Corpus des Deutschen Bundesrechts

(C-DBR)

Compilation Report

Version 2023-10-03

DOI: 10.5281/zenodo.8402568

Titel	Source Code des »Corpus des Deutschen Bundesrechts«
Abkürzung	C-DBR-Source
Autor	Seán Fobbe
Version	2023-10-03
Download	https://doi.org/10.5281/zenodo.8402568
Lizenz	MIT No Attribution (MIT-0)

Zitiervorschlag

Seán Fobbe (2023). Source Code des »Corpus des Deutschen Bundesrechts« (C-DBR-Source). Version 2023-10-03. Zenodo. DOI: 10.5281/zenodo.8402568.

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 2023-10-03. 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.4072934. 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 öffentlichen Stellen der Bundesrepublik Deutschland.

Inhaltsverzeichnis

1 README: Corpus des Deutschen Bundesrechts (C-DBR)

1.1 Überblick

Das **Corpus des deutschen Bundesrechts (C-DBR)** ist eine möglichst vollständige Sammlung der konsolidierten Fassungen aller Gesetze und Verordnungen auf Bundesebene. Der Datensatz nutzt als seine Datenquelle das amtliche Internetangebot www.gesetze-im-internet.de des Bundesministeriums der Justiz und wertet dieses vollständig aus.

Alle mit diesem Skript erstellten Datensätze werden dauerhaft kostenlos und urheberrechtsfrei auf Zenodo, dem wissenschaftlichen Archiv des CERN, veröffentlicht. Alle Versionen sind mit einem separaten und langzeit-stabilen (persistenten) Digital Object Identifier (DOI) versehen.

Aktuellster, funktionaler und zitierfähiger Release des Datensatzes: https://doi.org/10.5281/zenodo.3832111

Aktuellster, funktionaler und zitierfähiger Release des Source Codes: https://doi.org/10.5281/zenodo.4072934

1.2 Funktionsweise

Primäre Endprodukte des Skripts sind folgende ZIP-Archive:

1. Der volle Datensatz im CSV-Format, unterteilt in Einzelnormen (nur Rechtsakte mit veröffentlichtem Normtext)
2. Die Metadaten aller Einzelnormen im CSV-Format (wie 1, aber ohne Text-Variable)
3. Der volle Datensatz im CSV-Format, unterteilt in Rechtsakte (nur Rechtsakte mit veröffentlichtem Normtext)
4. Die Metadaten aller Rechtsakte im CSV-Format (wie 3, aber ohne Text-Variable)
5. Die Metadaten aller veröffentlichten Rechtsakte, im CSV-Format (unabhängig davon ob Normtext veröffentlicht wurde)
6. Der volle Datensatz im XML-Format, unterteilt in Rechtsakte (Originaldaten von GII)
7. Alle Anlagen zu den XML-Dateien im jeweiligen Original-Format (Originaldaten von GII)
8. Alle Rechtsakte im TXT-Format, unterteilt in Rechtsakte (deutlich reduzierter Umfang an Metadaten)
9. Alle Rechtstexte im PDF-Format, unterteilt in Rechtsakte (deutlich reduzierter Umfang an Metadaten)
10. Alle Rechtstexte im EPUB-Format, unterteilt in Gesetze (deutlich reduzierter Umfang an Metadaten)
11. Alle Analyse-Ergebnisse (Tabellen als CSV, Grafiken als PDF und PNG)
12. Netzwerk-Strukturen (Adjazenzmatrizen, Edgelists, GraphML, und Netzwerk-Diagramme) für alle Rechtsakte (experimentell!)

Alle Ergebnisse werden im Ordner `output` abgelegt. Zusätzlich werden für alle ZIP-Archive kryptographische Signaturen (SHA2-256 und SHA3-512) berechnet und in einer CSV-Datei hinterlegt.

1.3 Systemanforderungen

- Docker

- Docker Compose
- 8 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/c-dbr.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 Packages laden

```
library(targets)
library(tarchetypes)
library(RcppTOML)
library(future)
library(data.table)
library(quanteda)
#> 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(knitr)
library(kableExtra)
library(igraph)
#>
#> Attaching package: 'igraph'
#> The following objects are masked from 'package:future':
#>
#>     %->%, %<-%
#> The following objects are masked from 'package:stats':
#>
#>     decompose, spectrum
#> The following object is masked from 'package:base':
#>
#>     union
library(ggraph)
#> Loading required package: ggplot2

tar_unscript()
```

3 Vorbereitung

3.1 Definitionen

```
## Datum
datestamp <- Sys.Date()
print(datestamp)
#> [1] "2023-10-03"

## Datum und Uhrzeit (Beginn)
begin.script <- Sys.time()

## Konfiguration
config <- RcppTOML::parseTOML("config.toml")
print(config)
#> List of 9
#>  $ cores   :List of 2
#>   ..$ max   : logi TRUE
#>   ..$ number: int 8
#>  $ debug   :List of 4
#>   ..$ cleanrun: logi FALSE
#>   ..$ qaSample: int 50
#>   ..$ sample  : int 500
#>   ..$ toggle  : logi FALSE
#>  $ doi     :List of 2
#>   ..$ data    :List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.3832111"
#>   .. ..$ version: chr "10.5281/zenodo.8402567"
#>   ..$ software:List of 2
#>   .. ..$ concept: chr "10.5281/zenodo.4072934"
#>   .. ..$ version: chr "10.5281/zenodo.8402568"
#>  $ download:List of 1
#>   ..$ timeout: int 10
#>  $ fig     :List of 3
#>   ..$ align : chr "center"
#>   ..$ dpi   : int 300
#>   ..$ format: chr [1:2] "pdf" "png"
#>  $ license :List of 2
#>   ..$ code: chr "MIT-0"
#>   ..$ data: chr "Creative Commons Zero 1.0 Universal"
#>  $ parallel:List of 10
#>   ..$ downloadEPUB     : logi TRUE
#>   ..$ downloadPDF      : logi TRUE
#>   ..$ downloadXML      : logi TRUE
#>   ..$ extractPDF       : logi TRUE
#>   ..$ htmlLandingPages : logi TRUE
#>   ..$ lingsummarize    : logi TRUE
#>   ..$ multihashes      : logi TRUE
#>   ..$ parseEinzelnormen: logi FALSE
#>   ..$ parseMeta        : logi FALSE
#>   ..$ parseNetworks    : logi FALSE
#>  $ project :List of 3
#>   ..$ author   : chr "Seán Fobbe"
#>   ..$ fullname : chr "Corpus des Deutschen Bundesrechts"
```

```
#>   ..$ shortname: chr "C-DBR"
#>  $ quanteda:List of 1
#>   ..$ tokens_locale: chr "de_DE"


# Analyse-Ordner
dir.analysis <- paste0(getwd(),
                       "/analysis")
```

