Code for Variety and Mainstays of the R Developer Community

Lijin Zhang

Xueyang Li

Zhiyong Zhang

December 06, 2022

Here we provided the code and data for the paper titled “Variety and Mainstays of the R Developer Community”.

```
## package and function perparation
library(dplyr)
library(tidytext)
library(janeaustenr)
library(ggplot2)
library(scales)
library(textdata)
library(tidyr)
library(igraph)
library(ggraph)
library(tm)
library(stringr)
library(wordcloud)
library(topicmodels)
library(wordcloud2)
library(spacyr) # for lemmatizing verbs / aux by importing python pa
        ckages
library(SemNetCleaner) # for lemmatizing nouns



# function for save data
write.csv.utf8.BOM <- function(df, filename){
    con <- file(filename, "w")
    tryCatch({
        for (i in 1:ncol(df))
            df[,i] = iconv(df[,i], to = "UTF-8")
        writeChar(iconv("\ufeff", to = "UTF-8"), con, eos = NULL)
        write.csv(df, file = con, row.names=FALSE)
    },finally = {close(con)})
}
# function for spliting the name and email of maintainer
split_mt <- function(maintainer){
  mt2name_email = strsplit(maintainer,'<')[[1]]
  name = mt2name_email[1]
  email = strsplit(mt2name_email[2],'>')[[1]]

  return(mt_name_email = list(name = name, email = email))
}
```

Extracted the data from CRAN on 2022-11-27.

```
# extract the pkg information from cran on 2022-11-17
cran <- tools::CRAN_package_db()
cran <- cran[!duplicated(cran$Package), ] # Remove duplicated packag
        es
cat(paste0('There are ', dim(cran)[1], ' packages on CRAN at ', Sys.
        time()))
# There are 18898 packages on CRAN at 2022-11-27 17:01:43
save.image("pkg_221127.RData")
write.csv(cran, 'pkg_221127.csv', row.names=F)
```

Load the extracted data which can be obtained from https://github.com/zhanglj37/R_Developer_Community.

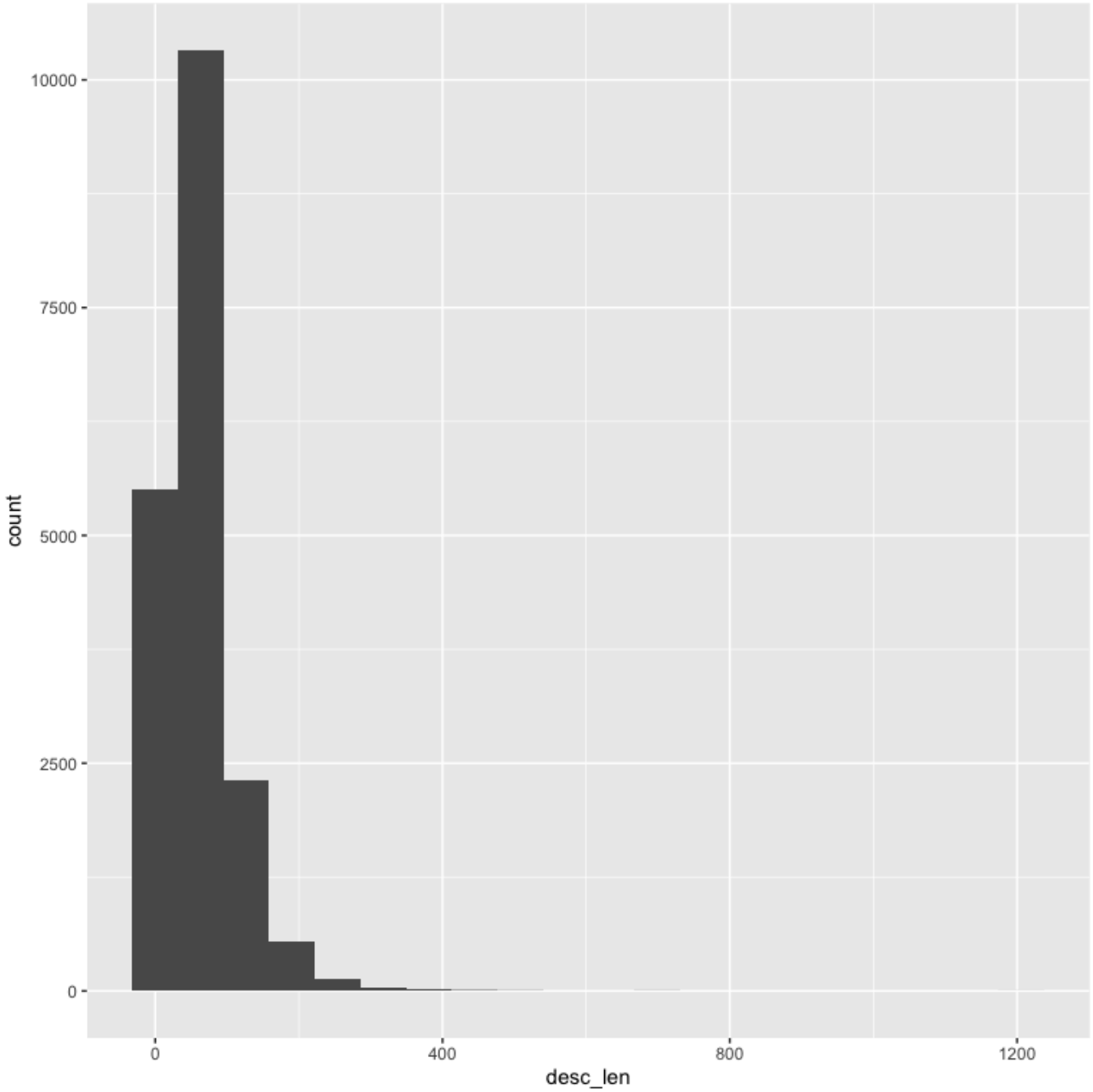
```
load("pkg_221127.RData")
```

Text Mining

1 Data Cleaning

1.1 Length of the descriptions

```
desc = cran[,c("Package","Description")]
desc_len = str_count(desc$Description, '\\w+')
ggplot(as.data.frame(desc_len)) + geom_histogram(aes(x=desc_len), bi
        ns=20) + theme_set(theme_bw())
```




```
summary(desc_len)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   28.00   49.00   60.15   78.00 1207.00
```

1.2 Convert upper cases, delete web links and doi

```
desc[,"Description"] = tolower(desc[,"Description"])
# delete links
```

```
desc[,"Description"] = str_remove_all(str_remove_all(desc[,"Descript
        ion"], "(?<=(http)).+?(?=\\s|$)"), fixed("http"))
# delete links
desc[,"Description"] = str_remove_all(str_remove_all(desc[,"Descript
        ion"], "(?<=(www\\.)).+?(?=\\s|$)"), fixed("www."))
# delete doi
desc[,"Description"] = str_remove_all(str_remove_all(desc[,"Descript
        ion"], "(?<=(doi)).+?(?=\\s|$)"), fixed("doi"))
# replace "\\n  " with " "
for(i in 1:nrow(desc)){
    desc[i,"Description"] = str_replace_all(desc[i,"Description"], "
        \\s+", " ")
}
```

1.3 Lemmatization

```
desc_sentence = desc[,"Description"]
names(desc_sentence) = desc[,"Package"]


# lemmatize verbs (applied, applies --> apply) and aux (was --> be)
         using spacyr spacy_parse()
# lemmatize NOUNs using SemNetCleaner singularize()
# https://spacy.io/usage
desc_token = spacy_parse(desc_sentence)
desc_token2 = desc_token
for (i in 1:nrow(desc_token2)){
    if(desc_token2[i,"pos"]=="VERB" || desc_token2[i,"pos"]=="AUX" )
        {
        desc_token2[i,"lemma"] = desc_token[i,"lemma"]
    }else if(desc_token2[i,"pos"]=="NOUN"){
        desc_token2[i,"lemma"] = singularize(as.character(desc_token
        2[i,"token"]))
    }else if(desc_token2[i,"pos"]=="PROPN"){
        desc_token2[i,"lemma"] = singularize(as.character(desc_token
        2[i,"token"]))
    }else if(desc_token2[i,"pos"]=="PRON"){
        desc_token2[i,"lemma"] = singularize(as.character(desc_token
        2[i,"token"]))
    }else{
        desc_token2[i,"lemma"] = desc_token[i,"token"]
    }
}

colnames(desc_token2) = c("doc_id", "sentence_id", "token_id", "toke
        n", "word", "pos", "entity")
```

```
# transferred desc_token2 back into sentence for phrase frequency an
        alysis
desc_lemma = desc
for (i in 1:nrow(desc)){
    loc_sentence = which(desc_token2[,"doc_id"] == desc[i,"Package"
        ])
    # combine words into sentense
    desc_lemma[i,"Description"] = paste(desc_token2[loc_sentence, "w
        ord"], collapse = " ")
}
```

1.4 Delete numbers and symbols

```
delete_pos = c("PUNCT", "NUM", "SYM")
pos_loc1 = which(desc_token2[,"pos"]==delete_pos[1])
pos_loc2 = which(desc_token2[,"pos"]==delete_pos[2])
pos_loc3 = which(desc_token2[,"pos"]==delete_pos[3])

loc_token = c(1:nrow(desc_token2))
# detect numbers # numbers that are not detected by spacy_sparse
numbers_only <- function(x) !grepl("\\D", x)
loc_number = loc_token[numbers_only(desc_token2[,"word"])]
# detect space
space_only <- function(x) !grepl("\\S", x)
loc_space = loc_token[space_only(desc_token2[,"word"])]
# detect symbols # symbols that are not detected by spacy_sparse
no_str_num <- function(x) !grepl("\\w", x)
loc_no_str_num = loc_token[no_str_num(desc_token2[,"word"])]
one_char1 <- function(x) grepl("^\\w\\W$", x)
one_char2 <- function(x) grepl("^\\W\\w$", x)
loc_onechar1 = loc_token[one_char1(desc_token2[,"word"])]
loc_onechar2 = loc_token[one_char2(desc_token2[,"word"])]

pos_delete = c(pos_loc1, pos_loc2, pos_loc3, loc_number, loc_space,
         loc_no_str_num, loc_onechar1, loc_onechar2)

token_clean = desc_token2[-pos_delete,]

token_clean = tibble(token_clean)
```

1.5 remove common stopwords

```
# our_stop_words
our_stop <- c("et", "etc", "al", "i.e.", "e.g.",
    "package", "provide", "method", "function", "approach", "referen
        ce", "implement", "contain", "include")

token_nostop <- token_clean %>%
    anti_join(stop_words) %>%
    filter(!word %in% our_stop)  %>%
    filter(!word %in% stop_words$word)
```

```
## Joining, by = "word"
```

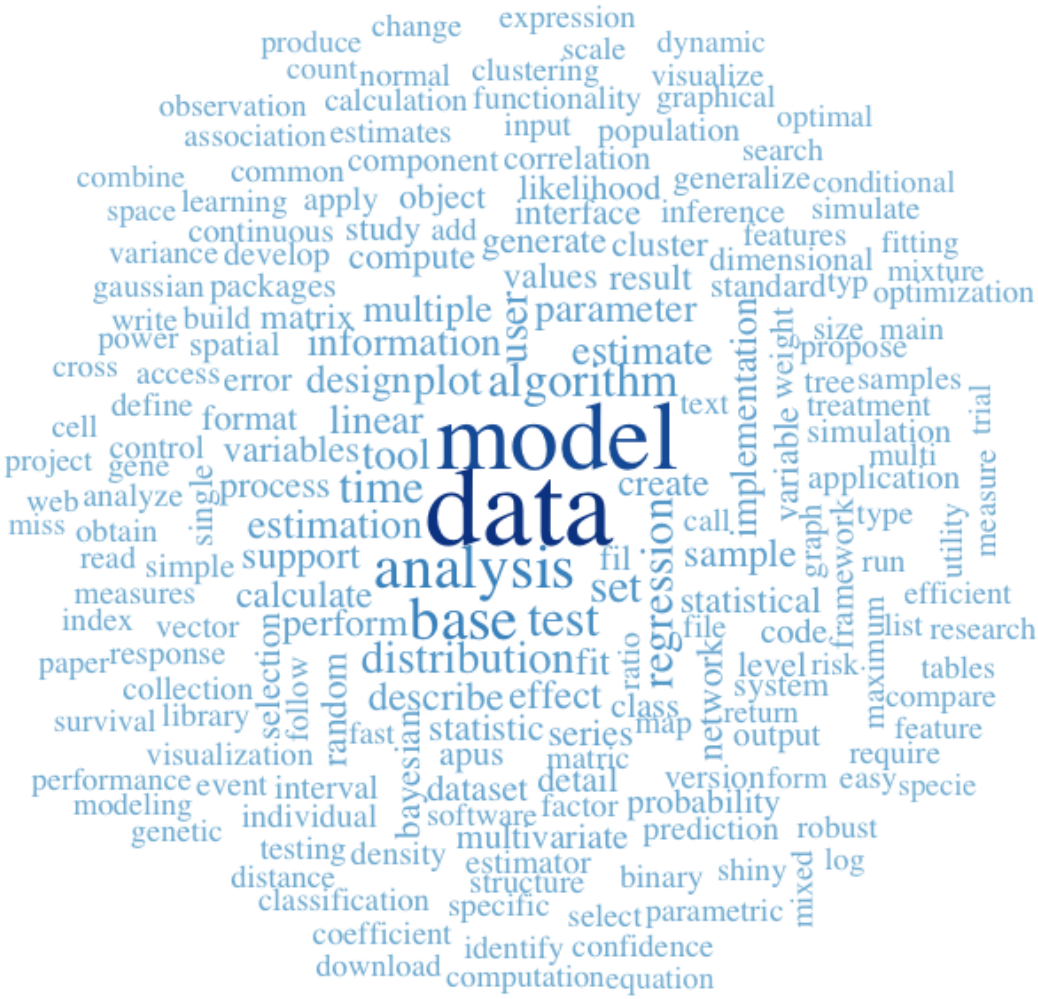
```
#  TF-IDF value
#token_count = token_nostop %>%
#  count(doc_id, word)

#total_words = token_count %>%
#  group_by(doc_id) %>%
#  summarize(total = sum(n))

#token_count2 <- left_join(token_count, total_words)

#token_tf_idf <- token_count2 %>%
#  bind_tf_idf(word, doc_id, n)

#temp = token_tf_idf %>%
#  select(-total) %>%
#  arrange(desc(tf_idf))
```

