49       0.06180112                0.838574927         -0.2187304
## 50      -0.12770476               -1.100767221         -0.6128861
## 51      -0.21811283               -0.309327687         -0.3491838
## 52      -1.59484174               -1.978133468         -0.5020043
## 53       0.01263833               -0.156348461         -0.5678342
## 54       0.63431110                0.911513418         -0.1852634
## 55       0.20093352               -0.005681645         -0.5470819
## 56       0.20088672                0.327332161         -0.5591730
## 57      -0.26656092                0.120554119         -0.5303646
## 58      -0.38210743               -0.487522316         -0.4594442
## 59      -0.70750885               -0.157249647          1.7516882
## 60      -0.03603216                0.879762063          2.7491053
## 61      -0.39947480               -0.743999955         -0.5969519
## 62       2.30775598                0.528648699          0.5787058
## 63      -0.95424618                1.781416916          0.1198844
##    BV_subordinate_post BV_avg_verb_edges      S_idMean      S_idSD   SI_dep_dl
## 1          0.892903421        1.98576759 -1.3568436916 -0.17345277  0.54954775
## 2          2.268443901        0.06410904 -1.5863483091  0.34629485 -0.31408118
## 3         -0.640792593       -0.14441090 -0.6077700045  1.22953223  0.42982106
## 4          1.006118797        1.71471879 -0.5194843659  1.32633290  1.04304203
## 5         -0.058963440       -0.31049697  0.7190170356  0.28422360 -0.45606952
## 6         -0.242179111       -0.27239435  1.1682772849 -0.71129745 -0.99229402
## 7         -0.352163553       -0.53585624  1.3472284641  0.14267310 -0.52088314
## 8         -0.175616423       -0.35917691  0.2784064948 -1.46598953 -1.54737683
```