3.2 Aufräumen

Löscht Dateien im Output-Ordner, die nicht vom heutigen Tag sind.

```
unlink(grep(datestamp,
            list.files("output",
                       full.names = TRUE),
            invert = TRUE,
            value = TRUE))
```

3.3 Ordner erstellen

```
#unlink("output", recursive = TRUE)
dir.create("files", showWarnings = FALSE)
dir.create("output", showWarnings = FALSE)
dir.create("temp", showWarnings = FALSE)

dir.create(dir.analysis, showWarnings = FALSE)
```

3.4 Vollzitate statistischer Software schreiben

```
knitr::write_bib(renv::dependencies()$Package,
                 "temp/packages.bib")
#> Finding R package dependencies ... Done!
```

4 Globale Variablen

4.1 Packages definieren

```
tar_option_set(packages = c("tarchetypes",
                            "RcppTOML",      # TOML-Dateien lesen und schreiben
                            "testthat",      # Unit Tests
                            "fs",            # Verbessertes File Handling
                            "zip",           # Verbessertes ZIP Handling
                            "mgsub",         # Vektorisiertes Gsub
                            "httr",          # HTTP-Werkzeuge
                            "xml2",          # XML-Extraktion
                            "rvest",         # HTML-Extraktion
                            "knitr",         # Professionelles Reporting
                            "kableExtra",    # Verbesserte Kable Tabellen
                            "pdftools",      # Verarbeitung von PDF-Dateien
                            "ggplot2",       # Datenvisualisierung
                            "ggraph",        # Visualisierung von Graphen
                            "igraph",        # Analyse von Graphen
                            "scales",        # Skalierung von Diagrammen
                            "data.table",    # Fortgeschrittene Datenverarbeitung
                            "readtext",      # TXT-Dateien einlesen
                            "quanteda",      # Computerlinguistik
                            "future",        # Parallelisierung
                            "future.apply"))# Funktionen für Future

tar_option_set(workspace_on_error = TRUE) # Save Workspace on Error
tar_option_set(format = "qs")

#> Establish _targets.R and _targets_r/globals/global-packages.R.
```

4.2 Konfiguration

```
datestamp <- Sys.Date()

config <- RcppTOML::parseTOML("config.toml")

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

## Caption for diagrams
caption <- paste("Fobbe | DOI:",
                 config$doi$data$version)


## Prefix for figure titles
prefix.figuretitle <- paste(config$project$shortname,
                            "| Version",
                            datestamp)

## File prefix
```

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


if (config$cores$max == TRUE){
    fullCores <- future::availableCores() - 1
}


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

#> Establish _targets.R and _targets_r/globals/global-config.R.
```

4.3 Funktionen definieren

```
lapply(list.files("functions", pattern = "\\.R$", full.names = TRUE), source)

#> Establish _targets.R and _targets_r/globals/global-functions.R.
```

4.4 ZIP-Datei für Source definieren

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

#> Establish _targets.R and _targets_r/globals/global-sourcefiles.R.
```

5 Pipeline: Konstruktion

5.1 File Tracking Targets

Mit diesem Abschnitt der Pipeline werden Input-Dateien getrackt und eingelesen. Mit der Option »format = "file"« werden für Input-Dateien Prüfsummen berechnet. Falls sich diese verändern werden alle von ihnen abhängigen Pipeline-Schritte als veraltet markiert und neu berechnet.

5.1.1 Source Code

Dies sind alle Dateien, die den Source Code für den Datensatz bereitstellen.

```
tar_target(files.source,
           files.source.raw,
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.file.source.R.
```

5.1.2 Changelog

```
tar_target(changelog,
           "CHANGELOG.md",
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.file.changelog.R.
```

5.1.3 Liste aller Variablen im Codebook

Die Variablen des Datensatzes, inklusive ihrer Erläuterung.

```
list(
    tar_target(file.var_codebook,
               "data/C-DBR_Variables.csv",
               format = "file"),
    tar_target(dt.var_codebook,
               fread(file.var_codebook))
)
#> Establish _targets.R and _targets_r/targets/tar.file.varlist.R.
```

5.2 Download Targets

Es werden von www.gesetze-im-internet.de alle Rechtsakte im XML-, EPUB- und PDF-Format heruntergeladen und auf der Festplatte gespeichert. Die Document Type Definition (DTD) für die XML-Dateien wird ebenfalls archiviert.

5.2.1 URLs für XML-Archive

```
tarchetypes::tar_age(url.xml,
                     f.links_xml(url = "https://www.gesetze-im-internet.de/gii-
    toc.xml"),
                     age = as.difftime(1, units = "days"))

#> Establish _targets.R and _targets_r/targets/tar.download.xmlurl.R.
```

5.2.2 Tabelle der Dateinamen erstellen

```
tarchetypes::tar_age(dt.filenames,
                     f.html_landing_pages(url.xml,
                                          multicore = config$parallel$
    htmlLandingPages,
                                          cores = fullCores),
                     age = as.difftime(1, units = "days"))

#> Establish _targets.R and _targets_r/targets/tar.download.filenames.R.
```

5.2.3 Download Tabelle erstellen

```
tar_target(dt.download,
           f.download_table_make(dt.filenames = dt.filenames,
                                 url.xml = url.xml,
                                 xml.toc = "https://www.gesetze-im-internet.de/
    gii-toc.xml"))

#> Establish _targets.R and _targets_r/targets/tar.download.table.R.
```

5.2.4 Konkordanzabelle erstellen

```
tar_target(dt.conctable,
           f.conctable(dt.download = dt.download))

#> Establish _targets.R and _targets_r/targets/tar.download.conc.R.
```

5.2.5 Document Type Definition (DTD) herunterladen

```
tar_target(file.dtd,
           f.download("https://www.gesetze-im-internet.de/dtd/1.01/gii-norm.dtd",
                      paste0(prefix.files,
                             "_DE_XML_Document-Type-Definition_v1-01.dtd"),
                      dir = "output",
                      clean = FALSE),
```

```
            format = "file")

#> Establish _targets.R and _targets_r/targets/tar.download.dtd.R.
```

