Corona-Rechtsprechung des Bundesverfassungsgerichts

(BVerfG-Corona-Source)

Compilation Report

Version 2023-02-26

DOI: 10.5281/zenodo.7807021

Titel	Source Code der »Corona-Rechtsprechung des Bundesverfassungsgerichts«
Abkürzung	BVerfG-Corona-Source
Autor	Seán Fobbe
Version	2023-02-26
Download	https://doi.org/10.5281/zenodo.7807021
Lizenz	MIT No Attribution (MIT-0)

Zitiervorschlag

Seán Fobbe (2023). Source Code der »Corona-Rechtsprechung des Bundesverfassungsgerichts« (BVerfG-Corona-Source). Version 2023-02-26. Zenodo. DOI: 10.5281/zenodo.7807021.

Digital Object Identifier (DOI): Concept DOI und Version DOI

Soweit nicht anders angegeben ist die DOI immer eine »Version DOI« und bezieht sich nur auf eine bestimmte Version der Software. Sie verlinkt daher nur Version 2023-02-26. Für das Gesamtkonzept der Software steht eine »Concept DOI« zur Verfügung, die auf der Zenodo-Seite jeder Version unter »Cite all versions?« zu finden ist. Sie lautet 10.5281/zenodo.4459415. Die »Concept DOI« verlinkt immer die aktuellste Version.

Lizenz: MIT No Attribution (MIT-0)

Disclaimer

Dieser Datensatz ist eine private wissenschaftliche Initiative und steht in keiner Verbindung zu Behörden, Gerichten oder anderen amtlichen Stellen der Bundesrepublik Deutschland.

Inhaltsverzeichnis

1 README

1.1 Überblick

Dieser R Code lädt den Corpus der Entscheidungen des Bundesverfassungsgerichts (CE-BVerfG) herunter, untersucht ihn auf mit SARS-CoV-2 assoziiertem Vokabular und speichert relevante Entscheidungen. Es ist die Grundlage für den Datensatz **Corona-Rechtsprechung des Bundesverfassungsgerichts (BVerfG-Corona)**.

Alle mit diesem Skript erstellten Datensätze werden dauerhaft kostenlos und urheberrechtsfrei auf Zenodo, dem wissenschaftlichen Archiv des CERN, veröffentlicht. Alle Versionen sind mit einem persistenten Digital Object Identifier (DOI) versehen. Die neueste Version des Datensatzes ist immer über den Link der Concept DOI erreichbar: https://doi.org/10.5281/zenodo.4459405

1.2 Funktionsweise

Primäre Endprodukte des Skripts (im Ordner 'output') sind folgende ZIP-Archive:

- Alle Corona-relevanten Entscheidungen im PDF-Format
- Alle Corona-relevanten Entscheidungen im TXT-Format
- Alle Analyse-Ergebnisse (Tabellen als CSV, Grafiken als PDF und PNG)
- Der Source Code und alle weiteren Quelldaten

Zusätzlich werden für alle ZIP-Archive kryptographische Signaturen (SHA2-256 und SHA3-512) berechnet und in einer CSV-Datei hinterlegt. Es kann optional ein PDF-Bericht erstellt werden (siehe unter "Kompilierung").

1.3 Systemanforderungen

- Docker
- Docker Compose
- 1 GB Speicherplatz auf Festplatte
- Multi-core CPU empfohlen (8 cores/16 threads für die Referenzdatensätze).

In der Standard-Einstellung wird das Skript vollautomatisch die maximale Anzahl an Rechenkernen/Threads auf dem System zu nutzen. Die Anzahl der verwendeten Kerne kann in der Konfigurationsatei angepasst werden. Wenn die Anzahl Threads auf 1 gesetzt wird, ist die Parallelisierung deaktiviert.

1.4 Anleitung

1.4.1 Schritt 1: Ordner vorbereiten

Kopieren Sie bitte den gesamten Source Code in einen leeren Ordner (!), beispielsweise mit:

```
$ git clone https://github.com/seanfobbe/bverfg-corona.git
```

Verwenden Sie immer einen separaten und *leeren* Ordner für die Kompilierung. Die Skripte löschen innerhalb von bestimmten Unterordnern (`files/`, `temp/`, `analysis` und `output/`) alle Dateien die den Datensatz verunreinigen könnten — aber auch nur dort.

1.4.2 Schritt 2: Docker Image erstellen

Ein Docker Image stellt ein komplettes Betriebssystem mit der gesamten verwendeten Software automatisch zusammen. Nutzen Sie zur Erstellung des Images einfach:

```
$ bash docker-build-image.sh
```

1.4.3 Schritt 3: Datensatz kompilieren

Falls Sie zuvor den Datensatz schon einmal kompiliert haben (ob erfolgreich oder erfolglos), können Sie mit folgendem Befehl alle Arbeitsdaten im Ordner löschen:

```
$ Rscript delete_all_data.R
```

Den vollständigen Datensatz kompilieren Sie mit folgendem Skript:

```
$ bash docker-run-project.sh
```

1.4.4 Ergebnis

Der Datensatz und alle weiteren Ergebnisse sind nun im Ordner `output/` abgelegt.

1.5 Pipeline visualisieren

Sie können die Pipeline visualisieren, aber nur nachdem sie die zentrale .Rmd-Datei mindestens einmal gerendert haben:

```
> targets::tar_glimpse()     # Nur Datenobjekte
> targets::tar_visnetwork()  # Alle Objekte
```

1.6 Troubleshooting

Hilfreiche Befehle um Fehler zu lokalisieren und zu beheben.

```
> tar_progress()  # Zeigt Fortschritt und Fehler an
> tar_meta()      # Alle Metadaten
> tar_meta(fields = "warnings", complete_only = TRUE)  # Warnungen
> tar_meta(fields = "error", complete_only = TRUE)  # Fehlermeldungen
> tar_meta(fields = "seconds")  # Laufzeit der Targets
```

1.7 Projektstruktur

Die folgende Struktur erläutert die wichtigsten Bestandteile des Projekts. Während der Kompilierung werden weitere Ordner erstellt (`files/`, `temp/` `analysis` und `output/`). Die Endergebnisse werden alle in `output/` abgelegt.

```
.
├ buttons                    # Buttons (nur optische Bedeutung)
├ CHANGELOG.md               # Alle Änderungen
├ compose.yaml               # Konfiguration für Docker
├ config.toml                # Zentrale Konfigurations-Datei
├ data                       # Datensätze, auf denen die Pipeline aufbaut
├ delete_all_data.R          # Löscht den Datensatz und Zwischenschritte
├ docker-build-image.sh      # Docker Image erstellen
├ Dockerfile                 # Definition des Docker Images
├ docker-run-project.sh      # Docker Image und Datensatz kompilieren
├ functions                  # Wichtige Schritte der Pipeline
├ gpg                        # Persönlicher Public GPG-Key für Seán Fobbe
├ old                        # Alter Code aus früheren Versionen
├ pipeline.Rmd               # Zentrale Definition der Pipeline
├ README.md                  # Bedienungsanleitung
├ reports                    # Markdown-Dateien
├ requirements-python.txt    # Benötigte Python packages
├ requirements-R.R           # Benötigte R packages
├ requirements-system.txt    # Benötigte system dependencies
├ run_project.R              # Kompiliert den gesamten Datensatz
└ tex                       # LaTeX-Templates
```