```
## 9        -1.053273978       -0.67651060  0.3128954417 -0.02251246  1.56733410
## 10        0.243564744        0.54189030  0.2383497613 -0.02998084 -0.30945538
## 11       -1.059405371        0.96446954  0.4258445331  0.20172379  0.27444167
## 12        1.085175363       -0.07450237 -0.9842163328  0.57383563 -0.08022851
## 13       -0.008822739       -0.10927346  1.4467815647 -0.70233882 -0.94266771
## 14        0.198659023       -0.94356338  0.5287128298  0.30293357  0.41663736
## 15        0.376761927       -0.49421998 -0.1375133604  0.03660201  0.77579295
## 16        0.565782061        0.43014147 -0.2915944659  0.24956868  0.37340425
## 17        0.870217568        0.59868001 -0.8518121959  0.42220773  1.83343673
## 18       -0.730125518        0.42021724  0.1905245931 -0.19175788 -1.29726561
## 19        0.687387481        0.36649836  0.0978210423  0.81126158          NA
## 20       -0.837048934       -1.72891257  1.1910209159 -0.16844564 -1.09057294
## 21        0.303094148       -0.76850543  0.8935888898 -1.09002351 -1.05790122
## 22        0.505706339        0.49859383 -1.1818062105  1.69051481  1.00713066
## 23        0.405691652       -0.67594238 -1.8052066154  1.55958579  1.60317753
## 24       -0.089357897       -0.43203894  0.7135259290 -2.15501867 -1.49550209
## 25        0.104976830        0.31538739  1.0097459674 -2.16679240  0.42997213
## 26       -0.264761885       -0.40859350  1.3873605540 -1.51076355  0.42313012
## 27        0.745045712       -0.12653367 -0.5462488880  0.72109092  0.95685248
## 28       -0.712548715        2.22362613 -0.7309108555  1.12104088  0.82403733
## 29        0.833941692        0.66741851  1.0123749307 -0.21726119 -0.55002668
## 30        0.255667756        1.74708369 -0.0492317161 -0.92506349          NA
## 31        0.742616608       -0.17407347 -0.0691837708  0.83209640          NA
## 32       -0.047858560       -0.15366785 -0.9914647135  1.73102084  1.22801679
## 33        0.876327706        0.38596275 -1.6071692134  1.07778021  1.10675061
## 34        0.918400406        0.37453836 -1.2206425479 -0.62723879  0.87266689
## 35       -2.859064826        1.26553544 -2.3639047198  0.84784566  0.99333786
## 36       -2.064567700       -1.89805740 -2.2157937148 -1.09649016 -0.42122706
## 37       -2.161603171       -1.19747814  0.7627639236 -1.19647194 -1.15858120
## 38        0.863319709       -0.43182368  1.4636559555 -2.44275749 -1.23585627
## 39       -0.355551459       -0.69952939  1.9955413747 -2.60542383 -1.17562861
## 40        0.158756804       -0.68328730  0.8988594163  0.45442667 -1.17783046
## 41       -0.230849183        0.47735551  0.4736339218  0.02566832  1.56519173
## 42       -1.240454139       -1.93597062 -0.3176669172 -0.04292098  1.53450328
## 43        0.374650720        0.40261008  0.8014307033  0.06322727 -0.23008586
## 44        0.488499727        0.10653390  0.9035584070  0.01106543 -0.53388290
## 45       -1.538470105       -0.70245434  0.1912664201 -0.13120974 -1.39431889
## 46       -1.198171354       -0.58533289 -0.0051921895 -0.40705471 -1.26334161
## 47        0.438445004       -0.29308740 -0.9066517673  1.41304441  1.39567382
## 48        0.877569059        0.46381825  0.8499804345 -0.22651216 -1.50083797
## 49        1.189221612        1.00774488  0.4244604440  1.16594217 -0.35357294
## 50       -0.197277469       -0.63913285  0.0001212286  0.31180670 -0.40158500
## 51        0.305267879       -0.64528217  1.3285377246  0.31326468 -0.59190610
## 52       -1.306933467       -0.92320275  0.1849675831 -0.11202374 -0.64096974
## 53        0.586839090        1.29127045  0.8459444547 -0.35113227 -0.63429037
## 54        1.052644555        0.95662693 -0.2577059605  1.26143756  0.71531090
## 55        0.773302165       -0.44003824 -0.8169750185  1.73749756  1.04086136
## 56        0.891570308        0.85276706  0.1804266128 -0.74106489 -0.58395604
## 57        0.721301507        0.90222919  0.2207180299 -0.26047194 -0.63558605
## 58        0.082331572        0.64071336  0.2081927212  0.20467670 -0.55229033
## 59       -2.039133018       -2.10605786  0.2365428125 -1.13803974          NA
## 60       -2.491742335       -1.23105526 -1.3079742606  0.05536483 -0.43635674
## 61        0.044769818       -0.53392855  0.6035523178 -0.23718340 -0.46225987
## 62       -0.063305404        2.97436051 -0.9394453516  0.95045488  1.24166140
```

```
## 63          1.285069687       -1.30627674 -1.8668735615 -0.32934837  1.83695605
```

Same as above for Track A.

```
cor.results.b <- corr.test(track.b.short.reorder, method = "spearman",
                           use = "pairwise.complete.obs", adjust = "holm")
# produce correlogram
correlogram.TrackB <- ggcorrplot(cor.results.b$r, p.mat = cor.results.b$p, type = "lower",
     outline.col = "white", colors = c("#3C77AE", "white", "#AE3C3C"),
     lab = T, insig = "pch")
correlogram.TrackB
```



Safe to file.

```
ggsave("Figures/Corrs/correlogram_TrackB.pdf", correlogram.TrackB,
       dpi = 300, scale = 1, width = 12, height = 12, device = cairo_pdf)
```

Detailed Scatterplots

We here plot the highest *positive* and *negative* correlations (in terms of Spearman coefficients) which are still significant after the Holm-Bonferroni correction *and* which are found between measures proposed by *different participants* (there are many measures by the same participants that highly correlate). These are hand-picked from the correlograms above.

TRACK A (Highest Positive Correlation)