5.2.6 XML (ZIP)-Archive herunterladen

```
tar_target(files.xmlzip,
                f.download(url = dt.download$url.xml,
                           filename = dt.download$title.xml,
                           dir = "files/xml_zip",
                           clean = TRUE,
                           multicore = config$parallel$downloadXML,
                           cores = fullCores,
                           sleep.min = 0,
                           sleep.max = 0,
                           retries = 3,
                           retry.sleep.min = 1,
                           retry.sleep.max = 2,
                           timeout = config$download$timeout,
                           debug.toggle = FALSE,
                           debug.files = 500),
                format = "file")

#> Establish _targets.R and _targets_r/targets/tar.download.xmlzip.R.
```

5.2.7 PDF-Dateien herunterladen

```
tar_target(files.pdf,
                f.download(url = dt.download$url.pdf,
                           filename = dt.download$title.pdf,
                           dir = "files/pdf",
                           clean = TRUE,
                           multicore = config$parallel$downloadPDF,
                           cores = fullCores,
                           sleep.min = 0,
                           sleep.max = 0,
                           retries = 3,
                           retry.sleep.min = 1,
                           retry.sleep.max = 2,
                           timeout = config$download$timeout,
                           debug.toggle = FALSE,
                           debug.files = 500),
                format = "file")

#> Establish _targets.R and _targets_r/targets/tar.download.pdf.R.
```

5.2.8 EPUB-Dateien herunterladen

```
tar_target(files.epub,
```

```
                f.download(url = dt.download$url.epub,
                           filename = dt.download$title.epub,
                           dir = "files/epub",
                           clean = TRUE,
                           multicore = config$parallel$downloadEPUB,
                           cores = fullCores,
                           sleep.min = 0,
                           sleep.max = 0,
                           retries = 3,
                           retry.sleep.min = 1,
                           retry.sleep.max = 2,
                           timeout = config$download$timeout,
                           debug.toggle = FALSE,
                           debug.files = 500),
                format = "file")

#> Establish _targets.R and _targets_r/targets/tar.download.epub.R.
```

5.3 Convert Targets

Dieser Abschnitt entpackt die ZIP-Dateien, erstellt ein Target mit allen XML-Dateien und konvertiert die PDF-Dateien in das TXT-Format.

5.3.1 Entpacken

```
tar_target(files.xml.all,
           f.tar_unzip(zipfiles = files.xmlzip,
                       exdir = "files/xml"),
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.convert.unzip.R.
```

5.3.2 XML-Dateien bestimmen

```
tar_target(files.xml,
           files.xml.all[grepl("\\.xml$", files.xml.all)],
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.convert.xmlfiles.R.
```

5.3.3 PDF zu TXT konvertieren

```
tar_target(files.txt,
           f.tar_pdf_extract(x = files.pdf,
                             outputdir = "files/txt",
                             multicore = config$parallel$extractPDF,
                             cores = fullCores),
           format = "file")
```

```
#> Establish _targets.R and _targets_r/targets/tar.convert.txt.R.
```

5.4 Parse Targets

Der Abschnitt zum Parsing extrahiert aus den XML-Dateien alle relevanten Normtexte und Metadaten. Es wird auch eine Netzwerk-Analyse der Struktur der Rechtsakte durchgeführt.

5.4.1 Datensatz erstellen: Einzelnormen

```
tar_target(dt.normen,
           f.dt.einzelnormen(file.xml = files.xml,
                             multicore = config$parallel$parseEinzelnormen,
                             cores = fullCores))

#> Establish _targets.R and _targets_r/targets/tar.parse.normen.R.
```

5.4.2 Datensatz erstellen: Rechtsakte (mit Text)

```
tar_target(dt.rechtsakte,
           f.dt.rechtsakte(dt.normen))

#> Establish _targets.R and _targets_r/targets/tar.parse.rechtsakte.R.
```

5.4.3 Datensatz erstellen: XML-Metadaten

Diese Datei unterscheidet sich von der Variante der "Rechtsakte (Metadaten)", weil sie auch Rechtsakte enthält, die ohne Text veröffentlicht wurden. Die Differenz betrifft etwa 1000 Rechtsakte, ist also erheblich.

```
tar_target(dt.meta,
           f.dt.meta(file.xml = files.xml,
                     multicore = config$parallel$parseMeta,
                     cores = fullCores))

#> Establish _targets.R and _targets_r/targets/tar.parse.xmlmeta.R.
```

5.4.4 Netzwerk-Analyse

```
tar_target(files.network,
           f.network.analysis(files.xml = files.xml,
                              prefix.figuretitle = prefix.figuretitle,
                              caption = caption,
                              dir.out = "netzwerke",
                              multicore = config$parallel$parseNetworks,
                              cores = fullCores),
```

```
            format = "file")

#> Establish _targets.R and _targets_r/targets/tar.parse.networks.R.
```

5.5 Enhance Targets

Hier werden vereinzelte Verbesserungen vorgenommen und weitere Variablen hinzugefügt. Schließlich werden die geprüften finalen Varianten erstellt.

5.5.1 Variablen erstellen: »zeichen, token, typen, saetze«

Berechnung klassischer linguistischer Kennzahlen.

```
tar_target(var_lingstats.normen,
                f.lingstats(dt.normen,
                            multicore = config$parallel$lingsummarize,
                            cores = fullCores,
                            germanvars = TRUE))
#> Establish _targets.R and _targets_r/targets/tar.enhance.lingstats.normen.R.
```

```
tar_target(var_lingstats.rechtsakte,
                f.lingstats(dt.rechtsakte,
                            multicore = config$parallel$lingsummarize,
                            cores = fullCores,
                            germanvars = TRUE))
#> Establish _targets.R and _targets_r/targets/tar.enhance.lingstats.rechtsakte.R
    .
```

5.5.2 Finale Datensätze erstellen

```
tar_target(dt.normen.final,
           f.finalize_einzelnormen(dt.normen = dt.normen,
                                   lingstats = var_lingstats.normen))
#> Establish _targets.R and _targets_r/targets/tar.enhance.finalize.normen.R.
```

```
tar_target(dt.rechtsakte.final,
           f.finalize_rechtsakte(dt.rechtsakte = dt.rechtsakte,
                                 lingstats = var_lingstats.rechtsakte))
#> Establish _targets.R and _targets_r/targets/tar.enhance.finalize.rechtsakte.R.
```

5.5.3 Varianten erstellen: Nur Metadaten

```
tar_target(dt.normen.meta,
           dt.normen.final[, !"text"])
#> Establish _targets.R and _targets_r/targets/tar.enhance.finalize.normen.meta.R
    .
```

```
tar_target(dt.rechtsakte.meta,
           dt.rechtsakte.final[, !"text"])
#> Establish _targets.R and _targets_r/targets/tar.enhance.finalize.rechtsakte.
    meta.R.
```