1.8 Weitere Open Access Veröffentlichungen (Fobbe)

Website — https://www.seanfobbe.de

Open Data — https://zenodo.org/communities/sean-fobbe-data/

Source Code — https://zenodo.org/communities/sean-fobbe-code/

Volltexte regulärer Publikationen — https://zenodo.org/communities/sean-fobbe-publications/

1.9 Kontakt

Fehler gefunden? Anregungen? Kommentieren Sie gerne im Issue Tracker auf GitHub oder schreiben Sie mir eine E-Mail an fobbe-data@posteo.de

2 Vorbereitung

2.1 Datumsstempel

Dieser Datumsstempel wird in alle Dateinamen eingefügt. Er wird am Anfang des Skripts gesetzt, für den den Fall, dass die Laufzeit die Datumsbarriere durchbricht.

2.2 Datum und Uhrzeit (Beginn)

```
begin.script <- Sys.time()
print(begin.script)
```

```
## [1] "2023-04-07 03:52:33 CEST"
```

2.3 Packages Laden

```
library(stringr)      # String-Manipulation
library(magick)       # Cropping von PNG-Dateien
```

```
## Linking to ImageMagick 6.9.11.60
## Enabled features: fontconfig, freetype, fftw, heic, lcms, pango, webp, x11
## Disabled features: cairo, ghostscript, raw, rsvg
```

```
## Using 16 threads
```

```
library(RcppTOML)     # Verarbeitung von TOML-Format
library(ggplot2)      # Fortgeschrittene Datenvisualisierung
library(rmarkdown)    # Wissenschaftliches Reporting
library(knitr)        # Wissenschaftliches Reporting
library(kableExtra)   # Verbesserte Kable Tabellen
library(data.table)   # Fortgeschrittene Datenverarbeitung
library(quanteda)     # Fortgeschrittenes Natural Language Processing
```

```
## Package version: 3.2.4
## Unicode version: 14.0
## ICU version: 70.1
```

```
## Parallel computing: 16 of 16 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

```
library(quanteda.textplots) # Quanteda: Diagramme
library(future)       # Parallelisierung mit Futures
```

```
##
## Attaching package: 'future'
```

```
## The following object is masked from 'package:rmarkdown':
##
##     run
```

```
library(future.apply) # Apply-Funtionen für Futures
library(zip)
```

```
##
## Attaching package: 'zip'
```

```
## The following objects are masked from 'package:utils':
##
##     unzip, zip
```

2.4 Zusätzliche Funktionen einlesen

Hinweis: Die hieraus verwendeten Funktionen werden jeweils vor der ersten Benutzung in vollem Umfang angezeigt um den Lesefluss zu verbessern.

```
source("functions/f.fast.freqtable.R")
source("functions/f.future_multihashes.R")
```

2.5 Verzeichnis für Analyse-Ergebnisse und Diagramme definieren

```
dir.analysis <- paste0(getwd(),
                       "/analyse")
```

2.6 Allgemeine Konfiguration

2.6.1 Konfiguration einlesen

```
config <- parseTOML("config.toml")
```

2.6.2 Konfiguration anzeigen

```
print(config)
```

```
## List of 11
##  $ cebverfg :List of 2
##   ..$ date: chr "2023-02-26"
##   ..$ doi :List of 1
##   .. ..$ data:List of 1
##   .. .. ..$ version: chr "10.5281/zenodo.7659109"
##  $ cores    :List of 2
##   ..$ max   : logi TRUE
##   ..$ number: int 8
##  $ debug    :List of 1
##   ..$ cleanrun: logi TRUE
##  $ doi      :List of 2
##   ..$ data    :List of 2
##   .. ..$ concept: chr "10.5281/zenodo.4459405"
##   .. ..$ version: chr "10.5281/zenodo.7807020"
##   ..$ software:List of 2
##   .. ..$ concept: chr "10.5281/zenodo.4459415"
##   .. ..$ version: chr "10.5281/zenodo.7807021"
##  $ download :List of 1
##   ..$ timeout: int 600
##  $ fig      :List of 3
##   ..$ align : chr "center"
##   ..$ dpi   : int 300
##   ..$ format: chr [1:2] "pdf" "png"
##  $ freqtable:List of 1
##   ..$ ignore: chr [1:19] "text" "eingangsnummer" "datum" "doc_id" ...
##  $ license  :List of 2
##   ..$ code: chr "MIT-0"
##   ..$ data: chr "Creative Commons Zero 1.0 Universal"
##  $ parallel :List of 1
##   ..$ multihashes: logi TRUE
##  $ project  :List of 3
##   ..$ author   : chr "Seán Fobbe"
##   ..$ fullname : chr "Corona-Rechtsprechung des Bundesverfassungsgerichts"
##   ..$ shortname: chr "BVerfG-Corona"
##  $ quanteda :List of 1
##   ..$ tokens_locale: chr "de_DE"
```

2.6.3 Knitr Optionen setzen

```
knitr::opts_chunk$set(fig.path = paste0(dir.analysis, "/"),
                      dev = config$fig$format,
                      dpi = config$fig$dpi,
                      fig.align = config$fig$align)
```

2.6.4 Download Timeout setzen

```
options(timeout = config$download$timeout)
```

2.6.5 Quellenangabe für Diagramme definieren

```
caption <- paste("Fobbe | DOI:",
                 config$doi$data$version)
print(caption)
```

```
## [1] "Fobbe | DOI: 10.5281/zenodo.7807020"
```

2.6.6 Präfix für Dateien definieren

```
prefix.files <- paste0(config$project$shortname,
                 "_",
                 config$cebverfg$date)
print(prefix.files)
```

```
## [1] "BVerfG-Corona_2023-02-26"
```

2.6.7 Präfix für Diagramme definieren

```
prefix.figuretitle <- paste(config$project$shortname,
                            "| Version",
                            config$cebverfg$date)
```

2.6.8 Quanteda-Optionen setzen

```
quanteda_options(tokens_locale = config$quanteda$tokens_locale)
```

2.7 Dateien aus vorherigen Runs bereinigen

```
if(config$debug$cleanrun == TRUE){

    source("delete_all_data.R")

    }
```

2.8 Weitere Verzeichnisse definieren

```
dirs <- c("output",
          "temp",
          "files",
          "txt",
          "pdf")
```

2.9 Verzeichnisse anlegen

```
dir.create(dir.analysis)

lapply(dirs, dir.create)
```

```
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
```

2.10 Vollzitate statistischer Software schreiben