```
track.a.positive.detailed <- ggplot(track.a, aes(x = GM_TTR_fullyparallelised, y = O_WID)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = loess, alpha = 0.3) +
  geom_label_repel(data = track.a[track.a$language == "Fijian"
                                  | track.a$language == "Sango"
                                  | track.a$language == "Vietnamese"
                                  | track.a$language == "English"
                                  | track.a$language == "Georgian"
                                  | track.a$language == "Russian"
                                  | track.a$language == "Swahili"
                                  | track.a$language == "Basque"
                                  | track.a$language == "Finnish"
                                  | track.a$language == "Turkish"
                                  | track.a$language == "Korean"
                                  | track.a$language == "Kalaallisut"
                                  | track.a$language == "Burmese", ],
                   min.segment.length = 0,
                   #nudge_x = 0.1,
                   aes(label = language),
                       size = 3) +
  ggtitle("a) High Positive Correlation Track A (r = 0.81)") +
  xlab("Type-Token-Ratio (GM_TTR_fullyparallelized)") +
  ylab("Word Information Density (O_WID)") +
  theme(legend.position = "none")
# track.a.positive.detailed
```

Some comments: This plot shows that the Type-Token Ratio (TTR) and the Word Information Density (WID) are highly correlated across the languages of the Parallel Bible Corpus sample. Burmese (mya) is an outlier here with very high TTR and WID. This is an artifact of the writing system, since it does not delimit orthographic words by white spaces, but rather phrases. For Kalaallisut, on the other hand, the result makes sense (if we accept the latinized writing proposed for this language). Some of the low TTR languages include Sango (sag), Fijian (fij), Thai (tha), and Yoruba (yor).

TRACK B (Highest Positive Correlation)

```
#track.b <- track.b[track.b$id != "uig", ] # remove the outlier Uyghur (uig)

track.b.positive.detailed <- ggplot(track.b, aes(x = CR_msp, y = SBS_INF)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = loess, alpha = 0.3) +
  geom_label_repel(data = track.b[track.b$language == "Chinese"
```

```
                                  | track.b$language == "Vietnamese"
                                  | track.b$language == "English"
                                  | track.b$language == "Russian"
                                  | track.b$language == "Old Church Slavonic"
                                  | track.b$language == "Basque"
                                  | track.b$language == "Finnish"
                                  | track.b$language == "Turkish"
                                  | track.b$language == "Latin"
                                  | track.b$language == "Uyghur"
                                  | track.b$language == "Ancient Greek", ],
                  min.segment.length = 0,
                  #nudge_x = 0.1,
                  aes(label = language),
                  size = 3) +
  ggtitle("b) High Positive Correlation Track B (r = 0.95)") +
  xlab("Mean Size of Morphological Paradigms (CR_msp)") +
  ylab("Inflectional Entropy (SBS_INF)") +
  theme(legend.position = "none")
# track.b.positive.detailed
```

Some comments: This plot shows the correlation between the so-called Mean Size of Morphological Paradigms (MSP), which is defined by CR as "simply the number of word-form types divided by the number of lemma types", and the difference in unigram entropy of word tokens in the original texts and the lemmatized texts (INF) as defined by SBS. It is certainly not unexpected, but reassuring, to see these measures highly correlated. The outlier to the high end Uyghur (uig) is likely *not* an artifact, as this language indeed has many productive morphological paradigms. Other languages to the high end of morphological complexity include Ancient Greek (grc), Classical Latin (lat), Turkish (tur), and Old Church Slavonic (chu). Languages to the low end are Vietnamese (vie), Indonesian (ind), Mandarin Chinese (cmn), and Afrikaans (afr). Note that the very low morphological complexity scores of Korean (kor) are an artifact of the way the Korean data is presented in the UD. Namely, the "lemmas" given for Korean are actually merely morphologically segmented forms rather than inflectionally neutralized forms as for the other languages. Thus, it makes sense that the MSP is exactly 1 and the INF is 0.

TRACK A (Highest Negative Correlation)