5.6 Write Targets

Dieser Abschnitt der Pipeline schreibt den Datensatz und alle Hash-Prüfsummen auf die Festplatte.

```
values <- tibble::tibble(
                        name = c("download",
                                 "conctable",
                                 "normen",
                                 "normen_meta",
                                 "rechtsakte",
                                 "rechtsakte_meta",
                                 "xml_meta"),
                        input = c(quote(dt.download),
                                  quote(dt.conctable),
                                  quote(dt.normen.final),
                                  quote(dt.normen.meta),
                                  quote(dt.rechtsakte.final),
                                  quote(dt.rechtsakte.meta),
                                  quote(dt.meta)
                                  ),
                        filename = paste0(prefix.files,
                                          c("_02_Download-Tabelle",
                                            "_DE_Alle-Rechtsakte-Verzeichnis",
                                            "_DE_CSV_Einzelnormen_Datensatz",
                                            "_DE_CSV_Einzelnormen_Metadaten",
                                            "_DE_CSV_Rechtsakte_Datensatz",
                                            "_DE_CSV_Rechtsakte_Metadaten",
                                            "_DE_CSV_Metadaten-XML_Datensatz"),
                                          ".csv"),
                        dir = c(dir.analysis,
                                rep("output", 6))
                    )



csv.all <- tarchetypes::tar_map(unlist = FALSE,
                                values = values,
                                names = name,
                                tar_target(csv,
                                           f.tar_fwrite(x = input,
                                                        filename = file.path(dir,

    filename)),
                                           format = "file")
                                )
```

```
#> Establish _targets.R and _targets_r/targets/tar.write.R.
```

5.7 Report Targets

Dieser Abschnitt der Pipeline erstellt die finalen Berichte (Codebook und Robustness Checks).

5.7.1 LaTeX-Definitionen schreiben

Um Variablen aus der Pipeline in die LaTeX-Kompilierung einzuführen, müssen diese als .tex-Datei auf die Festplatte geschrieben werden.

```
tar_target(latexdefs,
                f.latexdefs(config,
                            dir = "temp",
                            version = datestamp),
           format = "file")

#> Establish _targets.R and _targets_r/targets/tar.report.defs.R.
```

5.7.2 Zusammenfassungen linguistischer Kennwerte berechnen

```
tar_target(lingstats.summary.normen,
                f.lingstats_summary(dt.normen.final,
                                    germanvars = TRUE))

#> Establish _targets.R and _targets_r/targets/tar.report.lingstatsummary.normen.
    R.
```

```
tar_target(lingstats.summary.rechtsakte,
                f.lingstats_summary(dt.rechtsakte.final,
                                    germanvars = TRUE))

#> Establish _targets.R and _targets_r/targets/tar.report.lingstatsummary.
    rechtsakte.R.
```

5.7.3 Report erstellen: Robustness Checks

```
tarchetypes::tar_render(report.robustness,
                        file.path("reports",
                                  "RobustnessChecks.Rmd"),
                        output_file = file.path("../output",
                                                paste0(config$project$shortname,
                                                       "_",
                                                       datestamp,
```

```
                                                    "_RobustnessChecks.pdf")))

#> Establish _targets.R and _targets_r/targets/tar.report.robustness.R.
```

5.7.4 Report erstellen: Codebook

```
tarchetypes::tar_render(report.codebook,
                        file.path("reports",
                                  "Codebook.Rmd"),
                        output_file = file.path("../output",
                                                paste0(config$project$shortname,
                                                       "_",
                                                       datestamp,
                                                       "_Codebook.pdf")))

#> Establish _targets.R and _targets_r/targets/tar.report.codebook.R.
```

5.8 ZIP Targets

Diese Abschnitt der Pipeline erstellt ZIP-Archive für alle zentralen Rechenergebnisse und speichert diese im Ordner »output«.

5.8.1 ZIP erstellen: Static Branching

```
values <- tibble::tibble(
                        name = c("source",
                                 "networks",
                                 "pdf",
                                 "txt",
                                 "epub",
                                 "xml",
                                 "attachments",
                                 "einzelnormen",
                                 "einzelnormen_meta",
                                 "rechtsakte",
                                 "rechtsakte_meta",
                                 "xml_meta"),
                        input = c(quote(files.source),
                                  quote(files.network),
                                  quote(files.pdf),
                                  quote(files.txt),
                                  quote(files.epub),
                                  quote(files.xml),
                                  quote(setdiff(files.xml.all, files.xml)),
                                  quote(csv_normen),
                                  quote(csv_normen_meta),
                                  quote(csv_rechtsakte),
                                  quote(csv_rechtsakte_meta),
                                  quote(csv_xml_meta)),
                        filename = paste0(prefix.files,
                                          c("_Source_Code",
```

```
                                          "_DE_Netzwerke",
                                          "_DE_PDF_Datensatz",
                                          "_DE_TXT_Datensatz",
                                          "_DE_EPUB_Datensatz",
                                          "_DE_XML_Datensatz",
                                          "_DE_XML_Anlagen",
                                          "_DE_CSV_Einzelnormen_Datensatz",
                                          "_DE_CSV_Einzelnormen_Metadaten",
                                          "_DE_CSV_Rechtsakte_Datensatz",
                                          "_DE_CSV_Rechtsakte_Metadaten",
                                          "_DE_CSV_Metadaten-XML"),
                                        ".zip"),
                      mode = c(rep("mirror", 2),
                               rep("cherry-pick", 10))
                  )



zip.list <- tarchetypes::tar_map(unlist = FALSE,
                                 values = values,
                                 names = name,
                                 tar_target(zip,
                                            f.tar_zip(x = input,
                                                      filename = filename,
                                                      dir = "output",
                                                      mode = mode),
                                            format = "file")
                                 )

#> Establish _targets.R and _targets_r/targets/tar.zip.R.
```

5.8.2 ZIP erstellen: Analyse-Dateien

```
tar_target(zip_analysis,
           f.tar_zip("analysis/",
                     filename = paste(prefix.files,
                                      "DE_Analyse.zip",
                                      sep = "_"),
                     dir = "output",
                     mode = "cherry-pick",
                     report.codebook,    # manually enforced dependency
    relationship
                     report.robustness), # manually enforced dependency
    relationship
           format = "file")
#> Establish _targets.R and _targets_r/targets/tar.zip.analysis.R.
```

5.9 Kryptographische Hashes

Zum Ende hin werden für alle wichtigen Ergebnisse kryptographische Prüfsummen berechnet, die abschließend (außerhalb der Pipeline) mit dem persönlichen GPG-Key von Seán Fobbe signiert werden.