```
knitr::write_bib(c(.packages()),
                 "temp/packages.bib")
```

2.11 LaTeX Konfiguration

2.11.1 LaTeX Parameter definieren

```
latexdefs <- c("%==========================\n% Definitionen\n
    %==========================",
                "\n% NOTE: Diese Datei wurde während des Kompilierungs-Prozesses
    automatisch erstellt.\n",
                "\n%-----Autor-----",
                paste0("\\newcommand{\\projectauthor}{",
                       config$project$author,
                       "}"),
                "\n%-----Version-----",
                paste0("\\newcommand{\\version}{",
                       config$cebverfg$date,
                       "}"),
                "\n%-----Titles-----",
                paste0("\\newcommand{\\datatitle}{",
                       config$project$fullname,
                       "}"),
                paste0("\\newcommand{\\datashort}{",
                       config$project$shortname,
                       "}"),
                paste0("\\newcommand{\\softwaretitle}{Source Code der \\enquote{",
                       config$project$fullname,
                       "}}"),
                paste0("\\newcommand{\\softwareshort}{",
                       config$project$shortname,
                       "-Source}"),
                "\n%-----Data DOIs-----",
                paste0("\\newcommand{\\dataconceptdoi}{",
                       config$doi$data$concept,
                       "}"),
                paste0("\\newcommand{\\dataversiondoi}{",
                       config$doi$data$version,
                       "}"),
                paste0("\\newcommand{\\dataconcepturldoi}{https://doi.org/",
                       config$doi$data$concept,
                       "}"),
                paste0("\\newcommand{\\dataversionurldoi}{https://doi.org/",
                       config$doi$data$version,
                       "}"),
                "\n%-----Software DOIs-----",
                paste0("\\newcommand{\\softwareconceptdoi}{",
                       config$doi$software$concept,
                       "}"),
                paste0("\\newcommand{\\softwareversiondoi}{",
                       config$doi$software$version,
                       "}"),
                paste0("\\newcommand{\\softwareconcepturldoi}{https://doi.org/",
                       config$doi$software$concept,
                       "}"),
                paste0("\\newcommand{\\softwareversionurldoi}{https://doi.org/",
                       config$doi$software$version,
                       "}"))
```

2.11.2 LaTeX Parameter schreiben

```
writeLines(latexdefs,
           paste0("temp/",
                  config$project$shortname,
                  "_Definitions.tex"))
```

2.12 Parallelisierung aktivieren

Parallelisierung wird zur Beschleunigung der Konvertierung von PDF zu TXT und der Datenanalyse mittels **quanteda** und **data.table** verwendet. Die Anzahl threads wird automatisch auf das verfügbare Maximum des Systems gesetzt, kann aber auch nach Belieben auf das eigene System angepasst werden. Die Parallelisierung kann deaktiviert werden, indem die Variable **fullCores** auf 1 gesetzt wird.

2.12.1 Anzahl logischer Kerne festlegen

```
if (config$cores$max == TRUE){
    fullCores <- availableCores()
}


if (config$cores$max == FALSE){
    fullCores <- as.integer(config$cores$number)
}



print(fullCores)
```

```
## system
##     16
```

2.12.2 Quanteda

```
quanteda_options(threads = fullCores)
```

2.12.3 Data.table

```
setDTthreads(threads = fullCores)
```

3 Stamm-Datensatz einlesen (CE-BVerfG)

Der Stamm-Datensatz ist der »Corpus der Entscheidungen des Bundesverfassungsgerichts« (CE-BVerfG). Dieser enthält alle vom Bundesverfassungsgericht seit 1998 veröffentlichten Entscheidungen. Dessen **aktuellste** Version ist immer über diesen Digital Object Identifier (DOI) abrufbar: https://doi.org/10.5281/zenodo.3902658

3.1 Download der CSV-Datei

Der Datensatz im CSV-Format wird automatisch über einen verschlüsselten und langzeitstabilen Link aus dem wissenschaftlichen Archiv des CERN heruntergeladen. Dieses Vorgehen garantiert die Verwendung einer authentischen Version des Datensatzes.

```
zip.csv <- paste0("CE-BVerfG_",
                  config$cebverfg$date,
                  "_DE_CSV_Datensatz.zip")

print(zip.csv)
```

```
## [1] "CE-BVerfG_2023-02-26_DE_CSV_Datensatz.zip"
```

```
link.csv <- paste0("https://zenodo.org/record/",
                   gsub("10\\.5281/zenodo\\.([0-9]+)",
                        "\\1",
                        config$cebverfg$doi$data$version),
                   "/files/",
                   zip.csv,
                   "?download=1")

print(link.csv)
```

```
## [1] "https://zenodo.org/record/7659109/files/CE-BVerfG_2023-02-26_DE_CSV_
    Datensatz.zip?download=1"
```

```
if (file.exists(file.path("files", zip.csv)) == FALSE){

    download.file(link.csv,
                  file.path("files", zip.csv))

}
```

3.2 CSV-Datei einlesen

```
dt.bverfg <- fread(cmd = paste("unzip -cq",
                               file.path("files", zip.csv)))
```

3.3 Korpus-Objekt erstellen

```
corpus.bverfg <- corpus(dt.bverfg)
```

4 Keywords in Context (KWIC)

Bei einer KWIC-Analyse (keywords in context) wird nach einer bestimmten Zeichengefolge gesucht und sowohl diese, als auch die angrenzenden Wörter werden angezeigt. Konkret wird an dieser Stelle eine alternative Suche nach den Mustern “Corona”, “COVID” oder “SARS-CoV” durchgeführt. Groß- und Kleinschreibung wird ignoriert um eventuelle Tippfehler zu vernachlässigen. Das Sichtfenster wird auf 15 Tokens vor und nach dem Treffer gesetzt.

4.1 Tokenisierung

```
tokens <- tokens(corpus.bverfg,
                 what = "word",
                 remove_punct = FALSE,
                 remove_symbols = FALSE,
                 remove_numbers = FALSE,
                 remove_url = FALSE,
                 remove_separators = TRUE,
                 split_hyphens = FALSE,
                 include_docvars = TRUE,
                 padding = FALSE)
```

4.2 KWIC-Analyse durchführen

```
kwic <- kwic(tokens,
             pattern = "(Corona)|(COVID)|(SARS-CoV)",
             window = 15,
             valuetype = "regex",
             case_insensitive = TRUE)
```

4.3 KWIC-Tabelle speichern