```
track.a.negative.detailed <- ggplot(track.a, aes(x = GM_TTR_fullyparallelised, y = O_SID)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = loess, alpha = 0.3) +
  geom_label_repel(data = track.a,
                  min.segment.length = 0,
                  #nudge_x = 0.1,
                  aes(label = language),
                      size = 3) +
  ggtitle("c) High Negative Correlation Track A (r = -0.65)") +
  xlab("Type-Token-Ratio (GM_TTR_fullyparallelized)") +
  ylab("Syllable Information Density (O_SID)") +
  theme(legend.position = "none") +
  theme(legend.position = "none") +
  xlim(0.05, 0.4)
# track.a.negative.detailed
```

Some comments: This plot shows a negative correlation between type-token-ratios in parallel texts

(GM_TTR_fullyparallelised) and syllable information density (O_SID). This can be seen as a trade-off between the diversity of word types and the information carried by syllables. Languages with agglutinative morphology, e.g. Finnish and Turkish have many word types, but low syllable information density. Languages with rather isolating morphology, e.g. English and French, have fewer word types, but more information-dense syllables.

Track B (Highest Negative Correlation)

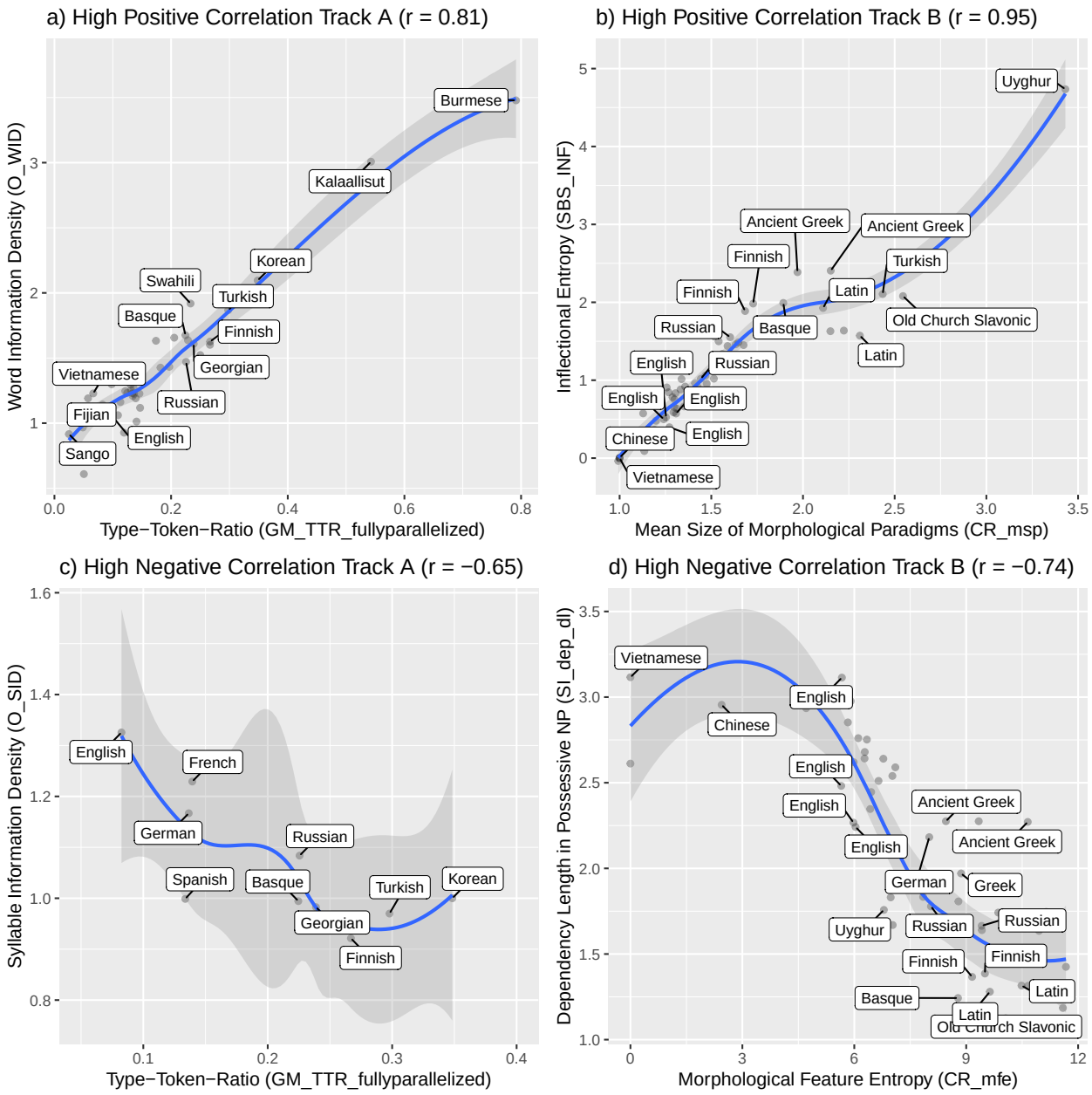
```
track.b <- track.b[track.b$language != "Korean", ] # remove the outlier Korean