5.9.1 Zu hashende ZIP-Archive definieren

```
tar_target(zip.all,
           c(zip_source,
             zip_analysis,
             zip_networks,
             zip_pdf,
             zip_txt,
             zip_epub,
             zip_xml,
             zip_attachments,
             zip_einzelnormen,
             zip_einzelnormen_meta,
             zip_rechtsakte,
             zip_rechtsakte_meta,
             zip_xml_meta))
#> Establish _targets.R and _targets_r/targets/tar.hashes.list.R.
```

5.9.2 Kryptographische Hashes berechnen

```
tar_target(hashes,
           f.tar_multihashes(c(zip.all,
                               report.codebook[1],
                               report.robustness[1]),
                             multicore = config$parallel$multihashes,
                             cores = fullCores))
#> Establish _targets.R and _targets_r/targets/tar.hashes.compute.R.
```

5.9.3 CSV schreiben: Kryptographische Hashes

```
tar_target(csv.hashes,
           f.tar_fwrite(x = hashes,
                        filename = file.path("output",
                                             paste0(prefix.files,
                                                    "_KryptographischeHashes.csv"
    ))
                        )
           )
#> Establish _targets.R and _targets_r/targets/tar.hashes.csv.R.
```

6 Pipeline: Kompilierung

6.1 Durchführen der Kompilierung

```
tar_make()
```

6.2 Pipeline archivieren

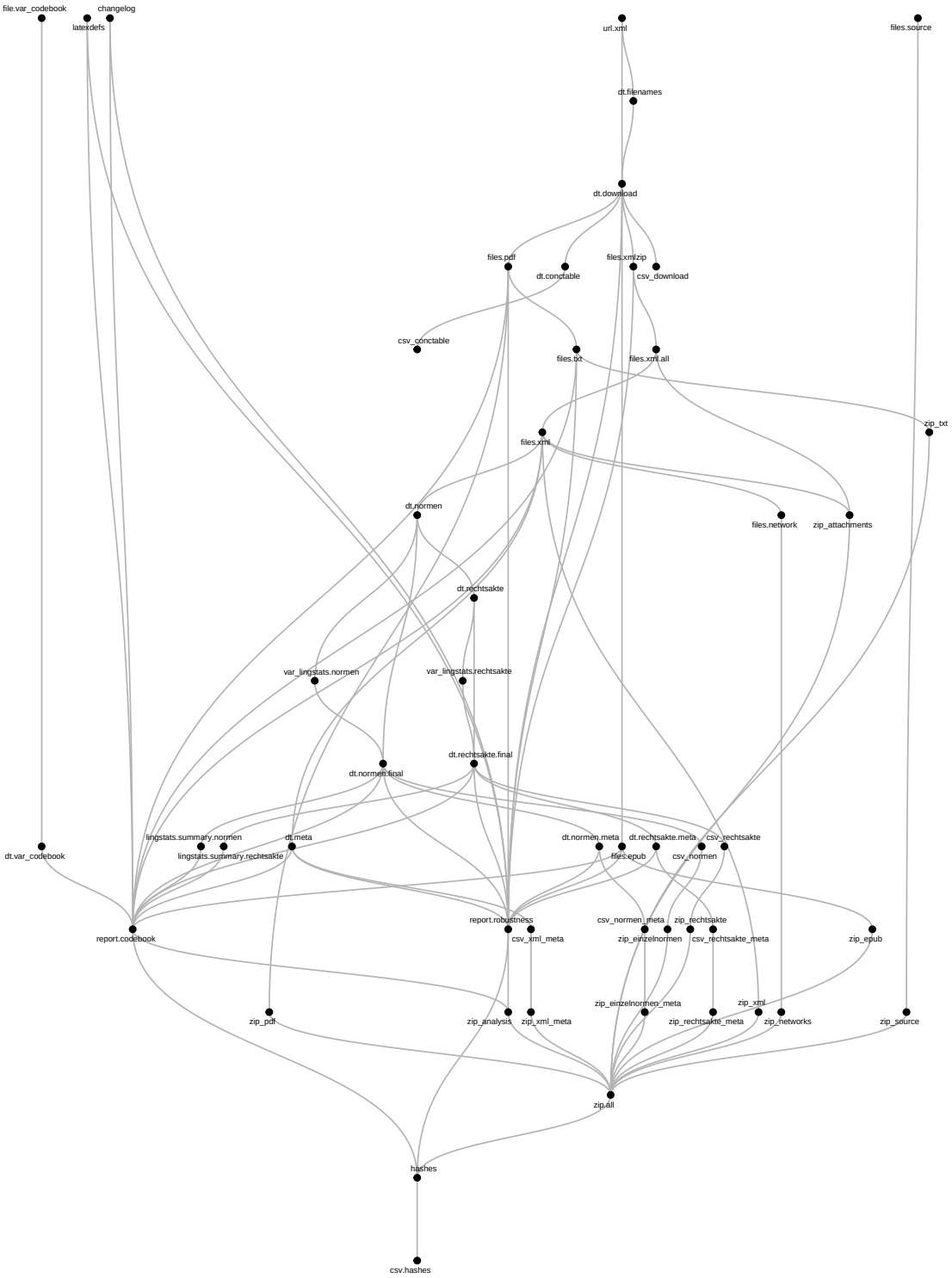
```
zip(paste0("output/",
           paste0(config$project$shortname,
                  "_",
                  datestamp),
           "_Targets_Storage.zip"),
    "_targets/")
```

6.3 Visualisierung

```
edgelist <- tar_network(targets_only = TRUE)$edges
setDT(edgelist)

g  <- igraph::graph.data.frame(edgelist,
                               directed = TRUE)


ggraph(g,
       'sugiyama') +
    geom_edge_diagonal(colour = "grey70")+
    geom_node_point(size = 2)+
    geom_node_text(aes(label = name),
                   size = 2,
                   repel = TRUE)+
    theme_void()
#> Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2
    3.4.0.
#> i Please use `linewidth` in the `default_aes` field and elsewhere instead.
```


file.var_codebook
changelog
latexdefs
url.xml
files.source
dt.filenames
dt.download
files.pdf
dt.conctable
files.xml.zip
csv_download
csv_conctable
files.txt
files.xml.all
zip_txt
files.xml
dt.normen
files.network
zip_attachments
dt.rechtsakte
var_lingstats.normen
var_lingstats.rechtsakte
dt.rechtsakte.final
dt.normen.final
lingstats.summary.normen
dt.meta
dt.normen.meta
dt.rechtsakte.meta
csv_rechtsakte
dt.var_codebook
lingstats.summary.rechtsakte
files.epub
csv_normen
report.robustness
csv_normen_meta
zip_rechtsakte
report.codebook
csv_xml_meta
zip_einzelnormen
csv_rechtsakte_meta
zip_epub
zip_einzelnormen_meta
zip_xml
zip_pdf
zip_analysis
zip_xml_meta
zip_rechtsakte_meta
zip_networks
zip_source
zip.all
hashes
csv.hashes