```
file.kwic.sansdate <- paste(config$project$shortname,
                            "02_KeywordsInContext.csv",
                            sep = "_")

file.kwic.date <- paste(prefix.files,
                        "ANALYSE_02_KeywordsInContext.csv",
                        sep = "_")


fwrite(data.frame(kwic),
       file.path(dir.analysis,
              file.kwic.sansdate))


fwrite(data.frame(kwic),
       file.path("output",
                 file.kwic.date))
```

5 Lexical Dispersion Plot

Lexical Dispersion Plots zeigen mit einem vertikalen Strich an, an welcher Stelle in einem Dokument sich ein Token befindet. Alle Dokumente sind auf eine Länge von 1.0 normalisiert, d.h. ein Wert von 0.5 heißt immer, dass sich das Token in der Mitte des jeweiligen Dokumentes befindet. Viele und/oder dicke Striche deuten auf eine große Häufigkeit des Tokens hin.

5.1 Aufbereitung der Labels

Die Labels müssen speziell aufbereitet werden, damit sie in der Grafik noch sinnvoll anzeigbar sind. Insbesondere die Dateinamen mit Entscheidungsnamen sind zu lang.

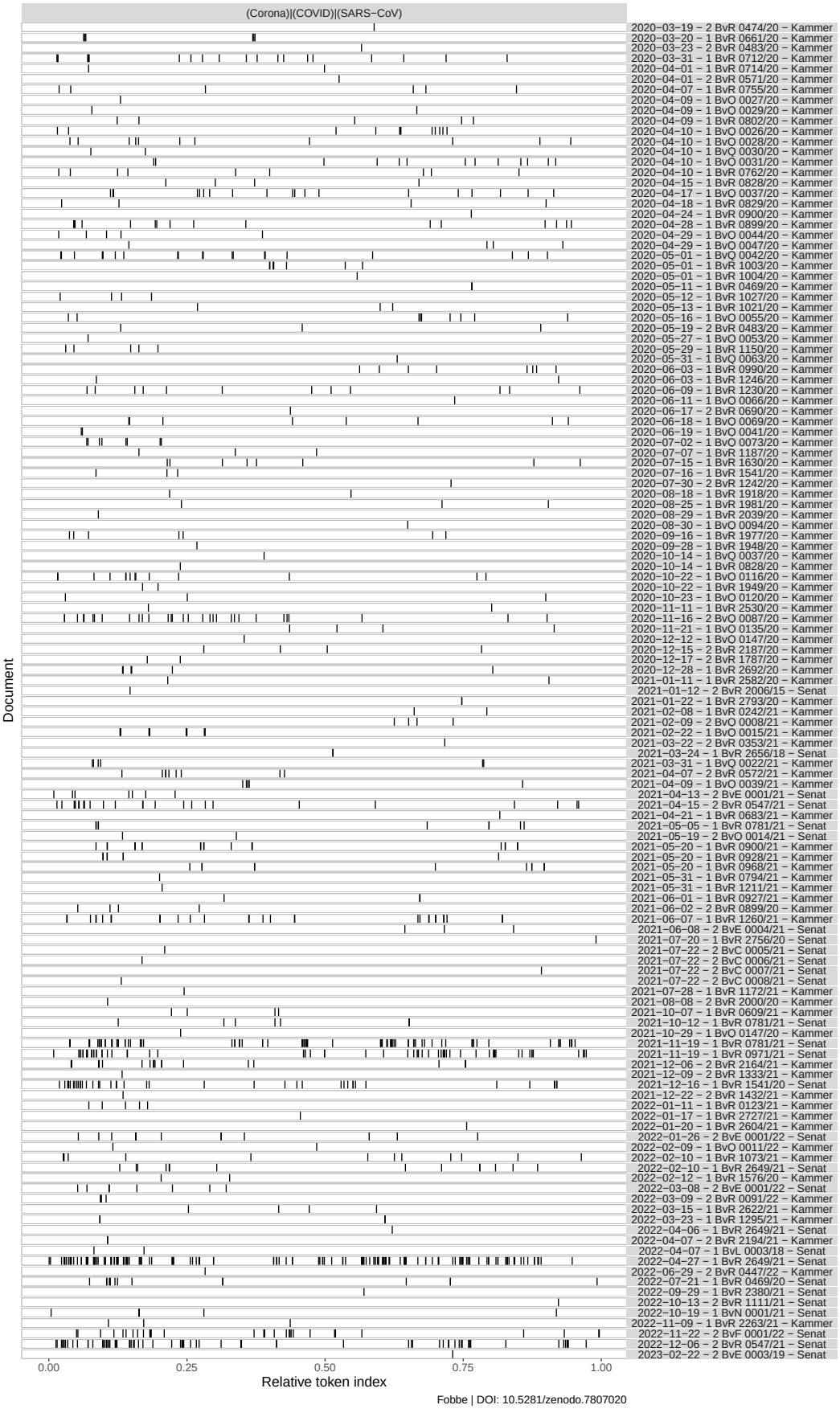
```
kwic.display <- kwic

split <- str_split(kwic$docname, pattern = "_")

f.recombine <- function(x){

    recombine <- paste0(x[2],
                        " - ",
                        x[4],
                        " ",
                        x[5],
                        " ",
                        x[6],
                        "/",
                        x[7],
                        " - ",
                        x[3] )

    recombine <- gsub("S$", "Senat", recombine)
    recombine <- gsub("K$", "Kammer", recombine)

    return(recombine)

}

filenames.display <- unlist(lapply(split, f.recombine))
kwic.display$docname <- filenames.display

split.attr <- str_split(attr(attr(kwic, "ntoken"), "names"), pattern = "_")
attrs.display <- unlist(lapply(split.attr, f.recombine))
attr(attr(kwic.display, "ntoken"), "names") <- attrs.display
```

5.2 Rechteckiges Format