2 Frequency Analysis

2.1 Word Frequency

```
word_totals <- token_nostop %>%
    count(word, sort = TRUE)
```

```
word_totals_sub = word_totals[c(1:200),]

wordcloud(word_totals_sub$word, word_totals_sub$n, random.color=FALS
        E, random.order=FALSE, color=colorRampPalette(brewer.pal(8,
        "Blues"))(50)[25:50], scale=c(4,1),family="serif")
```



2.2 Phrase Frequency

Get Phrases and Frequency

save the frequency results and select meaningful phrases manully

```
#### 2-gram
desc_lemma.2gram <- unnest_tokens(desc, word, Description, token =
        "ngrams", n = 2)
desc_lemma.2gram %>% filter(word=="p value" | word == "p values") %
        >% count(word, sort = T)
```

```
##       word   n
## 1 p values 233
## 2  p value 161
```

```
desc_lemma.2count <- desc_lemma.2gram %>% separate(word, c("word1",
        "word2"), sep = " ")  %>%
    filter(!word1 %in% stop_words$word & !word2 %in% stop_words$wor
        d) %>% # remove phrase consist with all stop words
    unite(word, word1, word2, sep = " ") %>%
    count(word, sort = TRUE)
head(desc_lemma.2count)
```

```
##                 word    n
## 1        time series 1018
## 2          data sets  523
## 3 maximum likelihood  410
## 4  regression models  395
## 5      data analysis  393
## 6        monte carlo  369
```

```
#write.csv(desc_lemma.2count, 'freq/2word_freq.csv', row.names=F)

#### 3-gram
desc_lemma.3gram <- unnest_tokens(desc, word, Description, token =
        "ngrams", n = 3)

desc_lemma.3count <- desc_lemma.3gram %>% separate(word, c("word1",
        "word2", "word3"), sep = " ")  %>%
    filter(!word1 %in% stop_words$word & !word2 %in% stop_words$word
        & !word3 %in% stop_words$word) %>% # remove phrase consist
         with all stop words
    unite(word, word1, word2, word3, sep = " ") %>%
    count(word, sort = TRUE)
head(desc_lemma.3count)
```

```
##                             word   n
## 1             chain monte carlo 140
## 2            markov chain monte 140
## 3     generalized linear models 133
## 4              time series data 133
## 5 maximum likelihood estimation 119
## 6  principal component analysis  98
```

```
#write.csv(desc_lemma.3count, 'freq/3word_freq.csv', row.names=F)

# select meaningful phrases and combine phrases with same meaning ma
        nully
word2 = read.csv("freq/2WordTermTop30.csv")
word3 = read.csv("freq/3WordTermTop30.csv")
```

```
### 456-gram
desc_lemma.4gram <- unnest_tokens(desc_lemma, word, Description, tok
        en = "ngrams", n = 4)
desc_lemma.4count <- desc_lemma.4gram %>%  count(word, sort = TRUE)
#write.csv(desc_lemma.4count, 'freq/4word_freq.csv', row.names=F)

desc_lemma.5gram <- unnest_tokens(desc_lemma, word, Description, tok
        en = "ngrams", n = 5)
desc_lemma.5count <- desc_lemma.5gram %>%  count(word, sort = TRUE)
#write.csv(desc_lemma.5count, 'freq/5word_freq.csv', row.names=F)

desc_lemma.6gram <- unnest_tokens(desc_lemma, word, Description, tok
        en = "ngrams", n = 6)
desc_lemma.6count <- desc_lemma.6gram %>%  count(word, sort = TRUE)
#write.csv(desc_lemma.6count, 'freq/6word_freq.csv', row.names=F)
```

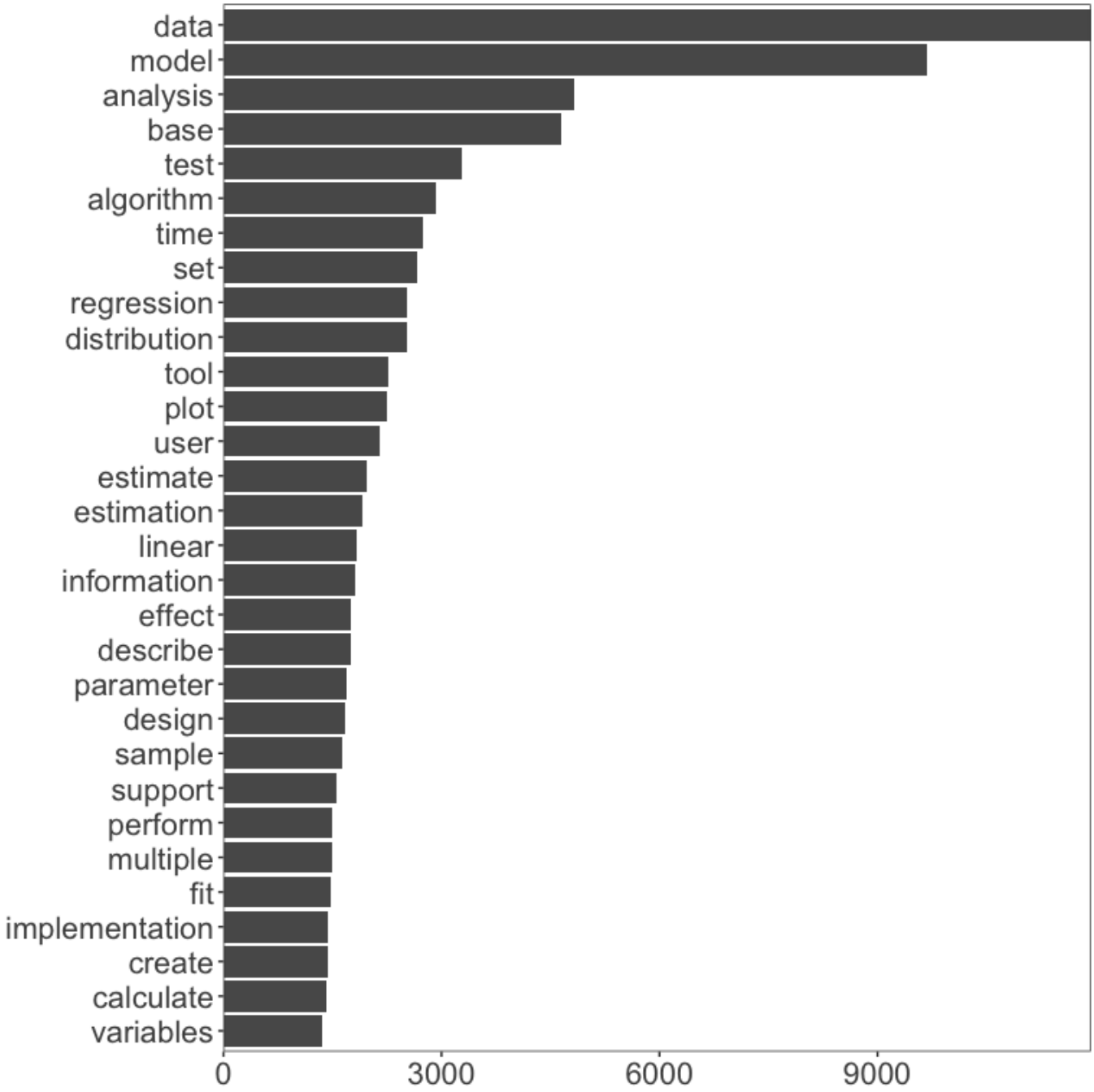
Frequency Figures

```
word1_plot = word_totals %>%
    top_n(30) %>%
    mutate(word = reorder(word, n)) %>%
    ggplot(aes(word, n)) +
    theme(axis.title.x = element_text(face = 'bold'), axis.title.y =
        element_text(face = 'bold'), axis.title = element_text(face
        = 'bold'), text=element_text(size=20)) +
    geom_col(show.legend = FALSE) +
    coord_flip() +
```

```
    scale_y_continuous(expand = c(0,0)) +
    labs(x = NULL, y = NULL) +
    theme(panel.grid=element_blank())
```

```
## Selecting by n
```

```
word1_plot
```

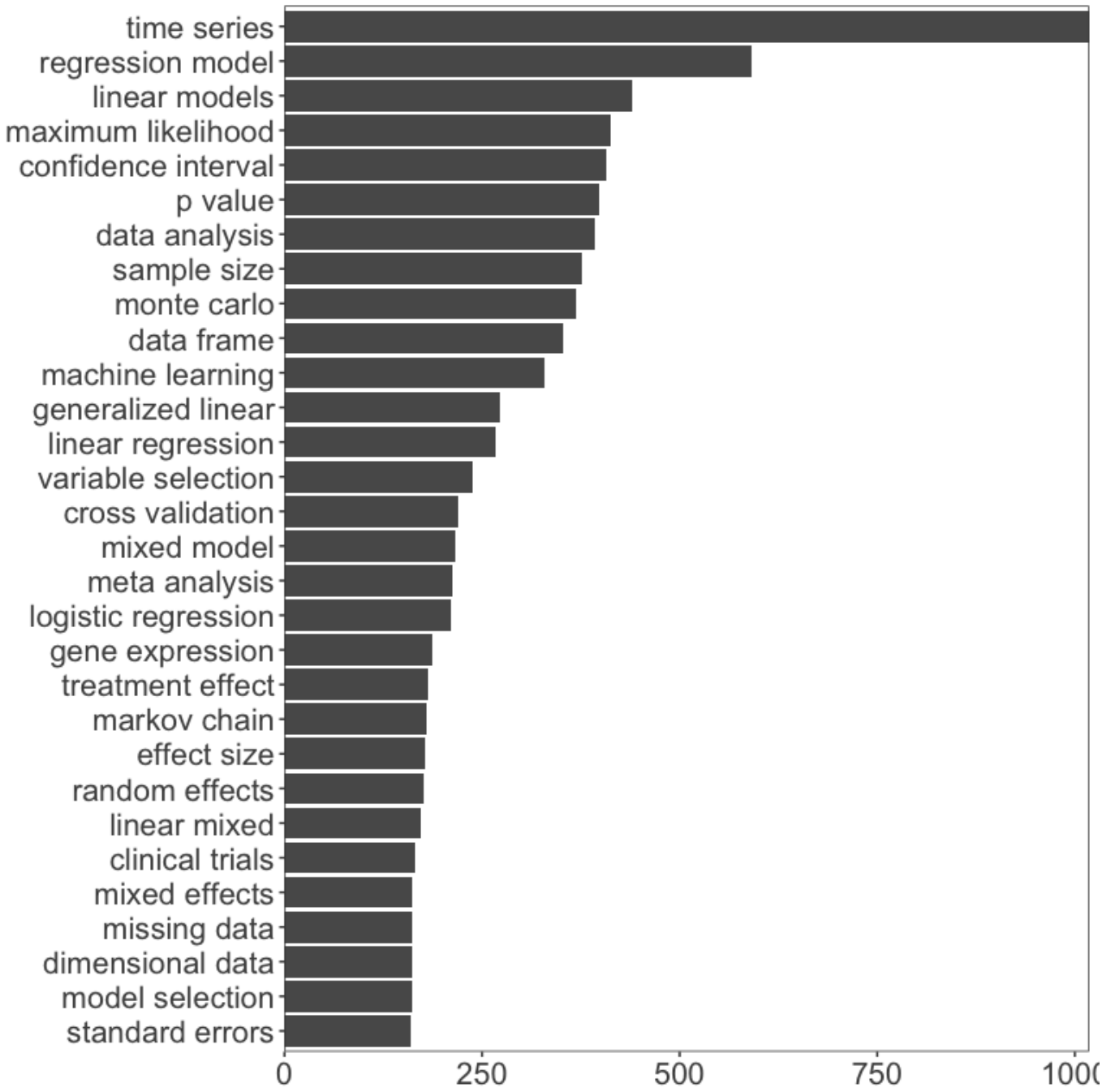



```
word2_plot = word2 %>%
    top_n(30) %>%
    mutate(term = reorder(term, frequency)) %>%
```

```
    ggplot(aes(term, frequency)) +
    theme(axis.title.x = element_text(face = 'bold'), axis.title.y =
        element_text(face = 'bold'), axis.title = element_text(face
        = 'bold'), text=element_text(size=20)) +
    geom_col(show.legend = FALSE) +
    coord_flip() +
    scale_y_continuous(expand = c(0,0)) +
    labs(x = NULL, y = NULL) +
    theme(panel.grid=element_blank())
```

```
## Selecting by frequency
```