7 Pipeline: Analyse

7.1 Gesamte Liste

Die vollständige Liste aller Targets, inklusive ihres Types und ihrer Größe. Targets die auf Dateien verweisen (z.B. alle PDF-Dateien) geben die Gesamtgröße der Dateien auf der Festplatte an.

```
meta <- tar_meta(fields = c("type", "bytes", "format"), complete_only = TRUE)
setDT(meta)
meta$MB <- round(meta$bytes / 1e6, digits = 2)

# Gesamter Speicherplatzverbrauch
sum(meta$MB, na.rm = TRUE)
#> [1] 4886.33

kable(meta[order(type, name)],
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

name	type	bytes	format	MB
changelog	stem	5371	file	0.01
csv.hashes	stem	90	qs	0.00
csv_conctable	stem	1002023	file	1.00
csv_download	stem	6359282	file	6.36
csv_normen	stem	218065749	file	218.07
csv_normen_meta	stem	65279974	file	65.28
csv_rechtsakte	stem	155062983	file	155.06
csv_rechtsakte_meta	stem	2472558	file	2.47
csv_xml_meta	stem	3023315	file	3.02
dt.conctable	stem	235282	qs	0.24
dt.download	stem	1333748	qs	1.33
dt.filenames	stem	74108	qs	0.07
dt.meta	stem	473805	qs	0.47
dt.normen	stem	37339078	qs	37.34
dt.normen.final	stem	37771181	qs	37.77
dt.normen.meta	stem	2053044	qs	2.05

(continued)

name	type	bytes	format	MB
dt.rechtsakte	stem	36397043	qs	36.40
dt.rechtsakte.final	stem	36435442	qs	36.44
dt.rechtsakte.meta	stem	413768	qs	0.41
dt.var_codebook	stem	3615	qs	0.00
file.dtd	stem	8915	file	0.01
file.var_codebook	stem	11497	file	0.01
files.epub	stem	468571724	file	468.57
files.network	stem	78208899	file	78.21
files.pdf	stem	678557308	file	678.56
files.source	stem	967242	file	0.97
files.txt	stem	210823581	file	210.82
files.xml	stem	279397378	file	279.40
files.xml.all	stem	590696000	file	590.70
files.xmlzip	stem	372863365	file	372.86
hashes	stem	2136	qs	0.00
latexdefs	stem	1195	file	0.00
lingstats.summary.normen	stem	385	qs	0.00
lingstats.summary.rechtsakte	stem	390	qs	0.00
report.codebook	stem	662384	file	0.66
report.robustness	stem	524580	file	0.52
url.xml	stem	47601	qs	0.05
var_lingstats.normen	stem	449547	qs	0.45
var_lingstats.rechtsakte	stem	38005	qs	0.04
zip.all	stem	234	qs	0.00
zip_analysis	stem	4702872	file	4.70
zip_attachments	stem	281550400	file	281.55
zip_einzelnormen	stem	40274132	file	40.27
zip_einzelnormen_meta	stem	2963103	file	2.96
zip_epub	stem	454189754	file	454.19

(continued)

name	type	bytes	format	MB
zip_networks	stem	72576694	file	72.58
zip_pdf	stem	605230453	file	605.23
zip_rechtsakte	stem	36589708	file	36.59
zip_rechtsakte_meta	stem	501078	file	0.50
zip_source	stem	686760	file	0.69
zip_txt	stem	49510032	file	49.51
zip_xml	stem	51387251	file	51.39
zip_xml_meta	stem	552766	file	0.55

7.2 Timing

7.2.1 Gesamte Laufzeit

```
meta <- tar_meta(fields = c("time", "seconds"), complete_only = TRUE)
setDT(meta)
meta$mins <- round(meta$seconds / 60, digits = 2)

runtime.sum <- sum(meta$seconds)

## Sekunden
print(runtime.sum)
#> [1] 1894.045

## Minuten
runtime.sum / 60
#> [1] 31.56742

## Stunden
runtime.sum / 3600
#> [1] 0.5261236
```

7.2.2 Laufzeit einzelner Targets

Der Zeitpunkt an dem die Targets berechnet wurden und ihre jeweilige Laufzeit in Sekunden.