```
textplot_xray(kwic.display,
              scale = "relative")+
    labs(
```

```
        title = paste(prefix.figuretitle,
                      "| Lexical Dispersion Plot"),
        caption = caption)+
    theme(
        text = element_text(size = 14),
        plot.title = element_text(size = 14,
                                  face = "bold"),
        legend.position = "none",
        plot.margin = margin(10, 20, 10, 10)
    )
```

BVerfG−Corona | Version 2023−02−26 | Lexical Dispersion Plot

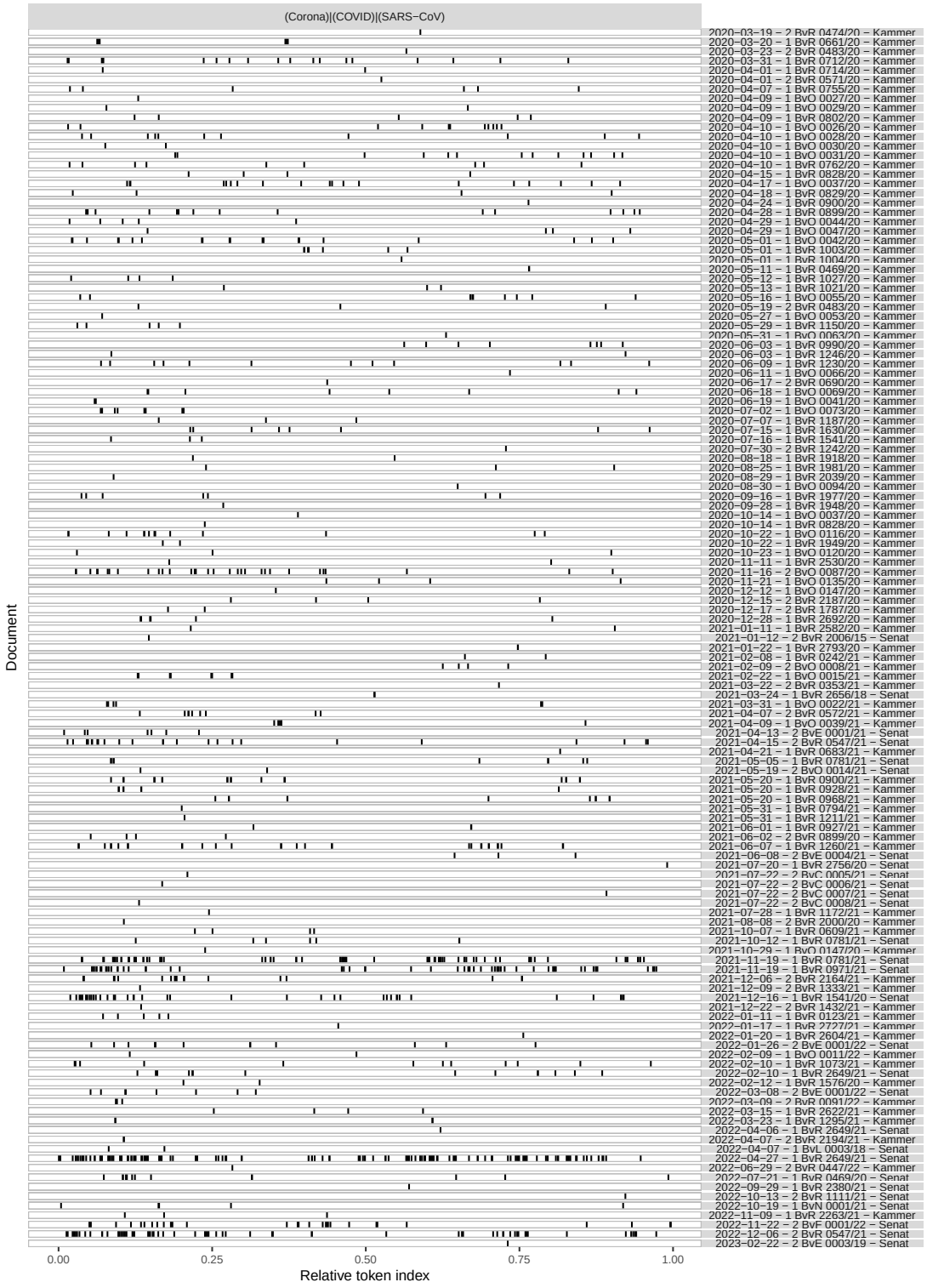


Fobbe | DOI: 10.5281/zenodo.7807020

5.3 A4-Format

```
textplot_xray(kwic.display,
              scale = "relative")+
    labs(
        title = paste(prefix.figuretitle,
                      "| Lexical Dispersion Plot"),
        caption = caption)+
    theme(
        text = element_text(size = 9),
        plot.title = element_text(size = 14,
                                  face = "bold"),
        legend.position = "none",
        plot.margin = margin(10, 20, 10, 10)
    )
```

BVerfG−Corona | Version 2023−02−26 | Lexical Dispersion Plot



Fobbe | DOI: 10.5281/zenodo.7807020

6 TXT-Datensatz erstellen

6.1 Namen der Corona-Entscheidungen definieren

```
keep.txt <- unique(kwic$docname)
```

6.2 Anzahl der TXT-Dateien

```
length(keep.txt)
```

```
## [1] 129
```

6.3 TXT-Datensatz herunterladen

```
zip.txt <- paste0("CE-BVerfG_",
                  config$cebverfg$date,
                  "_DE_TXT_Datensatz.zip")


link.txt <- paste0("https://zenodo.org/record/",
                   gsub("10\\.5281/zenodo\\.([0-9]+)",
                        "\\1",
                        config$cebverfg$doi$data$version),
                   "/files/",
                   zip.txt,
                   "?download=1")

zip.txt.rel <- file.path("files", zip.txt)



if(file.exists(zip.txt.rel) == FALSE){

    download.file(link.txt,
                  zip.txt.rel)

}
```

6.4 ZIP-Archiv entpacken

```
unzip(zip.txt.rel,
      files = keep.txt,
      exdir = "txt")
```

6.5 Corona-Entscheidungen verpacken

```
zip(paste0("output/",
           prefix.files,
           "_DE_TXT_Datensatz.zip"),
    list.files("txt", full.names = TRUE),
    mode = "cherry-pick")
```

7 PDF-Datensatz erstellen

7.1 Namen der Corona-Entscheidungen definieren

```
keep.pdf <- gsub(".txt",
                 ".pdf",
                 keep.txt)
```

7.2 Anzahl der PDF-Dateien

```
length(keep.pdf)
```

```
## [1] 129
```

7.3 PDF-Datensatz herunterladen

```
zip.pdf <- paste0("CE-BVerfG_",
                  config$cebverfg$date,
                  "_DE_PDF_Datensatz.zip")


link.pdf <- paste0("https://zenodo.org/record/",
                   gsub("10\\.5281/zenodo\\.([0-9]+)",
                        "\\1",
                        config$cebverfg$doi$data$version),
                   "/files/",
                   zip.pdf,
                   "?download=1")

zip.pdf.rel <- file.path("files", zip.pdf)


if(file.exists(zip.pdf.rel) == FALSE){

    download.file(link.pdf,
                  zip.pdf.rel)

}
```

7.4 ZIP-Archiv entpacken

```
unzip(zip.pdf.rel,
      files = keep.pdf,
      exdir = "pdf")
```

7.5 Corona-Entscheidungen verpacken

```
zip(paste0("output/",
           prefix.files,
           "_DE_PDF_Datensatz.zip"),
    list.files("pdf", full.names = TRUE),
    mode = "cherry-pick")
```

8 Frequenztabellen erstellen

8.1 CE-BVerfG auf Corona-Entscheidungen reduzieren

```
dt.corona <- dt.bverfg[doc_id %in% keep.txt]
```

8.2 Ignorierte Variablen

```
print(config$freqtable$ignore)
```

```
##  [1] "text"              "eingangsnummer"    "datum"
##  [4] "doc_id"            "seite"             "name"
##  [7] "ecli"              "aktenzeichen"      "aktenzeichen_alle"
## [10] "zeichen"           "tokens"            "typen"
## [13] "saetze"            "version"           "pressemitteilung"
## [16] "zitiervorschlag"   "kurzbeschreibung"  "url_pdf"
## [19] "url_html"
```

8.3 Liste zu prüfender Variablen

```
varlist <- names(dt.corona)

varlist <- setdiff(varlist,
                   config$freqtable$ignore)

print(varlist)
```

```
##  [1] "gericht"           "entscheidungsjahr" "entscheidung_typ"
##  [4] "spruchkoerper_typ" "spruchkoerper_az"  "registerzeichen"
##  [7] "verfahrensart"     "eingangsjahr_az"   "eingangsjahr_iso"
## [10] "kollision"         "band"              "praesi"
## [13] "v_praesi"          "richter"           "doi_concept"
## [16] "doi_version"       "lizenz"
```

8.4 Frequenztabellen erstellen