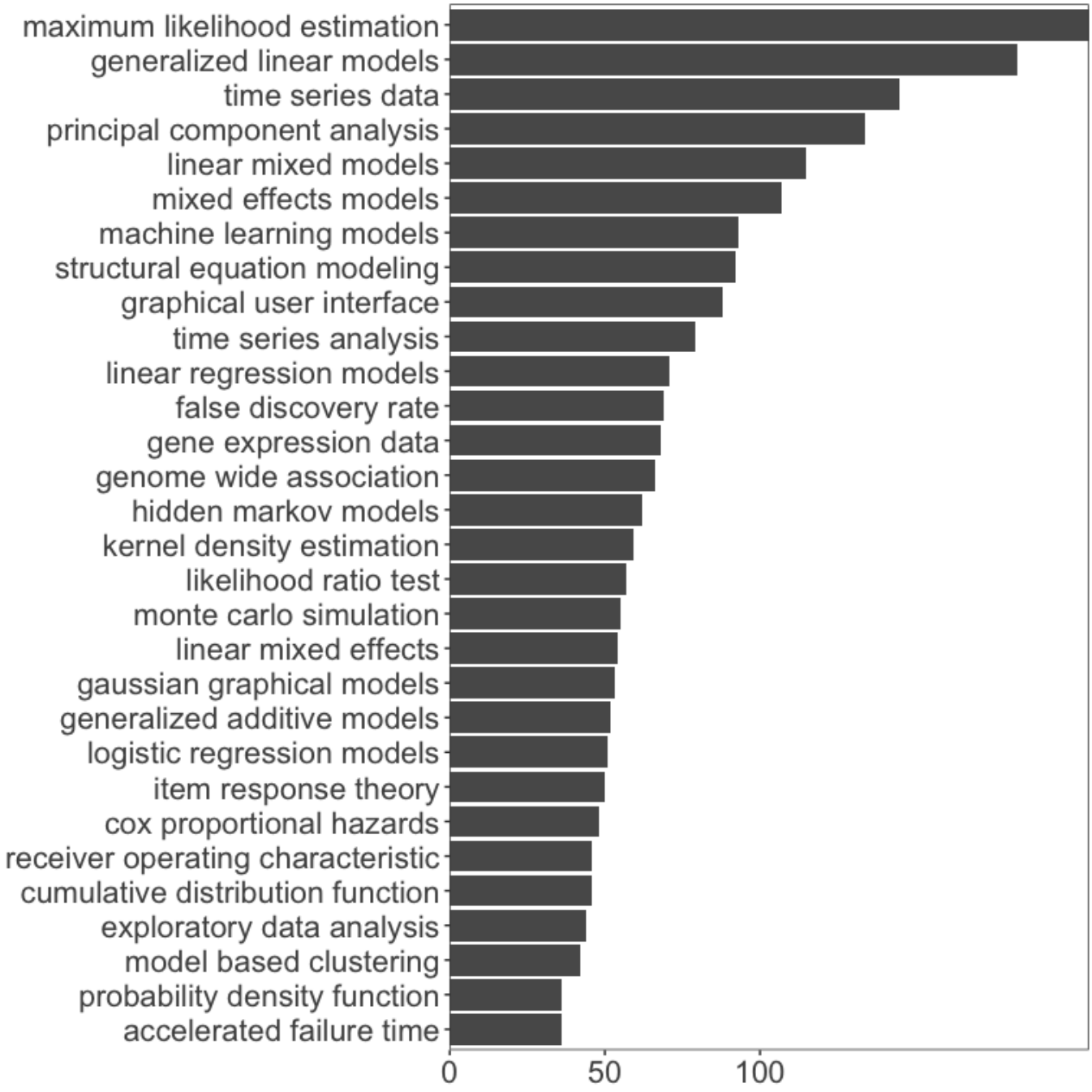
```
word2_plot
```




```
word3_plot = word3 %>%
    top_n(30) %>%
    mutate(term = reorder(term, frequency)) %>%
    ggplot(aes(term, frequency)) +
    theme(axis.title.x = element_text(face = 'bold'), axis.title.y =
        element_text(face = 'bold'), axis.title = element_text(face
        = 'bold'), text=element_text(size=20)) +
    geom_col(show.legend = FALSE) +
    coord_flip() +
    scale_y_continuous(expand = c(0,0),breaks = c(0, 50,100)) +
    labs(x = NULL, y = NULL) +
    theme(panel.grid=element_blank())
```

```
## Selecting by frequency
```

```
word3_plot
```




```
ggsave("freq/word1.pdf", word1_plot, dpi = 400,width=5, height = 7)
ggsave("freq/word2.pdf", word2_plot, dpi = 400,width=5, height = 7)
ggsave("freq/word3.pdf", word3_plot, dpi = 400,width=5, height = 7)
```

3 Topic Modeling

3.1 Determine the number of topics

Since the cross-validation is very time-consuming, here we just show the code but didn't run it.

```
# data perparation
library(topicmodels)
desc_dtm_select = token_nostop[,c("doc_id","word")]
desc_dtm_select2 = desc_dtm_select %>%
    group_by(doc_id) %>%
    count(word) %>% ungroup()

desc_dtm = desc_dtm_select2 %>% cast_dtm(doc_id, word, n)
tm::inspect(desc_dtm)
```

```
## <<DocumentTermMatrix (documents: 18898, terms: 42268)>>
## Non-/sparse entries: 437158/798343506
## Sparsity           : 100%
## Maximal term length: 65
## Weighting          : term frequency (tf)
## Sample             :
##              Terms
## Docs          algorithm analysis base data distribution model regre
##   arthistory          0        0    0    1            0     0
##   ff                  0        0    0    6            0     0
##   frailtypack         0        1    0   10            2    26
##   KoulMde             0        4    3    1            0     8
##   lactcurves          0        1    2    1            0     9
##   MHCtools            0        2    2   14            0     2
##   mMARCH.AC           0        1    0   13            0     0
##   RJafroc             1        6    1    9            0     6
##   rSHAPE              0        1    0    0            0     2
##   spatstat            0        2    0    6            0    12
##              Terms
## Docs          test time
##   arthistory     0    0
##   ff             0    0
##   frailtypack    0    3
##   KoulMde        0    4
```

```
##   lactcurves     1    0
##   MHCtools       0    0
##   mMARCH.AC      0    1
##   RJafroc        0    0
##   rSHAPE         2    2
##   spatstat       5    1
```

```
k_max = 30
k.topics <- 2:k_max
doc_num = nrow(cran)

folding4 <- rep(1:4, each = round(doc_num/5,0))
folding = c(folding4, rep(5, (doc_num-length(folding4))))
```

```
## parallel computation
runonce_paral <- function(sed) {
    res <- NULL
    for (k in k.topics) {
        for (fold in 1:5) {
            testing.dtm <- which(folding == fold)
            training.dtm <- which(folding != fold)

            training.model <- LDA(desc_dtm[training.dtm, ], k = k, c
        ontrol = list(seed = sed*100))
            test.model <- LDA(desc_dtm[testing.dtm, ], model = train
        ing.model, control = list(estimate.beta = FALSE))

            prep = perplexity(test.model)
            cat(paste(sed, k, fold, prep, "\n", sep="\t"),append=T)
            res <- rbind(res, c(sed, k, fold, prep))
        }
    }
    return(res)
}


library(doParallel)

ncores <- 32

#switch between %do% (serial) and %dopar% (parallel)
if (ncores == 1){  #serial
    `%is_par%` <- `%do%`
}else{  #parallel
    `%is_par%` <- `%dopar%`
```

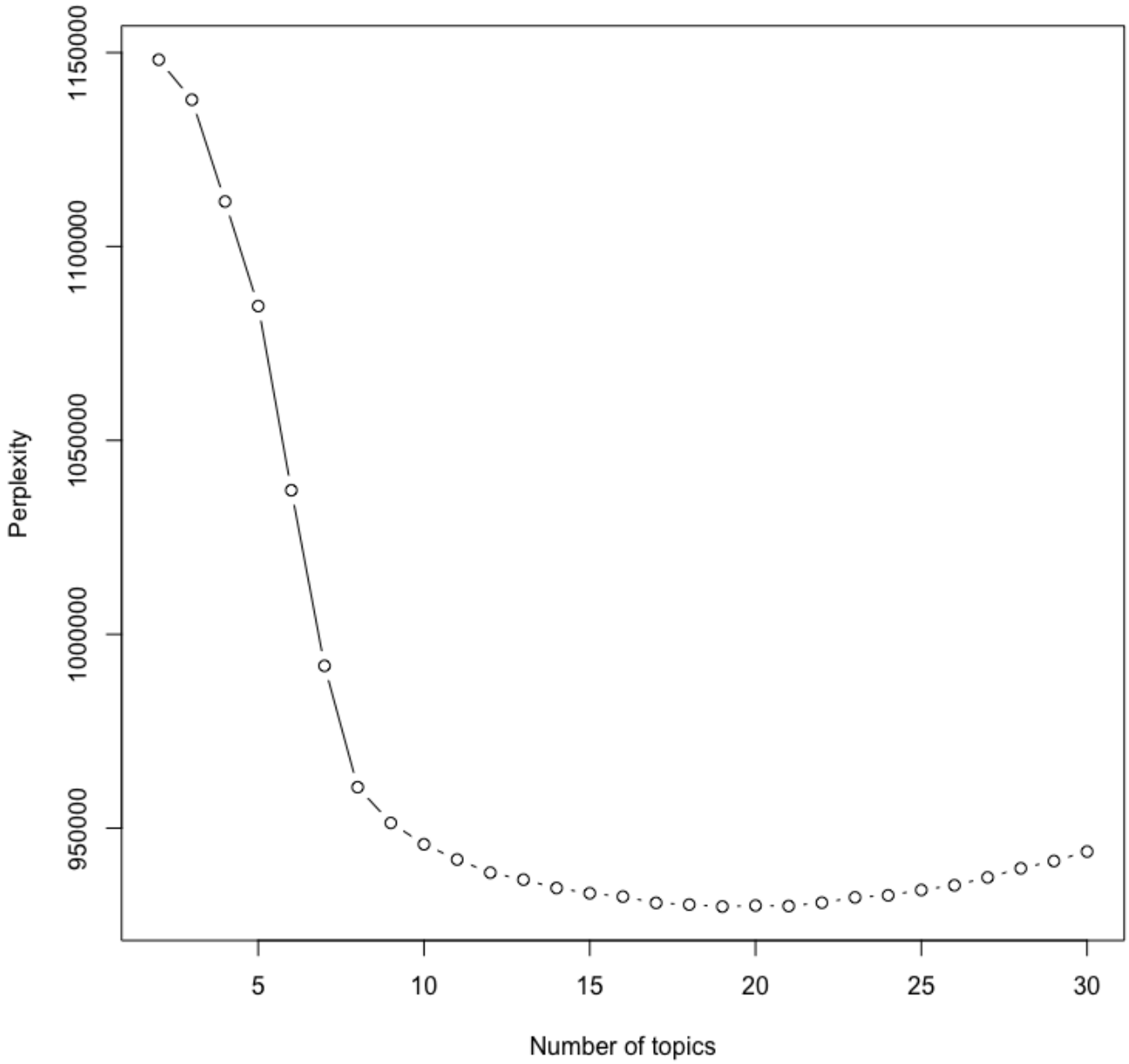
```
    cl <- makeCluster(ncores)#, outfile="res.txt")
    registerDoParallel(cores = ncores)
}

foreach(sed = 1:100, .packages = c("topicmodels") ) %is_par%{
    sink("res.txt", append=TRUE) # divert the output to the log file
    res = runonce_paral(sed)

}

if(ncores > 1) stopCluster(cl)
```