```
kable(meta[order(-seconds)],
      format = "latex",
      align = "r",
      booktabs = TRUE,
      longtable = TRUE) %>% kable_styling(latex_options = "repeat_header")
```

name	time	seconds	mins
files.network	2023-10-03 12:49:43	743.793	12.40
var_lingstats.normen	2023-10-03 12:55:14	327.745	5.46
dt.normen	2023-10-03 12:36:57	255.973	4.27
var_lingstats.rechtsakte	2023-10-03 12:57:21	122.028	2.03
lingstats.summary.rechtsakte	2023-10-03 12:59:50	111.967	1.87
dt.filenames	2023-10-03 12:30:06	66.461	1.11
dt.meta	2023-10-03 12:32:40	59.448	0.99
lingstats.summary.normen	2023-10-03 12:57:58	34.587	0.58
zip_pdf	2023-10-03 12:31:22	25.002	0.42
zip_epub	2023-10-03 12:30:43	17.406	0.29

(continued)

name	time	seconds	mins
report.robustness	2023-10-03 13:00:32	16.454	0.27
report.codebook	2023-10-03 13:00:16	16.409	0.27
files.txt	2023-10-03 12:30:56	14.203	0.24
zip_attachments	2023-10-03 12:37:09	12.006	0.20
zip_txt	2023-10-03 12:31:40	11.327	0.19
zip_xml	2023-10-03 12:37:20	10.239	0.17
zip_rechtsakte	2023-10-03 13:00:42	9.467	0.16
zip_einzelnormen	2023-10-03 12:59:59	9.225	0.15
files.xml.all	2023-09-29 21:35:06	6.067	0.10
zip_networks	2023-10-03 12:55:19	4.956	0.08
files.epub	2023-10-03 12:30:09	3.551	0.06
files.pdf	2023-10-03 12:30:15	3.205	0.05
hashes	2023-10-03 13:00:46	3.042	0.05
files.xmlzip	2023-10-03 12:30:21	3.028	0.05
dt.rechtsakte	2023-10-03 12:49:46	1.365	0.02
changelog	2023-10-03 12:20:51	1.313	0.02
zip_einzelnormen_meta	2023-10-03 13:00:43	0.990	0.02
dt.download	2023-10-03 12:30:07	0.726	0.01
zip_analysis	2023-10-03 13:00:43	0.521	0.01
url.xml	2023-10-03 12:28:59	0.339	0.01
file.dtd	2023-10-03 12:28:59	0.312	0.01
csv_normen	2023-10-03 12:57:23	0.183	0.00
dt.normen.final	2023-10-03 12:57:22	0.159	0.00
zip_xml_meta	2023-10-03 12:55:19	0.126	0.00
zip_rechtsakte_meta	2023-10-03 13:00:43	0.105	0.00
csv_rechtsakte	2023-10-03 12:59:50	0.082	0.00
dt.rechtsakte.final	2023-10-03 12:57:23	0.070	0.00
zip_source	2023-10-03 12:30:06	0.055	0.00
csv_normen_meta	2023-10-03 12:59:59	0.037	0.00

(continued)

name	time	seconds	mins
dt.normen.meta	2023-10-03 12:57:58	0.018	0.00
latexdefs	2023-10-03 12:28:59	0.016	0.00
csv_download	2023-10-03 12:30:25	0.008	0.00
dt.conctable	2023-10-03 12:30:25	0.007	0.00
csv_xml_meta	2023-10-03 12:49:44	0.006	0.00
csv_rechtsakte_meta	2023-10-03 13:00:16	0.005	0.00
csv_conctable	2023-10-03 12:31:29	0.004	0.00
files.xml	2023-09-29 21:35:06	0.004	0.00
dt.rechtsakte.meta	2023-10-03 12:59:50	0.002	0.00
dt.var_codebook	2023-07-09 17:26:24	0.002	0.00
csv.hashes	2023-10-03 13:00:46	0.001	0.00
file.var_codebook	2023-05-09 12:40:36	0.000	0.00
files.source	2023-10-03 12:20:51	0.000	0.00
zip.all	2023-10-03 13:00:43	0.000	0.00

8 Warnungen

```
meta <- tar_meta(fields = "warnings", complete_only = TRUE)
setDT(meta)
meta <- meta[name != "files.network"]
meta$warnings <- gsub("(\\.pdf|\\.html?|\\.txt)|\\.xml", "\\1 \n\n", meta$
    warnings)

if (meta[,.N > 0]){

    for(i in 1:meta[,.N]){

        cat(paste("##", meta[i]$name), "\n\n")
        cat(paste(meta[i]$warnings, "\n\n"))

    }

}else{

    cat("No warnings to report.")

}
```

8.1 report.codebook

Package microtype Warning Unable to apply patch footnote on input line 210.

8.2 report.robustness

Package microtype Warning Unable to apply patch footnote on input line 220.

9 Fehlermeldungen

```
meta <- tar_meta(fields = "error", complete_only = TRUE)
setDT(meta)

if (meta[,.N > 0]){

    for(i in 1:meta[,.N]){

        cat(paste("##", meta[i]$name), "\n\n")
        cat(paste(meta[i]$error, "\n\n"))

    }

}else{

    cat("No errors to report.")

}
#> No errors to report.
```

10 Dateigrößen der Endergebnisse

10.1 ZIP-Dateien

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

filesize <- round(file.size(files) / 10^6, digits = 2)

table.size <- data.table(basename(files),
                         filesize)


kable(table.size,
      format = "latex",
      align = c("l", "r"),
      format.args = list(big.mark = ","),
      booktabs = TRUE,
      longtable = TRUE,
      col.names = c("Datei",
                    "Größe in MB"))
```

Datei	Größe in MB
C-DBR_2023-10-03_DE_Analyse.zip	4.70
C-DBR_2023-10-03_DE_CSV_Einzelnormen_Datensatz.zip	40.27
C-DBR_2023-10-03_DE_CSV_Einzelnormen_Metadaten.zip	2.96
C-DBR_2023-10-03_DE_CSV_Metadaten-XML.zip	0.55
C-DBR_2023-10-03_DE_CSV_Rechtsakte_Datensatz.zip	36.59
C-DBR_2023-10-03_DE_CSV_Rechtsakte_Metadaten.zip	0.50
C-DBR_2023-10-03_DE_EPUB_Datensatz.zip	454.19
C-DBR_2023-10-03_DE_Netzwerke.zip	72.58
C-DBR_2023-10-03_DE_PDF_Datensatz.zip	605.23
C-DBR_2023-10-03_DE_TXT_Datensatz.zip	49.51
C-DBR_2023-10-03_DE_XML_Anlagen.zip	281.55
C-DBR_2023-10-03_DE_XML_Datensatz.zip	51.39
C-DBR_2023-10-03_Source_Code.zip	0.69
C-DBR_2023-10-03_Targets_Storage.zip	153.95

10.2 CSV-Dateien

```
files <- list.files("output", pattern = "\\.csv", full.names = TRUE)

filesize <- round(file.size(files) / 10^6, digits = 2)

table.size <- data.table(basename(files),
                         filesize)


kable(table.size,
      format = "latex",
      align = c("l", "r"),
      format.args = list(big.mark = ","),
      booktabs = TRUE,
      longtable = TRUE,
      col.names = c("Datei",
                    "Größe in MB"))
```

Datei	Größe in MB
C-DBR_2023-10-03_DE_Alle-Rechtsakte-Verzeichnis.csv	1.00
C-DBR_2023-10-03_DE_CSV_Einzelnormen_Datensatz.csv	218.07
C-DBR_2023-10-03_DE_CSV_Einzelnormen_Metadaten.csv	65.28
C-DBR_2023-10-03_DE_CSV_Metadaten-XML_Datensatz.csv	3.02
C-DBR_2023-10-03_DE_CSV_Rechtsakte_Datensatz.csv	155.06
C-DBR_2023-10-03_DE_CSV_Rechtsakte_Metadaten.csv	2.47
C-DBR_2023-10-03_KryptographischeHashes.csv	0.00

11 Kryptographische Signaturen

11.1 Signaturen laden

```
tar_load(hashes)
```

11.2 Leerzeichen hinzufügen um bei SHA3-512 Zeilenumbruch zu ermöglichen

Hierbei handelt es sich lediglich um eine optische Notwendigkeit. Die normale 128 Zeichen lange Zeichenfolge von SHA3-512-Signaturen wird ansonsten nicht umgebrochen und verschwindet über die Seitengrenze. Das Leerzeichen erlaubt den automatischen Zeilenumbruch und damit einen für Menschen sinnvoll lesbaren Abdruck im Codebook. Diese Variante wird nur zur Anzeige verwendet und danach verworfen.