```
prefix <- paste0(config$project$shortname,
                 "_00_Frequenztabelle_var-")
```

```
f.fast.freqtable(dt.corona,
                 varlist = varlist,
                 sumrow = TRUE,
                 output.list = FALSE,
                 output.kable = TRUE,
                 output.csv = TRUE,
                 outputdir = dir.analysis,
                 prefix = prefix,
                 align = c("p{5cm}",
                           rep("r", 4)))
```

Frequency Table for Variable: gericht

1 unique value(s) detected.

gericht	N	exactpercent	roundedpercent	cumulpercent
BVerfG	129	100	100	100
Total	129	100	100	100

Frequency Table for Variable: entscheidungsjahr

4 unique value(s) detected.

entscheidungsjahr	N	exactpercent	roundedpercent	cumulpercent
2020	63	48.8372093	48.84	48.84
2021	41	31.7829457	31.78	80.62
2022	24	18.6046512	18.60	99.22
2023	1	0.7751938	0.78	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: entscheidung_typ

2 unique value(s) detected.

entscheidung_typ	N	exactpercent	roundedpercent	cumulpercent
B	127	98.449612	98.45	98.45
U	2	1.550388	1.55	100.00
Total	129	100.000000	100.00	100.00

Frequency Table for Variable: spruchkoerper_typ

2 unique value(s) detected.

spruchkoerper_typ	N	exactpercent	roundedpercent	cumulpercent
K	100	77.51938	77.52	77.52
S	29	22.48062	22.48	100.00
Total	129	100.00000	100.00	100.00

Frequency Table for Variable: spruchkoerper_az

2 unique value(s) detected.

spruchkoerper_az	N	exactpercent	roundedpercent	cumulpercent
1	94	72.86822	72.87	72.87
2	35	27.13178	27.13	100.00
Total	129	100.00000	100.00	100.00

Frequency Table for Variable: registerzeichen

7 unique value(s) detected.

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
BvC	4	3.1007752	3.10	3.10
BvE	5	3.8759690	3.88	6.98
BvF	1	0.7751938	0.78	7.75

(continued)

registerzeichen	N	exactpercent	roundedpercent	cumulpercent
BvL	1	0.7751938	0.78	8.53
BvN	1	0.7751938	0.78	9.30
BvQ	31	24.0310078	24.03	33.33
BvR	86	66.6666667	66.67	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: verfahrensart

7 unique value(s) detected.

verfahrensart	N	exactpercent	roundedpercent	cumulpercent
Abstrakte Normenkontrolle	1	0.7751938	0.78	0.78
Divergenzvorlagen eines Landesverfassungsgerichts zur Auslegung des Grundgesetzes	1	0.7751938	0.78	1.55
Einstweilige Anordnungen	31	24.0310078	24.03	25.58
Konkrete Normenkontrolle	1	0.7751938	0.78	26.36
Organstreitverfahren	5	3.8759690	3.88	30.23
Verfassungsbeschwerden; Kommunalverfassungsbeschwerden	86	66.6666667	66.67	96.90
Wahlprüfungsverfahren	4	3.1007752	3.10	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: eingangsjahr_az

6 unique value(s) detected.

eingangsjahr_az	N	exactpercent	roundedpercent	cumulpercent
15	1	0.7751938	0.78	0.78

(continued)

eingangsjahr_az	N	exactpercent	roundedpercent	cumulpercent
18	2	1.5503876	1.55	2.33
19	1	0.7751938	0.78	3.10
20	72	55.8139535	55.81	58.91
21	47	36.4341085	36.43	95.35
22	6	4.6511628	4.65	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: eingangsjahr_iso

6 unique value(s) detected.

eingangsjahr_iso	N	exactpercent	roundedpercent	cumulpercent
2015	1	0.7751938	0.78	0.78
2018	2	1.5503876	1.55	2.33
2019	1	0.7751938	0.78	3.10
2020	72	55.8139535	55.81	58.91
2021	47	36.4341085	36.43	95.35
2022	6	4.6511628	4.65	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: kollision

1 unique value(s) detected.

kollision	N	exactpercent	roundedpercent	cumulpercent
NA	129	100	100	100
Total	129	100	100	100

Frequency Table for Variable: band

6 unique value(s) detected.

band	N	exactpercent	roundedpercent	cumulpercent
NA	114	88.3720930	88.37	88.37
156	1	0.7751938	0.78	89.15
157	4	3.1007752	3.10	92.25
158	2	1.5503876	1.55	93.80
159	4	3.1007752	3.10	96.90
160	4	3.1007752	3.10	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: praesi

2 unique value(s) detected.

praesi	N	exactpercent	roundedpercent	cumulpercent
Harbarth	89	68.99225	68.99	68.99
Voßkuhle	40	31.00775	31.01	100.00
Total	129	100.00000	100.00	100.00

Frequency Table for Variable: v_praesi

2 unique value(s) detected.

v_praesi	N	exactpercent	roundedpercent	cumulpercent
Harbarth	40	31.00775	31.01	31.01
König	89	68.99225	68.99	100.00
Total	129	100.00000	100.00	100.00

Frequency Table for Variable: richter

20 unique value(s) detected.

richter	N	exactpercent	roundedpercent	cumulpercent
Baer|Ott|Radtke	17	13.1782946	13.18	13.18
Harbarth|Baer|Britz|Ott|Christ|Radtke|Härtel	2	1.5503876	1.55	14.73
Harbarth|Baer|Britz|Ott|Christ|Radtke|Härtel|Masing	1	0.7751938	0.78	15.50
Harbarth|Baer|Ott	5	3.8759690	3.88	19.38
Harbarth|Britz|Ott	2	1.5503876	1.55	20.93
Harbarth|Britz|Radtke	25	19.3798450	19.38	40.31
Harbarth|Ott|Härtel	1	0.7751938	0.78	41.09
Harbarth|Paulus|Baer|Britz|Ott|Christ|Radtke|Härtel	10	7.7519380	7.75	48.84
Hermanns|Maidowski|Langenfeld	2	1.5503876	1.55	50.39
Huber|Kessal-Wulf|König	5	3.8759690	3.88	54.26
Huber|Kessal-Wulf|Wallrabenstein	7	5.4263566	5.43	59.69
König|Huber|Hermanns|Müller|Kessal-Wulf|Langenfeld|Wallrabenstein	4	3.1007752	3.10	62.79
König|Huber|Hermanns|Müller|Kessal-Wulf|Maidowski|Langenfeld	1	0.7751938	0.78	63.57
König|Huber|Hermanns|Müller|Kessal-Wulf|Maidowski|Langenfeld|Wallrabenstein	10	7.7519380	7.75	71.32
König|Maidowski|Wallrabenstein	1	0.7751938	0.78	72.09
König|Müller|Maidowski	5	3.8759690	3.88	75.97
Masing|Paulus|Christ	15	11.6279070	11.63	87.60
Paulus|Britz|Ott|Christ|Radtke|Härtel	1	0.7751938	0.78	88.37
Paulus|Christ|Härtel	14	10.8527132	10.85	99.22
Paulus|Christ|Radtke	1	0.7751938	0.78	100.00
Total	129	100.0000000	100.00	100.00

Frequency Table for Variable: doi_concept

1 unique value(s) detected.

doi_concept	N	exactpercent	roundedpercent	cumulpercent
10.5281/zenodo.3902658	129	100	100	100
Total	129	100	100	100

Frequency Table for Variable: doi_version

1 unique value(s) detected.

doi_version	N	exactpercent	roundedpercent	cumulpercent
10.5281/zenodo.7659109	129	100	100	100
Total	129	100	100	100

Frequency Table for Variable: lizenz

1 unique value(s) detected.

lizenz	N	exactpercent	roundedpercent	cumulpercent
Creative Commons Zero 1.0 Universal	129	100	100	100
Total	129	100	100	100

9 Diagramm Kopieren