Import the results of cross-validation

```
res = read.table("lda/res.txt", sep="\t")

colnames(res) = c('sed', 'topics', 'fold', 'perplexity')

total.perp <- NULL
for(sedi in 1:100){
    for(ki in 2:k_max){
        loc = which(res[,'sed']==sedi & res[,'topics']==ki)
        total.perp = rbind(total.perp, c(sedi, ki, mean(res[loc,'per
        plexity'])))
    }
}

# round(total.perp)

total.perp.all = tapply(total.perp[, 3], total.perp[, 2], mean)
sort(total.perp.all)
```

```
##        19        21        20        18        17        22
##  929789.7  929932.2  930054.1  930299.9  930744.0  930788.1  932151
##        24        15        25        14        26        13
##  932677.5  933219.6  934061.0  934571.9  935270.0  936731.0  937306
##        28        29        11        30        10         9
##  939651.4  941527.3  941917.2  943974.6  945859.9  951360.7  960580
##         6         5         4         3         2
## 1037146.3 1084631.1 1111598.1 1137845.6 1148172.4
```

```
plot(2:k_max, total.perp.all, type = "b", xlab = "Number of topics",
        ylab = "Perplexity")
```




```
num_top = rep(0,k_max-1)
names(num_top) = c(2:k_max)
for(sedi in 71:100){
    loc = which(total.perp[,1]==sedi)
    temp = total.perp[loc,3]
    names(temp) = total.perp[loc,2]
    num_top[as.numeric(names(sort(temp)[1]))-1] = num_top[as.numeric
        (names(sort(temp)[1]))-1] + 1
}
# times that the number was selected as the best
num_top
```

```
##  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 2
##  0  0  0  0  0  0  0  0  1  0  3  2  2  1  2  3  1  4  3  2  1  2
## 28 29 30
##  0  0  0
```

3.2 Topic 20

```
desc_lda20 <- LDA(desc_dtm, k = 20, control = list(seed = 1000))
```

```
desc_topics <- tidy(desc_lda20, matrix = "beta")
desc_topics
```

```
## # A tibble: 845,360 × 3
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 analyze 1.05e- 3
##  2     2 analyze 1.57e- 3
##  3     3 analyze 1.42e-10
##  4     4 analyze 8.08e-12
##  5     5 analyze 6.57e-30
##  6     6 analyze 2.62e- 3
##  7     7 analyze 4.81e- 3
##  8     8 analyze 6.18e- 4
##  9     9 analyze 2.02e- 4
## 10    10 analyze 1.87e-12
## # … with 845,350 more rows
```

```
desc_terms <- desc_topics %>% group_by(topic) %>% top_n(100, beta) %
        >% ungroup() %>%
    arrange(topic, -beta)


library(ggwordcloud)

set.seed(1)
lda_wcloud = ggplot(desc_terms, aes(label = term, size = sqrt(beta)*
        100)) +
  geom_text_wordcloud_area(shape='circle') +
  scale_size_area(max_size = 5) +
  theme_minimal() +
```

```
  facet_wrap(~topic, ncol=4) +
  theme_set(theme_bw()) + theme(panel.grid=element_blank()) +
  theme(panel.grid.major=element_line(colour=NA), panel.spacing=unit
        (0.5, "lines"))

head(desc_terms)
```

```
## # A tibble: 6 × 3
##   topic term         beta
##   <int> <chr>       <dbl>
## 1     1 data      0.0245
## 2     1 book      0.0122
## 3     1 statistic 0.0110
## 4     1 de        0.0103
## 5     1 research  0.00960
## 6     1 national  0.00957
```

Relationships between topic and packages

```
desc_topics_doc <- tidy(desc_lda20, matrix = "gamma") %>% group_by(t
        opic) %>% top_n(100, gamma) %>% ungroup() %>% arrange(topi
        c, -gamma)
head(desc_topics_doc)
```

```
## # A tibble: 6 × 3
##   document      topic gamma
##   <chr>         <int> <dbl>
## 1 arthistory        1 0.995
## 2 bearishTrader     1 0.978
## 3 covidmx           1 0.964
## 4 stockAnalyst      1 0.957
## 5 orloca.es         1 0.954
## 6 malvinas          1 0.953
```

```
desc_alpha <- tidy(desc_lda20, matrix = "gamma")

desc_alpha[which(desc_alpha$document == 'Rcpp'),]
```

```
## # A tibble: 20 × 3
##    document topic   gamma
##    <chr>    <int>   <dbl>
```

```
##  1 Rcpp          1 0.0752
##  2 Rcpp          2 0.00191
##  3 Rcpp          3 0.00191
##  4 Rcpp          4 0.00191
##  5 Rcpp          5 0.00191
##  6 Rcpp          6 0.00191
##  7 Rcpp          7 0.00191
##  8 Rcpp          8 0.00191
##  9 Rcpp          9 0.00191
## 10 Rcpp         10 0.00191
## 11 Rcpp         11 0.00191
## 12 Rcpp         12 0.122
## 13 Rcpp         13 0.00191
## 14 Rcpp         14 0.00191
## 15 Rcpp         15 0.00191
## 16 Rcpp         16 0.771
## 17 Rcpp         17 0.00191
## 18 Rcpp         18 0.00191
## 19 Rcpp         19 0.00191
## 20 Rcpp         20 0.00191
```

Network Analysis

4 Package dependency network and author collaboration network

4.1 Build network and set edges

```
library(cranly)
library(influential)
```

```
##
## Attaching package: 'influential'
```

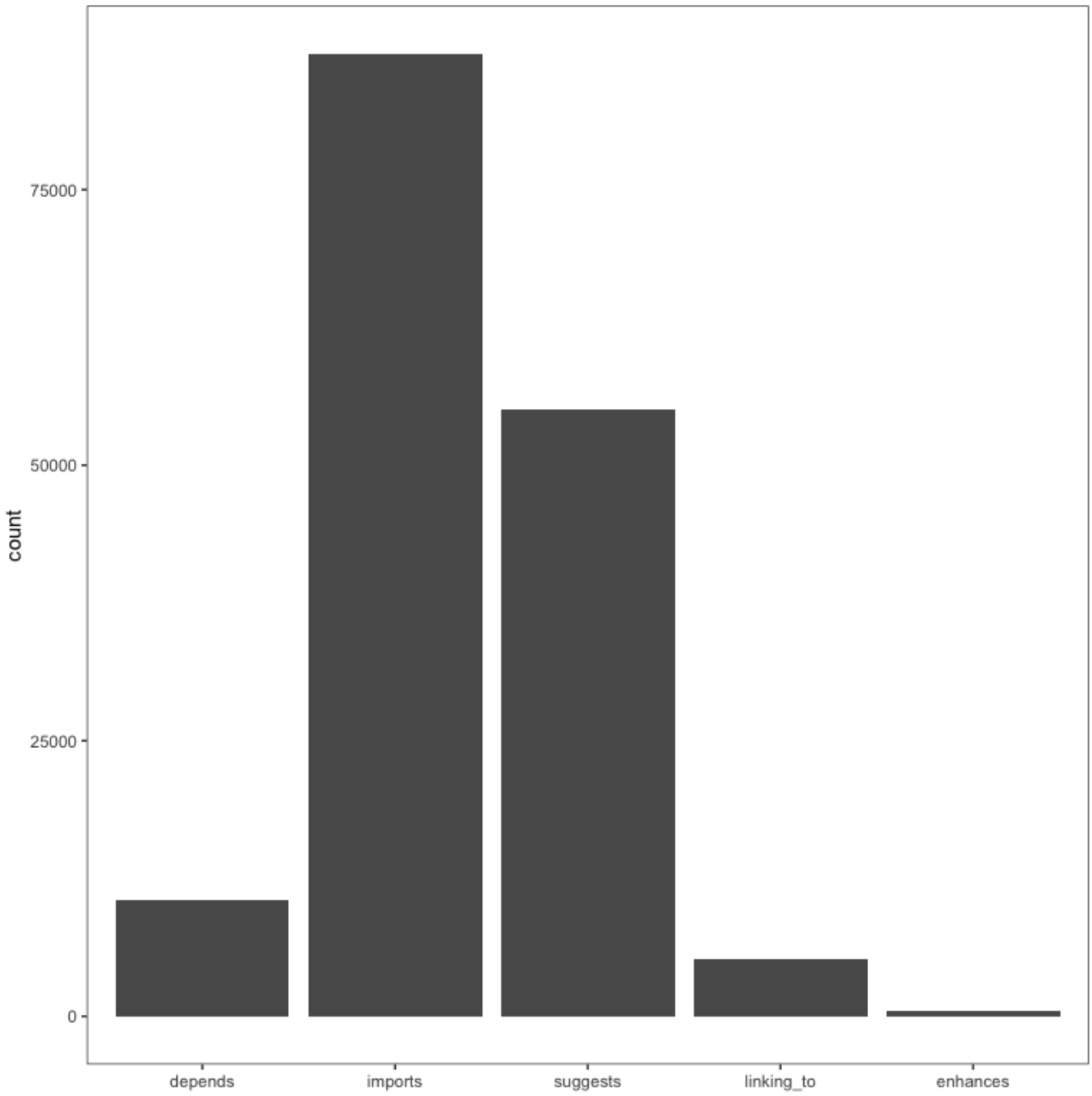
```
## The following objects are masked from 'package:igraph':
##
##     betweenness, graph_from_data_frame
```

```
library(igraph)
load("pkg_221127.RData")  # load cran
cran_ly <- clean_CRAN_db(cran)

pkg_net <- build_network(cran_ly, perspective = "package")
aut_net <- build_network(cran_ly, perspective = "author")

pkg_graph = as.igraph(pkg_net,reverse=TRUE) # reverse the direction
aut_graph = as.igraph(aut_net)


######### flag weights
######### pkg_graph: weight = relationships
pkg_net_type = E(pkg_graph)$type
pkg_net_type_num = pkg_net_type
pkg_type = c("depends", "imports", "suggests", "linking_to", "enhanc
        es")
pkg_type_num = c(5:1)
for (typei in 1:length(pkg_type)){
    loc = which(pkg_net_type==pkg_type[typei])
    pkg_net_type_num[loc] = pkg_type_num[typei]
}
pkg_graph_w = pkg_graph %>% set_edge_attr("weight", value = pkg_net_
        type_num)


######### special cases: same anthor with different names (Gábor Csá
        rdi and Kirill Müller)
aut_adj_matrix = as_adjacency_matrix(aut_graph)
loc1 = which(colnames(aut_adj_matrix)=="Gabor Csardi")
loc2 = which(colnames(aut_adj_matrix)=="Gábor Csárdi")
loc11 = which(aut_adj_matrix[loc1,]>0)
loc21 = which(aut_adj_matrix[loc2,]>0)
aut_adj_matrix[loc2,loc11]=aut_adj_matrix[loc2,loc11]+aut_adj_matrix
        [loc1,loc11]
aut_adj_matrix[loc11,loc2]=aut_adj_matrix[loc11,loc2]+aut_adj_matrix
        [loc11,loc1]
aut_adj_matrix = aut_adj_matrix[-loc1, -loc1]

loc1 = which(colnames(aut_adj_matrix)=="Kirill Muller")
loc2 = which(colnames(aut_adj_matrix)=="Kirill Müller")
loc11 = which(aut_adj_matrix[loc1,]>0)
```

```
loc21 = which(aut_adj_matrix[loc2,]>0)
aut_adj_matrix[loc2,loc11]=aut_adj_matrix[loc2,loc11]+aut_adj_matrix
        [loc1,loc11]
aut_adj_matrix[loc11,loc2]=aut_adj_matrix[loc11,loc2]+aut_adj_matrix
        [loc11,loc1]
aut_adj_matrix = aut_adj_matrix[-loc1, -loc1]

######### aut_graph: weight = number of coauthored pkgs (collaborati
        on intensity)
aut_graph_w = graph_from_adjacency_matrix(aut_adj_matrix,mode="undir
        ected",weighted=TRUE,diag=FALSE)
aut_graph_nw = graph_from_adjacency_matrix(aut_adj_matrix,mode="undi
        rected",weighted=NULL,diag=FALSE)



######### frequency figure
pkg_net_type = factor(pkg_net_type, levels = pkg_type)
table(pkg_net_type)
```

```
## pkg_net_type
##    depends    imports   suggests linking_to   enhances
##      10523      87328      54991       5153        558
```

```
ggplot(as.data.frame(pkg_net_type), aes(pkg_net_type)) +
    geom_bar() +
    labs(x = NULL) +
    theme(panel.grid=element_blank())
```