track.b.negative.detailed <- ggplot(track.b, aes(x = CR_mfe, y = SI_dep_dl)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = loess, alpha = 0.3) +
  geom_label_repel(data = track.b[track.b$language == "Chinese"
                                  | track.b$language == "Vietnamese"
                                  | track.b$language == "German"
                                  | track.b$language == "English"
                                  | track.b$language == "Hungarian"
                                  | track.b$language == "Greek"
                                  | track.b$language == "Russian"
                                  | track.b$language == "Old Church Slavonic"
                                  | track.b$language == "Basque"
                                  | track.b$language == "Finnish"
                                  | track.b$language == "Turkish"
                                  | track.b$language == "Latin"
                                  | track.b$language == "Uyghur"
                                  | track.b$language == "Ancient Greek", ],
                   min.segment.length = 0,
                  #nudge_x = 0.1,
                  aes(label = language),
                  size = 3) +
  ggtitle("d) High Negative Correlation Track B (r = -0.74)") +
  xlab("Morphological Feature Entropy (CR_mfe)") +
  ylab("Dependency Length in Possessive NP (SI_dep_dl)") +
  theme(legend.position = "none")
# track.b.negative.detailed
```

Combine Scatterplots

We here combine the four scatterplots with some of the highest positive and negative correlations in one panel.

```
scatterplots <- grid.arrange(track.a.positive.detailed, track.b.positive.detailed,
                             track.a.negative.detailed, track.b.negative.detailed, nrow = 2)
```



Safe to file.

```
ggsave("Figures/Corrs/scatterplots.pdf", scatterplots, dpi = 300,
       scale = 1, width = 9, height = 9, device = cairo_pdf)
```

Conclusions

Some more general observations based on these analyses include:

- Many of the measures proposed by the same participants highly correlate. This is the case, for instance, for the measures proposed by GM in Track A, but also measures of BV in Track B. In the case of GM, this is because many of the measures are virtually the same, but with minor shades of modification. In the case of BV, while at first sight the measures seem to conceptually differ, they essentially boil down

to the same underlying causes. For example, the number of tokens in a sentence highly predicts the average maximal depth of a tree over the sentence. So, arguably most of these positive intra-participant correlations are driven by redundancy in the proposed measures.

- There are several strong positive correlations between simple measures relating to the number of types and tokens (GM_TTR_fullyparallelised, BV_n_tokens, etc.), and measures of information density (O_WID, S_idSD). Interestingly, this is the case for both tracks, since Oh used the Bible texts, and Semenuks used the UD. Information density is generally assumed to be a measure which has psycholinguistic relevance in terms of language processing. However, the fact that it is highly predictable by some of the simplest word frequency measures (TTR) potentially goes to show that the underlying principles driving complexity are fairly similar.
- A negative correlation which seems robust in Track A is found between syllable information density (O_SID) and measures of lexical diversity like TTR (although there are few data points in O_SID). This is potential evidence for a trade-off between syllable complexity and word complexity reported also in earlier studies. A negative correlation in Track B which seems both robust and potentially interesting is that the dependency lengths in noun phrases with marked possessives (SI_dep_dl) apparently are in an inverse relationship with different measures of inflectional complexity.

References

McDonald, J.H. (2014). Handbook of Biological Statistics (3rd ed.). Sparky House Publishing, Baltimore, Maryland. online at http://www.biostathandbook.com