```
rechteckig.source <- list.files(dir.analysis,
                                pattern = "Rechteckig.*\\.pdf",
                                full.names = TRUE)

rechteckig.destination <- file.path("output",
                                    gsub("BVerfG-Corona",
                                         paste0(prefix.files, "_ANALYSE"),
                                         basename(rechteckig.source)))

rechteckig.destination <- gsub("-1\\.pdf",
                               "\\.pdf",
                               rechteckig.destination)

file.copy(rechteckig.source,
          rechteckig.destination)
```

```
## [1] TRUE
```

10 Erstellen der ZIP-Archive

10.1 Verpacken der Analyse-Dateien

```
zip(paste0("output/",
           prefix.files,
           "_DE_ANALYSE.zip"),
    basename(dir.analysis))
```

10.2 Verpacken der Source-Dateien

```
files.source <- c(system2("git", "ls-files", stdout = TRUE),
                  ".git")



zip(paste0("output/",
          prefix.files,
          "_Source_Files.zip"),
    files.source)
```

11 Kryptographische Hashes

Dieses Modul berechnet für jedes ZIP-Archiv zwei Arten von Hashes: SHA2-256 und SHA3-512. Mit diesen kann die Authentizität der Dateien geprüft werden und es wird dokumentiert, dass sie aus diesem Source Code hervorgegangen sind. Die SHA-2 und SHA-3 Algorithmen sind äußerst resistent gegenüber *collision* und *pre-imaging* Angriffen, sie gelten derzeit als kryptographisch sicher. Ein SHA3-Hash mit 512 bit Länge ist nach Stand von Wissenschaft und Technik auch gegenüber quantenkryptoanalytischen Verfahren unter Einsatz des *Grover-Algorithmus* hinreichend resistent.

11.1 Liste der ZIP-Archive erstellen

```
files.zip <- list.files("output",
                        pattern = "\\.zip$",
                        full.names = TRUE,
                        ignore.case = TRUE)
```

11.2 Funktion anzeigen: future_multihashes

```
print(f.future_multihashes)
```

```
## function (x)
## {
##     begin <- Sys.time()
##     message(paste("Processing", length(x), "files. Begin at:",
##         begin))
##     hashes.list <- future.apply::future_lapply(x, f.multihashes)
##     hashes.table <- data.table::rbindlist(hashes.list)
##     data.table::setDF(hashes.table)
##     end <- Sys.time()
##     duration <- end - begin
##     message(paste0("Processed ", length(x), " files. Runtime was ",
##         round(duration, digits = 2), " ", attributes(duration)$units,
##         "."))
##     return(hashes.table)
## }
```

11.3 Hashes berechnen

```
if(config$parallel$multihashes == TRUE){

    plan("multicore",
         workers = fullCores)

}else{
```

```
    plan("sequential")

     }


multihashes <- f.future_multihashes(files.zip)
```

```
## Processing 4 files. Begin at: 2023-04-07 03:56:48
```

```
## Processed 4 files. Runtime was 0.14 secs.
```

11.4 In Data Table umwandeln

```
setDT(multihashes)

setnames(multihashes,
         old = "x",
         new = "filename")
```

11.5 Index hinzufügen

```
multihashes$index <- seq_len(multihashes[,.N])
```

11.6 In Datei schreiben

```
fwrite(multihashes,
       file.path("output",
                 paste(prefix.files,
                       "KryptographischeHashes.csv",
                       sep = "_")),
       na = "NA")
```

11.7 Leerzeichen hinzufügen um Zeilenumbruch zu ermöglichen

```
multihashes$sha3.512 <- paste(substr(multihashes$sha3.512, 1, 64),
                              substr(multihashes$sha3.512, 65, 128))
```

11.8 In Bericht anzeigen

```
kable(multihashes[,.(index,filename)],
      format = "latex",
      align = c("p{1cm}",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	filename
1	output/BVerfG-Corona_2023-02-26_DE_ANALYSE.zip
2	output/BVerfG-Corona_2023-02-26_DE_PDF_Datensatz.zip
3	output/BVerfG-Corona_2023-02-26_DE_TXT_Datensatz.zip
4	output/BVerfG-Corona_2023-02-26_Source_Files.zip

```
kable(multihashes[,.(index,sha2.256)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha2.256
1	09a92b17027cfea7a71126cb35c6362d62cad9148fa3885124a757aa27aea3de
2	2b0705ea13b1e57e04877c2dabb7cd606553a864994eb9a4684a501e1a241e06
3	8da3cec3677587c2449b49f53d5a3eaa8b78ab464434391a566af97e244aaa67
4	510cba7cd6d262f80124e8cda8ffb808c74b5120ecad597207a7814f0ca594e9

```
kable(multihashes[,.(index,sha3.512)],
      format = "latex",
      align = c("c",
                "p{13cm}"),
      booktabs = TRUE,
      longtable = TRUE)
```

index	sha3.512
1	b2e7c16699e7dcfabdcab285cff4dec18e467439ef7bf0d5a11dd5ff8056ea98 b9b590b47958f57434345b2587b3a323faf1426cc0ea30e9cd12d357ce3639f7
2	05f6ddca2b7e37c56bb057fcb3d7dbfa386dfad82b3cf09ace0c25019b49bcf6 58ef0634dbe625bd6110963cd91b2e93a38ad77c90bdb8b0edf9ddd8ef985cf3
3	c17d7023c68e2001a129e0a293b03fa3a061dc2b7d0cd6626fc8ac244a6ec36c 24e6f63dc446ff36429cbcb5420fd764d958757bcc367a870279cd3973716d56
4	a939e5a607d4fbdb3731a1d2ea058c166e87dfbc5e3146d10edcc3a3fd365ff8 2de3b22988401a57b8cfb11b9d6a18c222d4dc6270a5455b296ace90e28daca9

12 Abschluss

12.1 Datumsstempel

Hinweis: der Datumsstempel weicht vom Zeitpunkt der tatsächlichen Erstellung des Datensatzes ab, weil sich der Datumsstempel nach dem Tag des Abrufs des CE-BVerfG richtet.