4.2 Measures of Influence

```
pkgw_betw = betweenness(pkg_graph)
pkgw_dg = degree(pkg_graph_w, mode = c("in"))
pkgw_pgRk = page_rank(pkg_graph_w, weights = E(pkg_graph_w)$weight)
pkgw_eigen = eigen_centrality(pkg_graph_w, weights = E(pkg_graph_w)$
        weight) #directed = TRUE,


autw_betw = betweenness(aut_graph_nw) # weight = distance, so no wei
        ght was include
autw_dg = degree(aut_graph_w)
autw_pgRk = page_rank(aut_graph_w, weights = E(aut_graph_w)$weight)
```

```
autw_eigen = eigen_centrality(aut_graph_w, weights = E(aut_graph_w)$
        weight)

## sort
pkg_rank_dg = sort(pkgw_dg, decreasing = TRUE)
aut_rank_dg = sort(autw_dg, decreasing = TRUE)
pkg_rank_betw = sort(pkgw_betw, decreasing = TRUE)
aut_rank_betw = sort(autw_betw, decreasing = TRUE)
pkg_rank_eigen = sort(pkgw_eigen[["vector"]], decreasing = TRUE)
aut_rank_eigen = sort(autw_eigen[["vector"]], decreasing = TRUE)
pkg_rank_pgRk = sort(pkgw_pgRk[["vector"]], decreasing = TRUE)
aut_rank_pgRk = sort(autw_pgRk[["vector"]], decreasing = TRUE)

## organize
table_pkg = array(NA, dim=c(50,4))
colnames(table_pkg) = c("In-degree", "Betweenness",
 "Eigenvector", "PageRank")
table_pkg[,"In-degree"] = names(pkg_rank_dg[1:50])
table_pkg[,"Betweenness"] = names(pkg_rank_betw[1:50])
table_pkg[,"PageRank"] = names(pkg_rank_pgRk[1:50])
table_pkg[,"Eigenvector"] = names(pkg_rank_eigen[1:50])
write.csv(table_pkg, "network/50pkg_w_54321.csv")

table_aut = array(NA, dim=c(50,4))
colnames(table_aut) = c("In-degree", "Betweenness",
 "Eigenvector", "PageRank")
table_aut[,"In-degree"] = names(aut_rank_dg[1:50])
table_aut[,"Betweenness"] = names(aut_rank_betw[1:50])
table_aut[,"PageRank"] = names(aut_rank_pgRk[1:50])
table_aut[,"Eigenvector"] = names(aut_rank_eigen[1:50])
write.csv.utf8.BOM(table_aut, "network/50aut_w.csv")
```

```
table_pkg
```

```
##       In-degree       Betweenness       Eigenvector     PageRank
##  [1,] "knitr"         "ggplot2"         "knitr"         "testthat"
##  [2,] "testthat"      "knitr"           "rmarkdown"     "utils"
##  [3,] "rmarkdown"     "broom"           "testthat"      "methods"
##  [4,] "stats"         "dplyr"           "stats"         "stats"
##  [5,] "Rcpp"          "emmeans"         "ggplot2"       "knitr"
##  [6,] "ggplot2"       "testthat"        "dplyr"         "patchSynct
##  [7,] "methods"       "shiny"           "methods"       "rmarkdown"
##  [8,] "dplyr"         "stats"           "utils"         "tools"
##  [9,] "utils"         "rmarkdown"       "Rcpp"          "covr"
## [10,] "graphics"      "survival"        "rlang"         "stringr"
```

```
## [11,] "MASS"          "bayestestR"      "magrittr"      "graphics"
## [12,] "magrittr"      "sf"              "tibble"        "mapmisc"
## [13,] "covr"          "gap"             "tidyr"         "Rcpp"
## [14,] "rlang"         "MASS"            "graphics"      "tis"
## [15,] "tibble"        "targets"         "covr"          "grDevices"
## [16,] "tidyr"         "caret"           "stringr"       "rlang"
## [17,] "stringr"       "multcomp"        "purrr"         "MASS"
## [18,] "grDevices"     "Hmisc"           "MASS"          "ggplot2"
## [19,] "purrr"         "texreg"          "grDevices"     "magrittr"
## [20,] "parallel"      "insight"         "data.table"    "withr"
## [21,] "Matrix"        "parameters"      "jsonlite"      "jsonlite"
## [22,] "data.table"    "enrichwith"      "Matrix"        "parallel"
## [23,] "jsonlite"      "cops"            "shiny"         "htmltools"
## [24,] "RcppArmadillo" "mlr"             "parallel"      "scales"
## [25,] "shiny"         "zoo"             "httr"          "fastcluste
## [26,] "httr"          "earth"           "glue"          "enrichwith
## [27,] "mvtnorm"       "tibble"          "scales"        "cops"
## [28,] "survival"      "AER"             "sf"            "QuasiSeq"
## [29,] "foreach"       "lme4"            "readr"         "tibble"
## [30,] "scales"        "jsonlite"        "lubridate"     "dplyr"
## [31,] "plyr"          "robustbase"      "igraph"        "lattice"
## [32,] "igraph"        "plotmo"          "reshape2"      "plyr"
## [33,] "reshape2"      "quantreg"        "withr"         "R6"
## [34,] "lubridate"     "effects"         "gridExtra"     "vctrs"
## [35,] "doParallel"    "rgl"             "foreach"       "reshape"
## [36,] "grid"          "nnet"            "cli"           "glue"
## [37,] "sp"            "nlme"            "tidyselect"    "digest"
## [38,] "gridExtra"     "sp"              "plyr"          "tinytest"
## [39,] "readr"         "doParallel"      "xml2"          "RUnit"
## [40,] "spelling"      "surveillance"    "sp"            "httr"
## [41,] "lattice"       "mice"            "plotly"        "Matrix"
## [42,] "glue"          "Matrix"          "grid"          "xml2"
## [43,] "RColorBrewer"  "data.tree"       "survival"      "shiny"
## [44,] "sf"            "car"             "lme4"          "curl"
## [45,] "xml2"          "spelling"        "vctrs"         "rstudioapi
## [46,] "raster"        "metafor"         "lifecycle"     "cli"
## [47,] "zoo"           "marginaleffects" "broom"         "yaml"
## [48,] "markdown"      "gtools"          "mvtnorm"       "survival"
## [49,] "R6"            "partykit"        "RcppArmadillo" "markdown"
## [50,] "glmnet"        "hunspell"        "doParallel"    "htmlwidget
```

```
table_aut
```

```
##       In-degree              Betweenness          Eigenvector
##  [1,] "Hadley Wickham"       "R Core"             "RStudio"
```

```
##  [2,] "RStudio"              "Hadley Wickham"     "Hadley Wickham"
##  [3,] "Dirk Eddelbuettel"    "Dirk Eddelbuettel"  "Jim Hester"
##  [4,] "Ben Bolker"           "Martin Maechler"    "JJ Allaire"
##  [5,] "R Core"               "RStudio"            "Winston Chang"
##  [6,] "Martin Maechler"      "Ben Bolker"         "Yihui Xie"
##  [7,] "Yihui Xie"            "Kurt Hornik"        "Joe Cheng"
##  [8,] "Brian Ripley"         "Brian Ripley"       "Max Kuhn"
##  [9,] "Michael Friendly"     "Roger Bivand"       "Gábor Csárdi"
## [10,] "Jim Hester"           "Achim Zeileis"      "Lionel Henry"
## [11,] "Roger Bivand"         "Scott Chamberlain"  "Kirill Müller"
## [12,] "JJ Allaire"           "Jeroen Ooms"        "Kevin Ushey"
## [13,] "Henrik Bengtsson"     "Yihui Xie"          "Christophe Dervi
## [14,] "Bill Venables"        "Zhian N Kamvar"     "Barret Schloerke
## [15,] "Kevin Ushey"          "Michael Friendly"   "Jennifer Bryan"
## [16,] "Achim Zeileis"        "John Muschelli"     "Jeroen Ooms"
## [17,] "Kurt Hornik"          "Rob Hyndman"        "Carson Sievert"
## [18,] "Max Kuhn"             "Tyler Rinker"       "Javier Luraschi"
## [19,] "Romain Francois"      "Bill Denney"        "Romain Francois"
## [20,] "Duncan Murdoch"       "John Wiseman"       "Daniel Falbel"
## [21,] "Zhian N Kamvar"       "Thomas Lumley"      "PBC"
## [22,] "David Robinson"       "Jim Hester"         "R Core"
## [23,] "Hao Zhu"              "Sahir Bhatnagar"    "Henrik Bengtsson
## [24,] "Michal Bojanowski"    "Henrik Bengtsson"   "Ben Bolker"
## [25,] "Joe Cheng"            "Toby Hocking"       "David Robinson"
## [26,] "Vilmantas Gegzna"     "Carl Boettiger"     "Michal Bojanowsk
## [27,] "Christophe Dervieux"  "Kevin Ushey"        "Thomas Lin Peder
## [28,] "Bill Denney"          "Kirill Müller"      "Davis Vaughan"
## [29,] "Adrian Baddeley"      "Bob Rudis"          "JooYoung Seo"
## [30,] "Jeroen Ooms"          "Cleve Moler"        "Atsushi Yasumoto
## [31,] "Hong Ooi"             "JJ Allaire"         "Yixuan Qiu"
## [32,] "Noam Ross"            "Noam Ross"          "Joseph Larmarang
## [33,] "David Hugh-Jones"     "Bill Venables"      "Dirk Eddelbuette
## [34,] "Joseph Larmarange"    "Max Kuhn"           "Malcolm Barrett"
## [35,] "John Muschelli"       "Ben Goodrich"       "Noam Ross"
## [36,] "Jonah Gabry"          "Jonah Gabry"        "Jeff Allen"

## [37,] "Malcolm Barrett"      "Christophe Dutang"  "Hao Zhu"
## [38,] "Greg Snow"            "Torsten Hothorn"    "Michael Friendly
## [39,] "Jim Lemon"            "Winston Chang"      "Jenny Bryan"
## [40,] "Alessandro Gasparini" "Kosuke Imai"        "Google Inc"
## [41,] "Eduard Szoecs"        "Apache Foundation"  "David Hugh-Jones
## [42,] "Tal Galili"           "Arni Magnusson"     "jQuery Foundatio
## [43,] "Jenny Bryan"          "Romain Francois"    "Roger Bivand"
## [44,] "Torsten Hothorn"      "Joe Cheng"          "Hiroaki Yutani"
## [45,] "Matthieu Stigler"     "Duncan Murdoch"     "Martin Maechler"
## [46,] "Garrick Aden-Buie"    "Steven G Johnson"   "Alex Hayes"
## [47,] "Dieter Menne"         "Yuan Tang"          "Mango Solutions"
## [48,] "Frank E Harrell Jr"   "Jacob Bien"         "Richard Iannone"
```

```
## [49,] "Rolf Turner"          "Lampros Mouselimis" "Simon Couch"
## [50,] "Michael Chirico"      "Yi Yang"            "Frederik Aust"
##       PageRank
##  [1,] "RStudio"
##  [2,] "Hadley Wickham"
##  [3,] "R Core"
##  [4,] "Martin Maechler"
##  [5,] "Ben Bolker"
##  [6,] "Dirk Eddelbuettel"
##  [7,] "Yihui Xie"
##  [8,] "Achim Zeileis"
##  [9,] "JJ Allaire"
## [10,] "Kurt Hornik"
## [11,] "Brian Ripley"
## [12,] "Roger Bivand"
## [13,] "Jim Hester"
## [14,] "Jeroen Ooms"
## [15,] "Scott Chamberlain"
## [16,] "Michael Friendly"
## [17,] "Zhian N Kamvar"
## [18,] "Winston Chang"
## [19,] "Joe Cheng"
## [20,] "Henrik Bengtsson"
## [21,] "Bob Rudis"
## [22,] "Kevin Ushey"
## [23,] "Duncan Murdoch"
## [24,] "Max Kuhn"
## [25,] "Bill Venables"
## [26,] "John Muschelli"
## [27,] "Gábor Csárdi"
## [28,] "Rob Hyndman"
## [29,] "Torsten Hothorn"
## [30,] "Kirill Müller"
## [31,] "Noam Ross"
## [32,] "Romain Francois"

## [33,] "Christophe Dervieux"
## [34,] "Carl Boettiger"
## [35,] "David Robinson"
## [36,] "Jonah Gabry"
## [37,] "Maëlle Salmon"
## [38,] "Michael Sumner"
## [39,] "Barret Schloerke"
## [40,] "Stéphane Laurent"
## [41,] "Bill Denney"
## [42,] "Edzer Pebesma"
## [43,] "Hao Zhu"
## [44,] "Indrajeet Patil"
```

```
## [45,] "Michel Lang"
## [46,] "Michal Bojanowski"
## [47,] "John Fox"
## [48,] "Google Inc"
## [49,] "Arni Magnusson"
## [50,] "Uwe Ligges"
```

4.3 Sensitivity Analysis

weight: 321

```
pkg_net_type = E(pkg_graph)$type
pkg_net_type_num321 = pkg_net_type
pkg_type = c("depends", "imports", "suggests", "linking_to", "enhanc
        es")
pkg_type_num321 = c(3,2,1,1,1)
for (typei in 1:length(pkg_type)){
    loc = which(pkg_net_type==pkg_type[typei])
    pkg_net_type_num321[loc] = pkg_type_num321[typei]
}
pkg_graph_w321 = pkg_graph %>% set_edge_attr("weight", value = pkg_n
        et_type_num321)

pkgw_321_betw = betweenness(pkg_graph)
pkgw_321_dg = degree(pkg_graph_w321, mode = c("in"))
pkgw_321_pgRk = page_rank(pkg_graph_w321, weights = E(pkg_graph_w32
        1)$weight)
pkgw_321_eigen = eigen_centrality(pkg_graph_w321, weights = E(pkg_gr
        aph_w321)$weight)
```

no weight

```
pkgw_1_betw = betweenness(pkg_graph)
pkgw_1_dg = degree(pkg_graph, mode = c("in"))
pkgw_1_pgRk = page_rank(pkg_graph)
pkgw_1_eigen = eigen_centrality(pkg_graph)
```

```
score_54321 = cbind(pkgw_dg, pkgw_betw, pkgw_eigen[["vector"]], pkgw
        _pgRk[["vector"]])
score_321 = cbind(pkgw_321_dg, pkgw_321_betw, pkgw_321_eigen[["vecto
        r"]], pkgw_321_pgRk[["vector"]])
```

```
score_1 = cbind(pkgw_1_dg, pkgw_1_betw, pkgw_1_eigen[["vector"]], pk
        gw_1_pgRk[["vector"]])
for (i in 1:ncol(score_1)){
    cat(cor(score_54321[,i], score_321[,i]))
    cat('\t')
    cat(cor(score_54321[,i], score_1[,i]))
    cat('\n')
}
```

```
## 1    1
## 1    1
## 0.9662078    0.9805951
## 0.9926748    0.9941666
```

4.4 Select Important Packages and Authors

Packages

```
imp_pkg = NULL
for(i in 1:nrow(table_pkg)){
  for(j in 1:ncol(table_pkg)){
    pkg_temp = as.character(table_pkg[i,j])
    loc = which(table_pkg==pkg_temp)
    if(length(loc)>1){
      if(length(which(imp_pkg==pkg_temp))==0){
        imp_pkg = c(imp_pkg, pkg_temp)
      }
    }
  }
}


imp_pkg2 = array(NA, dim=c(length(imp_pkg),3))
imp_pkg2[,1] = imp_pkg
colnames(imp_pkg2) = c('pkg','title','mt')
for(i in 1:length(imp_pkg)){
  imp_pkg2[i,'pkg'] = imp_pkg[i]
  loc = which(cran[,'Package']==imp_pkg[i])
  temp = try(cran[loc,'Title'])
  if(length(temp)>0){
    imp_pkg2[i,'title'] = temp
  }
  temp = try(cran[loc,'Maintainer'])
```

```
  if(length(temp)>0){
    imp_pkg2[i,'mt'] = split_mt(temp)$name
  }
}
write.csv.utf8.BOM(imp_pkg2,'network/imp_pkg2_54321.csv')
imp_pkg2
```

```
##       pkg
##  [1,] "knitr"
##  [2,] "ggplot2"
##  [3,] "testthat"
##  [4,] "rmarkdown"
##  [5,] "utils"
##  [6,] "broom"
##  [7,] "methods"
##  [8,] "stats"
##  [9,] "dplyr"
## [10,] "Rcpp"
## [11,] "shiny"
## [12,] "covr"
## [13,] "graphics"
## [14,] "survival"
## [15,] "rlang"
## [16,] "stringr"
## [17,] "MASS"
## [18,] "magrittr"
## [19,] "sf"
## [20,] "tibble"
## [21,] "tidyr"
## [22,] "grDevices"
## [23,] "purrr"
## [24,] "parallel"
## [25,] "data.table"
## [26,] "withr"
## [27,] "Matrix"
## [28,] "jsonlite"
## [29,] "enrichwith"
## [30,] "cops"
## [31,] "RcppArmadillo"
## [32,] "scales"
## [33,] "zoo"
## [34,] "httr"
## [35,] "glue"
## [36,] "mvtnorm"
## [37,] "foreach"
```

```
## [38,] "lme4"
## [39,] "readr"
## [40,] "lubridate"
## [41,] "plyr"
## [42,] "igraph"
## [43,] "lattice"
## [44,] "reshape2"
## [45,] "R6"
## [46,] "gridExtra"
## [47,] "vctrs"
## [48,] "doParallel"
## [49,] "grid"
## [50,] "cli"
## [51,] "sp"
## [52,] "xml2"
## [53,] "spelling"
## [54,] "markdown"
##       title
##  [1,] "A General-Purpose Package for Dynamic Report Generation in R
##  [2,] "Create Elegant Data Visualisations Using the Grammar of Grap
##  [3,] "Unit Testing for R"
##  [4,] "Dynamic Documents for R"
##  [5,] NA
##  [6,] "Convert Statistical Objects into Tidy Tibbles"
##  [7,] NA
##  [8,] NA
##  [9,] "A Grammar of Data Manipulation"
## [10,] "Seamless R and C++ Integration"
## [11,] "Web Application Framework for R"
## [12,] "Test Coverage for Packages"
## [13,] NA
## [14,] "Survival Analysis"
## [15,] "Functions for Base Types and Core R and 'Tidyverse' Features
## [16,] "Simple, Consistent Wrappers for Common String Operations"
## [17,] "Support Functions and Datasets for Venables and Ripley's MAS
## [18,] "A Forward-Pipe Operator for R"
## [19,] "Simple Features for R"
## [20,] "Simple Data Frames"
## [21,] "Tidy Messy Data"
## [22,] NA
## [23,] "Functional Programming Tools"
## [24,] NA
## [25,] "Extension of `data.frame`"
## [26,] "Run Code 'With' Temporarily Modified Global State"
## [27,] "Sparse and Dense Matrix Classes and Methods"
## [28,] "A Simple and Robust JSON Parser and Generator for R"
## [29,] "Methods to Enrich R Objects with Extra Components"
```

```
## [30,] "Cluster Optimized Proximity Scaling"
## [31,] "'Rcpp' Integration for the 'Armadillo' Templated Linear Alge
## [32,] "Scale Functions for Visualization"
## [33,] "S3 Infrastructure for Regular and Irregular Time Series (Z's
## [34,] "Tools for Working with URLs and HTTP"
## [35,] "Interpreted String Literals"
## [36,] "Multivariate Normal and t Distributions"
## [37,] "Provides Foreach Looping Construct"
## [38,] "Linear Mixed-Effects Models using 'Eigen' and S4"
## [39,] "Read Rectangular Text Data"
## [40,] "Make Dealing with Dates a Little Easier"
## [41,] "Tools for Splitting, Applying and Combining Data"
## [42,] "Network Analysis and Visualization"
## [43,] "Trellis Graphics for R"
## [44,] "Flexibly Reshape Data: A Reboot of the Reshape Package"
## [45,] "Encapsulated Classes with Reference Semantics"
## [46,] "Miscellaneous Functions for \"Grid\" Graphics"
## [47,] "Vector Helpers"
## [48,] "Foreach Parallel Adaptor for the 'parallel' Package"
## [49,] NA
## [50,] "Helpers for Developing Command Line Interfaces"
## [51,] "Classes and Methods for Spatial Data"
## [52,] "Parse XML"
## [53,] "Tools for Spell Checking in R"
## [54,] "Render Markdown with 'commonmark'"
##       mt
##  [1,] "Yihui Xie "
##  [2,] "Thomas Lin Pedersen "
##  [3,] "Hadley Wickham "
##  [4,] "Yihui Xie "
##  [5,] NA
##  [6,] "Simon Couch "
##  [7,] NA
##  [8,] NA
##  [9,] "Hadley Wickham "
## [10,] "Dirk Eddelbuettel "
## [11,] "Winston Chang "
## [12,] "Jim Hester "
## [13,] NA
## [14,] "Terry M Therneau "
## [15,] "Lionel Henry "
## [16,] "Hadley Wickham "
## [17,] "Brian Ripley "
## [18,] "Lionel Henry "
## [19,] "Edzer Pebesma "
## [20,] "Kirill Müller "
## [21,] "Hadley Wickham "
```

```
## [22,] NA
## [23,] "Lionel Henry "
## [24,] NA
## [25,] "Matt Dowle "
## [26,] "Lionel Henry "
## [27,] "Martin Maechler "
## [28,] "Jeroen Ooms "
## [29,] "Ioannis Kosmidis "
## [30,] "Thomas Rusch "
## [31,] "Dirk Eddelbuettel "
## [32,] "Hadley Wickham "
## [33,] "Achim Zeileis "
## [34,] "Hadley Wickham "
## [35,] "Jennifer Bryan "
## [36,] "Torsten Hothorn "
## [37,] "Folashade Daniel "
## [38,] "Ben Bolker "
## [39,] "Jennifer Bryan "
## [40,] "Vitalie Spinu "
## [41,] "Hadley Wickham "
## [42,] "Tamás Nepusz "
## [43,] "Deepayan Sarkar "
## [44,] "Hadley Wickham "
## [45,] "Winston Chang "
## [46,] "Baptiste Auguie "
## [47,] "Lionel Henry "
## [48,] "Folashade Daniel "
## [49,] NA
## [50,] "Gábor Csárdi "
## [51,] "Edzer Pebesma "
## [52,] "Hadley Wickham "
## [53,] "Jeroen Ooms "
## [54,] "Yihui Xie "
```

Authors

```
## important authors (identified by at least two indexes)
imp_aut = NULL
for(i in 1:nrow(table_aut)){
  for(j in 1:ncol(table_aut)){
    aut_temp = as.character(table_aut[i,j])
    loc = which(table_aut==aut_temp)
    if(length(loc)>1){
      if(length(which(imp_aut==aut_temp))==0){
        imp_aut = c(imp_aut, aut_temp)
      }
```

```
    }
  }
}

imp_aut
```

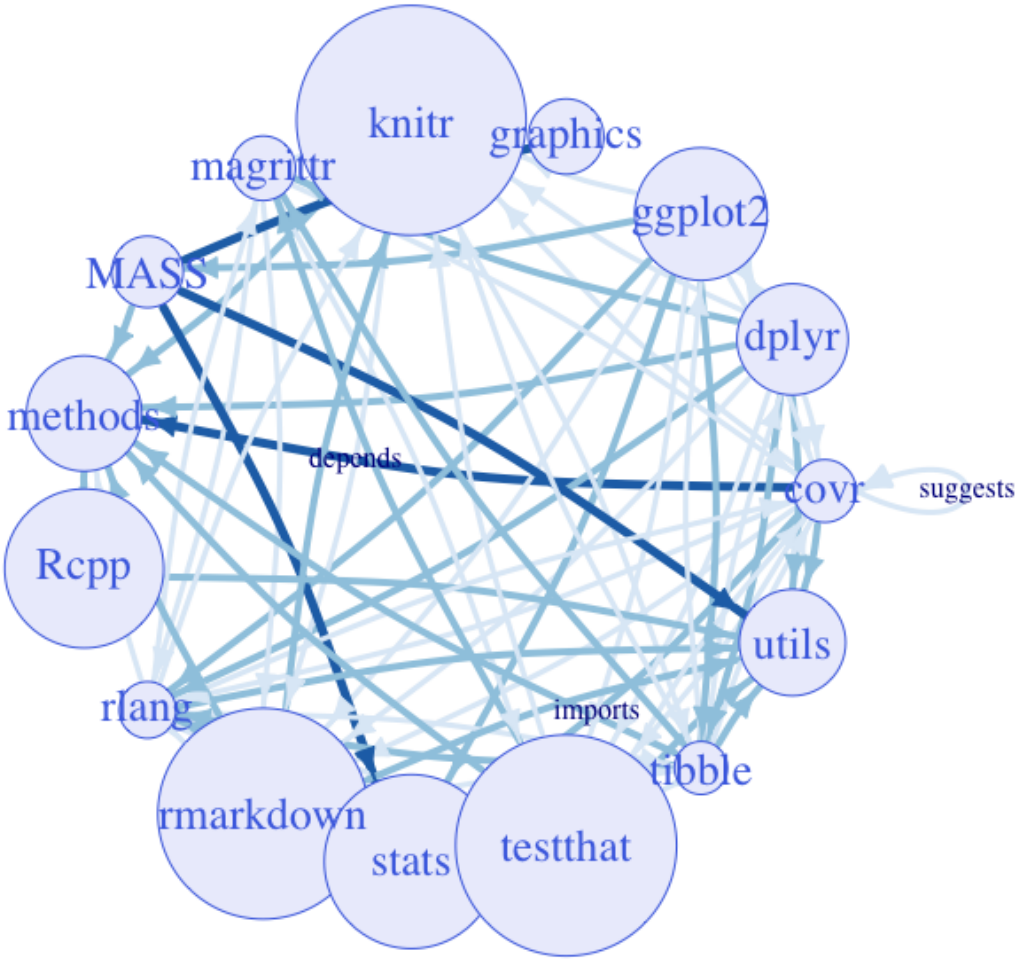
```
##  [1] "Hadley Wickham"      "R Core"              "RStudio"
##  [4] "Dirk Eddelbuettel"   "Jim Hester"          "Ben Bolker"
##  [7] "Martin Maechler"     "JJ Allaire"          "Winston Chang"
## [10] "Yihui Xie"           "Kurt Hornik"         "Joe Cheng"
## [13] "Brian Ripley"        "Max Kuhn"            "Achim Zeileis"
## [16] "Michael Friendly"    "Roger Bivand"        "Gábor Csárdi"
## [19] "Scott Chamberlain"   "Kirill Müller"       "Jeroen Ooms"
## [22] "Kevin Ushey"         "Henrik Bengtsson"    "Christophe Dervie
## [25] "Bill Venables"       "Zhian N Kamvar"      "Barret Schloerke"
## [28] "John Muschelli"      "Rob Hyndman"         "Romain Francois"
## [31] "Bill Denney"         "Duncan Murdoch"      "Bob Rudis"
## [34] "David Robinson"      "Hao Zhu"             "Michal Bojanowski
## [37] "Carl Boettiger"      "Torsten Hothorn"     "Noam Ross"
## [40] "Joseph Larmarange"   "David Hugh-Jones"    "Malcolm Barrett"
## [43] "Jonah Gabry"         "Jenny Bryan"         "Google Inc"
## [46] "Arni Magnusson"
```