```
print(config$cebverfg$date)
```

```
## [1] "2023-02-26"
```

12.2 Datum und Uhrzeit (Anfang)

```
print(begin.script)
```

```
## [1] "2023-04-07 03:52:33 CEST"
```

12.3 Datum und Uhrzeit (Ende)

```
end.script <- Sys.time()
print(end.script)
```

```
## [1] "2023-04-07 03:56:48 CEST"
```

12.4 Laufzeit des gesamten Skriptes

```
print(end.script - begin.script)
```

```
## Time difference of 4.251337 mins
```

12.5 Warnungen

```
warnings()
```

13 Parameter für strenge Replikationen

```
system2("openssl", "version", stdout = TRUE)
```

```
## [1] "OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)"
```

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.2 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] zip_2.2.2                future.apply_1.10.0
##  [3] future_1.32.0            quanteda.textplots_0.94.2
##  [5] quanteda_3.2.4           data.table_1.14.8
##  [7] kableExtra_1.3.4         knitr_1.42
##  [9] ggplot2_3.4.1            magick_2.7.4
## [11] stringr_1.5.0            RcppTOML_0.2.2
## [13] rmarkdown_2.20
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0   xfun_0.37          listenv_0.9.0      lattice_0.20-45
##  [5] colorspace_2.1-0   vctrs_0.6.0        generics_0.1.3     htmltools_0.5.4
##  [9] viridisLite_0.4.1  yaml_2.3.7         utf8_1.2.3         rlang_1.1.0
## [13] pillar_1.8.1       glue_1.6.2         withr_2.5.0        lifecycle_1.0.3
## [17] munsell_0.5.0      gtable_0.3.2       rvest_1.0.3        codetools_0.2-18
## [21] evaluate_0.20      labeling_0.4.2     fastmap_1.1.1      parallel_4.2.2
## [25] fansi_1.0.4        Rcpp_1.0.10        scales_1.2.1       RcppParallel
    _5.1.7
## [29] webshot_0.5.4      farver_2.1.1       parallelly_1.34.0  systemfonts
    _1.0.4
## [33] fastmatch_1.1-3    stopwords_2.3      digest_0.6.31      stringi_1.7.12
## [37] dplyr_1.1.0        grid_4.2.2         cli_3.6.0          tools_4.2.2
```

```
## [41] magrittr_2.0.3     tibble_3.2.1       pkgconfig_2.0.3    Matrix_1.5-1
## [45] xml2_1.3.3         svglite_2.1.1      httr_1.4.5         rstudioapi_0.14
## [49] R6_2.5.1           globals_0.16.2     compiler_4.2.2
```

14 Changelog

14.1 Version 2023-02-26

- Vollständige Aktualisierung der Daten
- Gesamte Laufzeitumgebung nun mit Docker versionskontrolliert
- Funktionen werden nun nicht mehr aus einem Submodule bezogen, sondern sind direkt im Projekt verankert
- Vereinfachung der Konfigurations-Datei
- ZIP-Archiv mit Source Code wird nun aus dem Git-Manifest generiert
- README im Hinblick auf Docker überarbeitet
- Speichern von temporären Dateien nun in speziellen Ordnern in files/, pdf/ und txt/
- Option für automatische Löschung der Dateien aus vorherigen Runs zu Konfiguration hinzugefügt
- Delete-Skript hinzugefügt

14.2 Version 2022-08-24

- Vollständige Aktualisierung der Daten
- Diagramme sind deutlich überarbeitet und die Labels verschönert worden
- Umbenennung des run scripts und der Konfigurations-Datei

14.3 Version 2022-02-01

- Vollständige Aktualisierung der Daten
- Strenge Versionskontrolle von R packages mit **renv**
- Kompilierung jetzt detailliert konfigurierbar, insbesondere die Parallelisierung
- Parallelisierung nun vollständig mit *future* statt mit *foreach* und *doParallel*
- Fehlerhafte Kompilierungen werden vor der nächsten Kompilierung vollautomatisch aufgeräumt
- Alle Ergebnisse werden automatisch fertig verpackt in den Ordner 'output' sortiert
- README und CHANGELOG sind jetzt externe Markdown-Dateien, die bei der Kompilierung automatisiert eingebunden werden

14.4 Version 2021-09-19

- Vollständige Aktualisierung der Daten

14.5 Version 2021-05-20

- Vollständige Aktualisierung der Daten

14.6 Version 2021-01-08

- Erstveröffentlichung

Literaturverzeichnis

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2023. *Rmarkdown: Dynamic Documents for R.*

Bengtsson, Henrik. 2021. "A Unifying Framework for Parallel and Distributed Processing in R Using Futures." *The R Journal* 13 (2): 208–27. https://doi.org/10.32614/RJ-2021-048.

———. 2022. *Future.apply: Apply Function to Elements in Parallel Using Futures.*

———. 2023. *Future: Unified Parallel and Distributed Processing in R for Everyone.*

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018a. "Quanteda: An R Package for the Quantitative Analysis of Textual Data." *Journal of Open Source Software* 3 (30): 774. https://doi.org/10.21105/joss.00774.

———. 2018b. "Quanteda: An R Package for the Quantitative Analysis of Textual Data." *Journal of Open Source Software* 3 (30): 774. https://doi.org/10.21105/joss.00774.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, Akitaka Matsuo, and William Lowe. 2022. *Quanteda: Quantitative Analysis of Textual Data.* https://quanteda.io.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2022. *Quanteda.textplots: Plots for the Quantitative Analysis of Textual Data.*

Csárdi, Gábor, Kuba Podgórski, and Rich Geldreich. 2022. *Zip: Cross-Platform Zip Compression.* https://github.com/r-lib/zip#readme.

Dowle, Matt, and Arun Srinivasan. 2023. *Data.table: Extension of 'Data.frame'.*

Eddelbuettel, Dirk. 2023. *RcppTOML: Rcpp Bindings to Parser for "Tom's Obvious Markup Language".* http://dirk.eddelbuettel.com/code/rcpp.toml.html.

Ooms, Jeroen. 2023. *Magick: Advanced Graphics and Image-Processing in R.*

R Core Team. 2022. *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York. https://ggplot2.tidyverse.org.

———. 2022. *Stringr: Simple, Consistent Wrappers for Common String Operations.*

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2023. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics.*

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC.

———. 2015. *Dynamic Documents with R and Knitr.* 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. https://yihui.org/knitr/.

———. 2023. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://yihui.org/knitr/.

Xie, Yihui, J. J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown-cookbook.

Zhu, Hao. 2021. *KableExtra: Construct Complex Table with Kable and Pipe Syntax*.