4.5 Visualization

package sub-graph

```
library(wordcloud)
colors <- colorspace::diverge_hcl(10, c = 100, l = c(50, 100), power
        = 1)
pkg_graph_w_v = pkg_graph_w %>% set_vertex_attr("indegree", value =
         pkgw_dg)
pkg_sub_graph = subgraph(pkg_graph_w_v,names(pkg_rank_dg)[1:15])

pkg_net_type = E(pkg_sub_graph)$type
pkg_net_type_color = pkg_net_type
pkg_type = c("depends", "imports", "suggests", "linking_to", "enhanc
        es")
pkg_type_color = c(brewer.pal(8,"Blues")[7],brewer.pal(8,"Blues")[4
        ],brewer.pal(8,"Blues")[2])


for (typei in 1:length(pkg_type)){
```

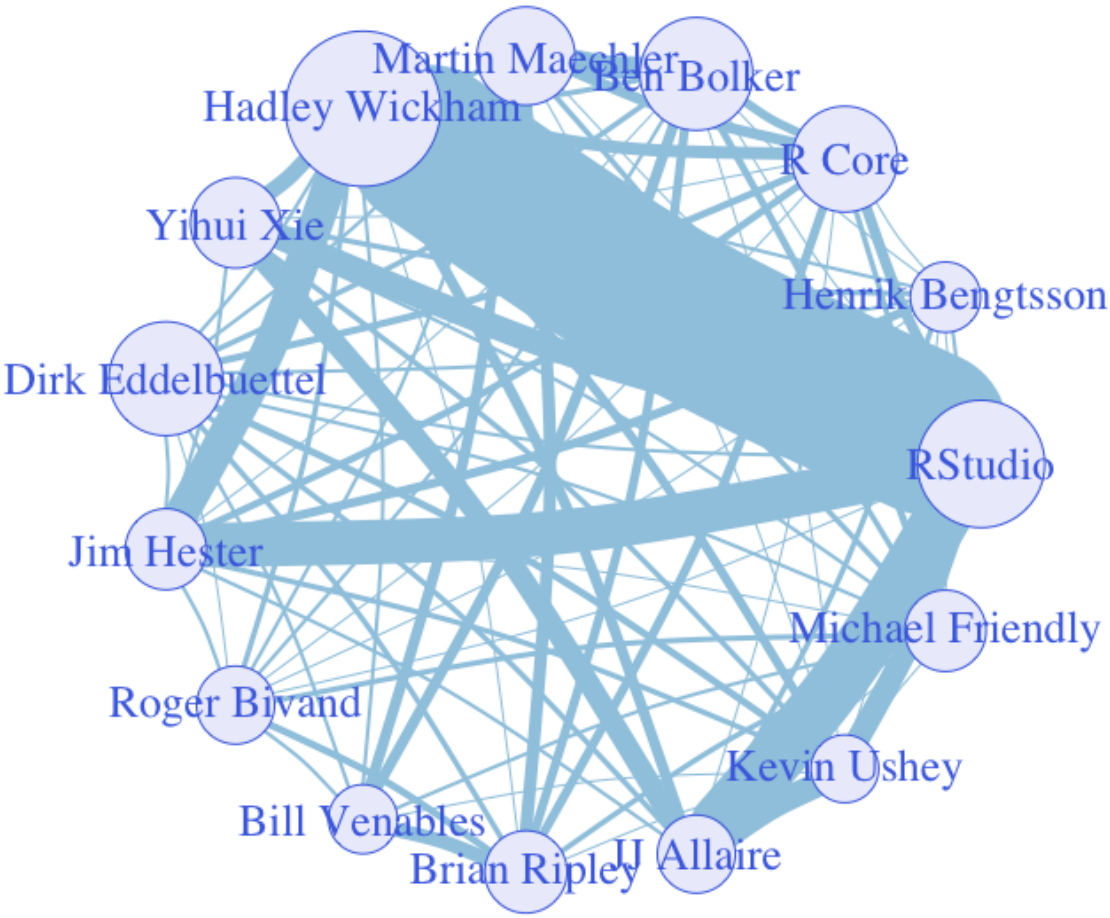
```
    loc = which(pkg_net_type==pkg_type[typei])
    pkg_net_type_color[loc] = pkg_type_color[typei]
}
pkg_net_type_label = rep(NA,length(pkg_net_type))
for(typei in 1:length(pkg_type)){ # select five labels as example
    loc = which(pkg_net_type==pkg_type[typei])[1]
    pkg_net_type_label[loc] = pkg_type[typei]
}


pkg_sub_graph = pkg_sub_graph %>%
    set_edge_attr("color", value = pkg_net_type_color) %>%
    set_edge_attr("label", value = pkg_net_type_label)


plot.igraph(pkg_sub_graph,
    edge.color = E(pkg_sub_graph)$color,
    edge.width = as.numeric(E(pkg_sub_graph)$weight),
    edge.label = E(pkg_sub_graph)$label,
    edge.arrow.size = 1,
    edge.curved = .1,
    vertex.color = colors[5],
    vertex.frame.color=colors[1],
    vertex.label.color=colors[1],
    layout=layout.circle,
    vertex.size = (V(pkg_sub_graph)$indegree)/120,
    vertex.label.cex = 1.6)
```



author sub-graph

```
colors <- colorspace::diverge_hcl(10, c = 100, l = c(50, 100), power
        = 1)
aut_graph_w_v = aut_graph_w %>% set_vertex_attr("degree", value = au
        tw_dg)
aut_sub_graph = subgraph(aut_graph_w_v,names(aut_rank_dg)[1:15]) #im
        p_aut2[,1]
aut_type_color = c(brewer.pal(8,"Blues")[4])


plot.igraph(aut_sub_graph,
    edge.color = aut_type_color,
    edge.width = E(aut_sub_graph)$weight,
```

```
    edge.arrow.size = 1,
    edge.curved = .1,
    vertex.color = colors[5],
    vertex.frame.color=colors[1],
    vertex.label.color=colors[1],
    layout=layout.circle,
    vertex.size = (V(aut_sub_graph)$degree)/20,
    vertex.label.cex = 1.6)
```

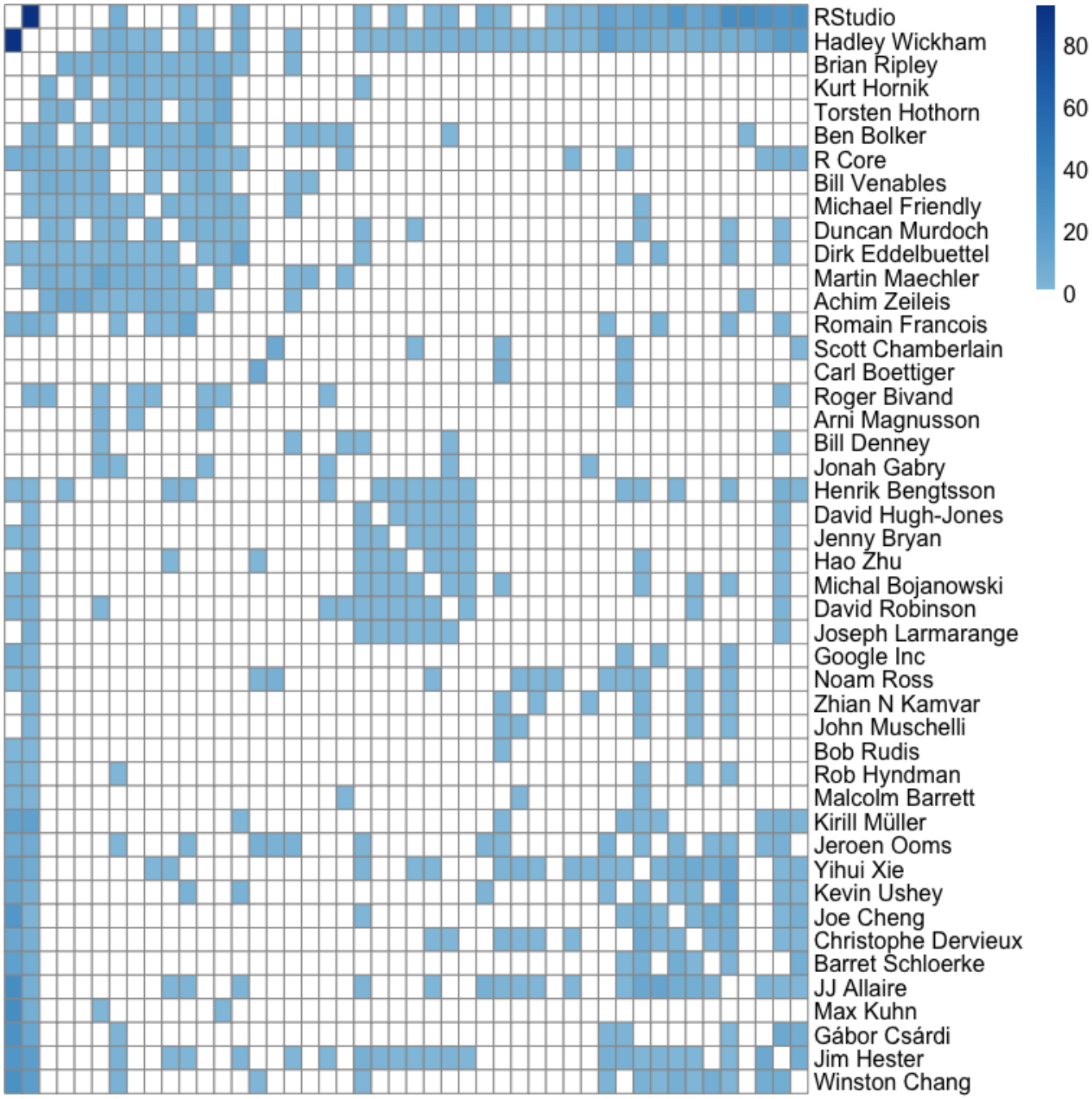


heatmap of author collaboration

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(pheatmap)
loc=NULL
for(i in 1:length(imp_aut)){
  loc = c(loc, which(colnames(aut_adj_matrix)==imp_aut[i]))
}
aut_corsub_matrix = as.matrix(aut_adj_matrix[loc,loc])


pheatmap(aut_corsub_matrix,col=c("white",colorRampPalette(brewer.pal
         (8, "Blues"))(170)[80:170]),
    angle_col="45",
    show_colnames = F,
    fontsize=12,
    treeheight_row = 0, treeheight_col = 0)
```



4.6 Correlation among importance indexes and downloads

```
library(cranlogs)
pkg_down_year = array(NA, dim = c(nrow(cran),2))
colnames(pkg_down_year) = c("pkg", "downloads")
pkg_down_year[,"pkg"] = cran[,"Package"]

for(pkgi in 1:nrow(pkg_down_year)){ #6402
    cat(paste0(pkgi,"-",pkg_down_year[pkgi, "pkg"],"\n"))
```

```
    pkg_down_year[pkgi, "downloads"] = sum(cran_downloads(pkg_down_y
        ear[pkgi, "pkg"], from = "2021-11-01", to = "2022-10-31")$c
        ount)
}

pkg_down_year = sort(pkg_down_year, decreasing = TRUE)
write.csv(pkg_down_year,"pkg_down_year.csv",row.names=F)
```

```
pkg_down_year = read.csv('pkg_down_year.csv')
pkg_down_year_rank = as.numeric(pkg_down_year[,2])
names(pkg_down_year_rank) = pkg_down_year[,1]

pkg_all_index = cbind(pkgw_dg, pkgw_betw, pkgw_eigen[["vector"]], pk
        gw_pgRk[["vector"]])
colnames(pkg_all_index) = c('dg','bet','eigen','pg')
pkg_down_rank_year_n = pkgw_dg
no_down = NULL
for(i in 1:length(pkg_down_rank_year_n)){
  loc = which(names(pkg_down_year_rank) == names(pkgw_dg)[i])
  if(length(loc)!=0){
    pkg_down_rank_year_n[i] = pkg_down_year_rank[loc]
  }else{
    no_down = c(no_down, names(pkgw_dg)[i])
  }
}

for (i in 1:4){
    cat(cor(pkg_down_rank_year_n, pkg_all_index[,i]))
    cat('\n')
}
```

```
## 0.4217525
## 0.4065768
## 0.4335971
## 0.3489176
```

5 Bipartite Network

5.1 Build bipartite network and set edges (weights)

```
basepkg = c("base", "boot", "class", "cluster", "codetools", "compil
        er", "datasets",
    "foreign", "graphics", "grDevices", "grid", "KernSmooth", "latti
        ce", "MASS",
    "Matrix", "methods", "mgcv", "nlme", "nnet", "parallel", "rpart"
        , "spatial",
    "splines", "stats", "stats4", "survival", "tcltk", "tools", "uti
        ls")

bi_matrix_w = array(0, dim = c(length(V(aut_graph)$name), length(V(p
        kg_graph_w)$name)))
colnames(bi_matrix_w) = V(pkg_graph_w)$name
rownames(bi_matrix_w) = V(aut_graph)$name

for (auti in 1:nrow(bi_matrix_w)){
    aut_pkg = V(aut_graph)$package[[auti]]
    aut_pkg_num = length(aut_pkg)
    for (aut_pkgi in 1:aut_pkg_num){
        pkgi = which(colnames(bi_matrix_w)==aut_pkg[aut_pkgi])
        #if(aut_pkg[aut_pkgi] == colnames(bi_matrix_w)[pkgi]){
        bi_matrix_w[auti, pkgi] = 1
        if(V(aut_graph)$name[auti] == V(pkg_graph_w)$maintainer[pkg
        i]){
            bi_matrix_w[auti, pkgi] = 3
        }
    }
}

# base r packages
locRcore = which(rownames(bi_matrix_w) == "R Core")
for (pkgi in 1:length(basepkg)){
    locpkg = which(colnames(bi_matrix_w) == basepkg[pkgi])
    if(is.na(V(pkg_graph_w)$maintainer[locpkg])){
            bi_matrix_w[locRcore, locpkg] = 3
    }
}
bi_matrix_w0 = bi_matrix_w


loc1 = which(rownames(bi_matrix_w)=="Kirill Muller")
loc2 = which(rownames(bi_matrix_w)=="Kirill Müller")
loc11 = which(bi_matrix_w[loc1,]>0)
loc21 = which(bi_matrix_w[loc2,]>0)
bi_matrix_w[loc2,loc11]=bi_matrix_w[loc2,loc11]+bi_matrix_w[loc1,loc
        11]
bi_matrix_w = bi_matrix_w[-loc1, ]

loc1 = which(rownames(bi_matrix_w)=="Gabor Csardi")
```

```
loc2 = which(rownames(bi_matrix_w)=="Gábor Csárdi")
loc11 = which(bi_matrix_w[loc1,]>0)
loc21 = which(bi_matrix_w[loc2,]>0)
bi_matrix_w[loc2,loc11]=bi_matrix_w[loc2,loc11]+bi_matrix_w[loc1,loc
        11]
bi_matrix_w = bi_matrix_w[-loc1, ]

bi_graph_w = graph.incidence(bi_matrix_w, weighted = T) #graph_from_
        incidence_matrix
```

5.2 Visulization

```
loc_sub1 = loc_sub2 = NULL
for (i in 1:15){
    loc_sub1 = c(loc_sub1, which(V(aut_graph)$name==names(aut_rank_d
        g)[i]))
    loc_sub2 = c(loc_sub2, which(V(pkg_graph_w)$name==names(pkg_rank
        _dg)[i]))

}
bi_matrix_sub = bi_matrix_w[loc_sub1, loc_sub2]

bi_graph_w_sub = graph.incidence(bi_matrix_sub, weighted = T)  #subg
        raph(bi_graph_w,c(names(pkg_rank_dg)[1:15],names(aut_rank_d
        g)[1:15]))


# all influential pkg and author
loc_sub1 = loc_sub2 = NULL
for (i in 1:nrow(imp_aut2)){
    loc_sub1 = c(loc_sub1, which(V(aut_graph)$name==imp_aut2[i,1]))
}
for (i in 1:nrow(imp_pkg2)){
    loc_sub2 = c(loc_sub2, which(V(pkg_graph_w)$name==imp_pkg2[i,1
        ]))
}
bi_matrix_sub = bi_matrix_w[loc_sub1, loc_sub2]

bi_graph_w_sub = graph.incidence(bi_matrix_sub, weighted = T)


V(bi_graph_w_sub)$color <- V(bi_graph_w_sub)$type
V(bi_graph_w_sub)$color=gsub("FALSE","#FAEBD7",V(bi_graph_w_sub)$col
        or)
V(bi_graph_w_sub)$color=gsub("TRUE",colors[5],V(bi_graph_w_sub)$colo
        r)
```

```
#plot(igraph::simplify(bi_graph_w_sub), edge.color="#E9967A", edge.w
        idth=E(bi_graph_w_sub)$weight, layout=LO, vertex.size = 6,
         vertex.label.cex = 1.4,vertex.label.color = "black", verte
        x.label.family="Arial")
#too large to be presented in this output
```

5.3 Centrality

```
library(bipartite)
library(birankr)
```

```
#bi_matrix_w
bi_adj = as_adjacency_matrix(bi_graph_w) #as_edgelist
bi_edge = as_edgelist(bi_graph_w)
bi_e_weight = E(bi_graph_w)$weight


bi_edge_dt_w = data.table(cbind(bi_edge,bi_e_weight))



bi_cohits_w_0 = br_cohits(bi_edge_dt_w, weight_name = "bi_e_weight")
bi_cohits_w = bi_cohits_w_0[,'rank']
names(bi_cohits_w) = bi_cohits_w_0[,1]
head(sort(bi_cohits_w,T))
```

```
## Dirk Eddelbuettel    Hadley Wickham           RStudio  Stéphane Lau
##       0.001799382       0.001761922       0.001732088       0.00157
##      Gábor Csárdi Scott Chamberlain
##       0.001487436       0.001469630
```

```
bi_birank_w_0 = br_birank(bi_edge_dt_w, weight_name = "bi_e_weight")
bi_birank_w = bi_birank_w_0[,'rank']
names(bi_birank_w) = bi_birank_w_0[,1]
head(sort(bi_birank_w,T))
```

```
## Dirk Eddelbuettel    Hadley Wickham           RStudio  Stéphane Lau
##      0.0002311202      0.0002293970      0.0002287313      0.000214
## Scott Chamberlain      Gábor Csárdi
##      0.0002120412      0.0002093672
```

Relationship between cohits and birank

```
cor(bi_cohits_w, bi_birank_w)
```

```
## [1] 0.8596433
```

```
bi_cohits_w_50 = sort(bi_cohits_w,T)[1:50]
bi_birank_w_50 = sort(bi_birank_w,T)[1:50]
```

Relationships between birank and other indexes

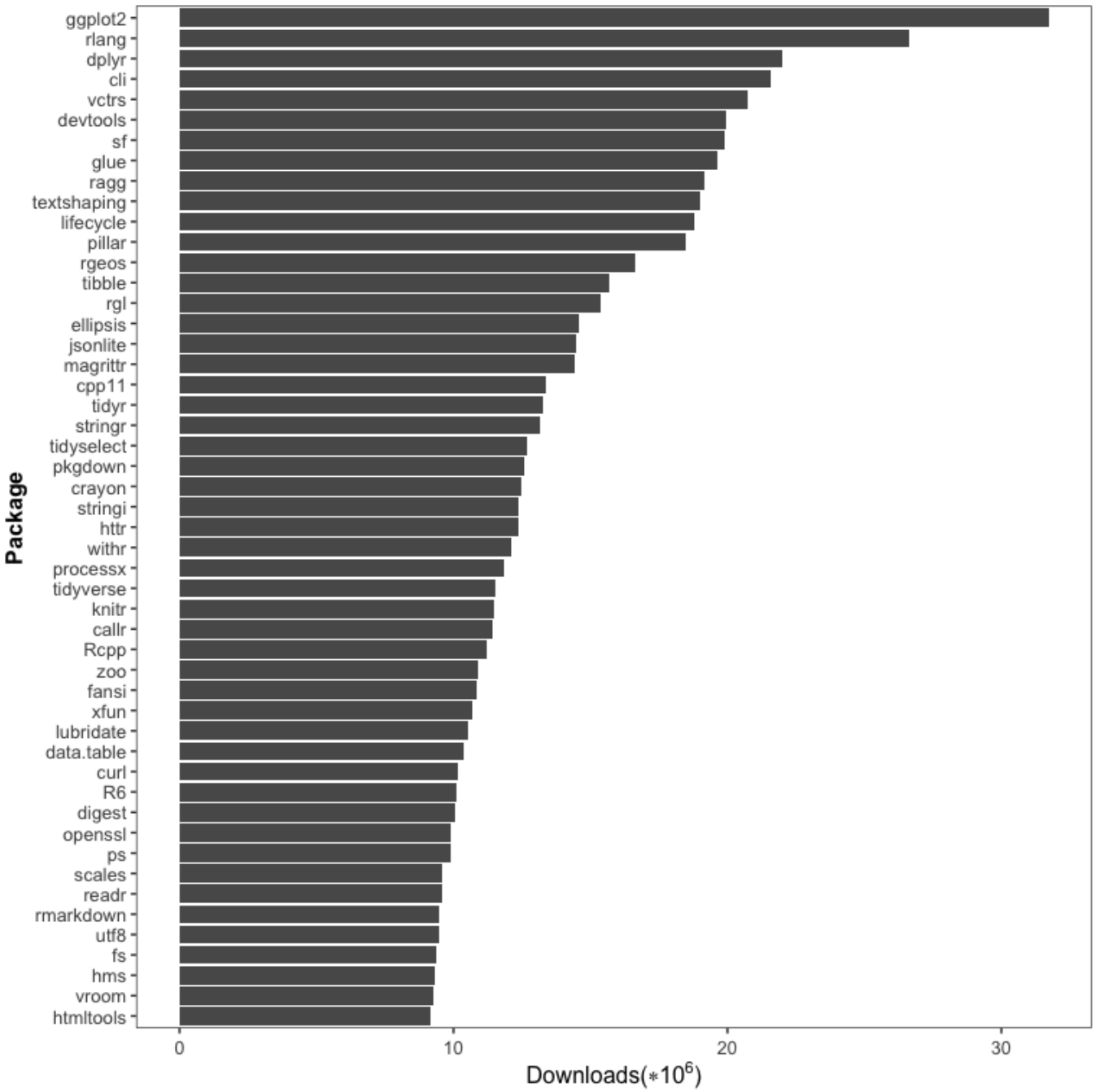
```
aut_birank = autw_dg
for(i in 1:length(aut_birank)){
    loc = which(names(bi_birank_w) == names(aut_birank)[i])
    aut_birank[i] = bi_birank_w[loc]
}
aut_allindex = cbind(autw_dg, autw_betw, autw_eigen[["vector"]], aut
        w_pgRk[["vector"]], aut_birank)
colnames(aut_allindex) = c('dgree', 'betw', 'eigen', 'pgRk',  'BiRan
        k')

for (i in 1:4){
    cat(colnames(aut_allindex)[i])
    cat('\t')
    cat(cor(aut_allindex[,i], aut_allindex[,5]))
    cat('\n')
}
```

```
## dgree    0.178544
## betw 0.3635495
## eigen    0.2471052
## pgRk 0.4269204
```

Visualization

```
pkg_download = sort(pkg_down_year_rank, decreasing = TRUE)
downloads_sorted = cbind(names(pkg_download), as.numeric(pkg_downloa
        d) )
```

```r
library(latex2exp)

downloads_50 = tibble(package = downloads_sorted[1:50,1],downloads =
        as.numeric(downloads_sorted[1:50,2])/1000000)
downloads_50_plot = downloads_50 %>%
    mutate(package = reorder(package, downloads)) %>%
    ggplot(aes(downloads,package)) +
    labs(y = "Package", x = TeX("Downloads(*$10^6$)")) +
    theme(panel.grid=element_blank())  +
    theme(axis.text.x = element_text(angle = 0, hjust = 0.5, vjust =
        0.5))+
    theme(axis.title.x = element_text(face = 'bold'),  axis.title =
        element_text(face = 'bold'), text=element_text(size=12))+
    geom_col()
downloads_50_plot
```


Package
ggplot2
rlang
dplyr
cli
vctrs
devtools
sf
glue
ragg
textshaping
lifecycle
pillar
rgeos
tibble
rgl
ellipsis
jsonlite
magrittr
cpp11
tidyr
stringr
tidyselect
pkgdown
crayon
stringi
httr
withr
processx
tidyverse
knitr
callr
Rcpp
zoo
fansi
xfun
lubridate
data.table
curl
R6
digest
openssl
ps
scales
readr
rmarkdown
utf8
fs
hms
vroom
htmltools
0
10
20
30
Downloads(∗10^6)