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

11.3 In Bericht anzeigen

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

index	filename
1	output/C-DBR_2023-10-03_Source_Code.zip
2	output/C-DBR_2023-10-03_DE_Analyse.zip
3	output/C-DBR_2023-10-03_DE_Netzwerke.zip
4	output/C-DBR_2023-10-03_DE_PDF_Datensatz.zip
5	output/C-DBR_2023-10-03_DE_TXT_Datensatz.zip
6	output/C-DBR_2023-10-03_DE_EPUB_Datensatz.zip
7	output/C-DBR_2023-10-03_DE_XML_Datensatz.zip
8	output/C-DBR_2023-10-03_DE_XML_Anlagen.zip
9	output/C-DBR_2023-10-03_DE_CSV_Einzelnormen_Datensatz.zip
10	output/C-DBR_2023-10-03_DE_CSV_Einzelnormen_Metadaten.zip

11	output/C-DBR_2023-10-03_DE_CSV_Rechtsakte_Datensatz.zip
12	output/C-DBR_2023-10-03_DE_CSV_Rechtsakte_Metadaten.zip
13	output/C-DBR_2023-10-03_DE_CSV_Metadaten-XML.zip
14	output/C-DBR_2023-10-03_Codebook.pdf
15	output/C-DBR_2023-10-03_RobustnessChecks.pdf

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

index	sha2.256
1	d91074eefe0e9a442c92687a4fd8a9774d6c2f080d543be7257a85f785f73f5f
2	dc974c4ddb839e0f63967e166ce0a4f543edb8b291bc7ca1cbdb2a9cc22e351c
3	309fc1ed54946ff35277c373622d2bc223078cb37641242c0809f1d79f79368f
4	41cf4b6db567e9c897e8ec75413a465863ef9cd0570af522c235005fa2725132
5	b951e463c57acedc630b91358ff2e051c35f09b42fb76ac3fd4a749811c35575
6	334981e8f444bb956c1f15fb81e4e0af573f118a097005cd0e88d93d078051b1
7	baecf9805134c4b5be25b1f0dab4d23330b1eb061e7a9d1082b88fc0d9457f7f
8	9c1f9493b6f2043ae6333328136a357278917cfb01aee41fb24f7186b19dde42
9	ee3c18e06c9ec2483dc6f93479a0cd10d369f120a56c78643de05d10d442153f
10	f9ac8723af5084ab18b2c1adbc5f66fb06f30543bb754f095d4fc450bd058645
11	8a6ee56bf2f1b79946c45d41ae3bca1eaadcd4ca3438e8ec986ec1d9a4041eb5
12	e3a2ee0d183851d7dde385f4b5a93ed8d10ae7deee54b857b079c433c5fab21f
13	4fcc97f56ea4512e87a45076ecfafea407fe23f4f5eb13821b58948705d67a6f
14	ac6f0148e3d06190dbed499b282e14bd85e8e68671e4c86bd0da1a9c7d6f67b1
15	5bfed0088653c3a41209a9d5eeadfd1e479a2f366ceee44106c961a64d01c745

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

index	sha3.512
1	cfc72c8b533552a9148727c7e443e58929a877a04e2fc8f0653a4069fde1a881 c5a51e1c87463e9db2302baab76c32d44b78c2174f1799a0983da4eed155e14a
2	6ee3ac07a5391b296708e962eb70d97805f247bd882c141e4b8bcd9b0855de53 e49dd095eba3d642d547cda7337cec8f902ce5f5ee2f32c1420e0c0adb74d5f3
3	fd987e66f667734d6928f81969f95e4b35fede264472475811d995a0ec2defc4 ca9322bfa447dc6c07f7ac414440d6506806160d2427b63cd15060c0d0602cd3
4	85f562c541f247db6f97f32edf5e41ceb4b50c48c08e773a0de057dab9262231 76ba-ba3669220f0a0208cb0c90c434323f0031ab16912e0367e96d1ed9a40dad
5	7833b56a8425fe6da31b1f3189865611d91ca80c084d82ef9194be2f294b21c8 895b4f0238d16153f2172752b1c85afd75d482f31556894017e20b6ec26fb79d
6	87f1b3add5721f929ecc08d75a0677689f98ae1934a9bc611988eef87d1eaa66 e3075224493451438b62541ec3efe6305f7c108d4f836eec960472cc501b6ba6
7	2ef28d9fcb4072ce5535245eb31be8fe379712373930efc97ebb6b009fe3b8c4 de02f03e19615799f363f61aa0e7fee0be1a139759215a23f6b31f7a33fdb99e
8	80f3ef8538c16f754d55491b5a6918ef503f0e4071dbe0c3688655cac2df959e e3d3d06dde1036f3ac9c4b4249b4b1dc344575ce68f527dedec88373835121c4
9	a213345cb6414a618058ff293c54bca5a8f999e8ca07aa2aea48fcb59321c258 5ab488851f856360b41ac0d1d79ae69e527c810c525aa46b5c3f94b55660f46d
10	e25c9ceb9c536600e3beb7ab2f3f628ca254a2cd45d754e4a3b009463962ad42 d77f53868e6781d3bef54543243a9896c83691eaffc8cdab81dad69a87c17a87
11	592940ec4374d527f4a2c653b303b38b3eef55929303e1765130d1f96107c682 eb69b0ba4986852fe9dbd8c1b7cbb1097e14fda305ebd74f5d0ba3863e0a5c19
12	6af62dffedf0c24361ac2c61b6f539385d16d0a4b1f0390a9e36161ffcda2401 4e02dad8633f06275b8e794f648da77f41ef4ced22430dbfc34b8e92648b1a6f
13	b74c46db7ebefc0784e1a2570b6074be252b91b0d22a0100f4ac0dfe5717ce48 7fe3af13579f949e8bc483025e1e995af012bbe5495507935f35d4e32f4f2af3
14	bfd167a0cc6241c37adb6a2d511e4b367459dd3b5ee4de7b6a681f4b89e71d9c 5c951d0b3f22e7cb52d8f6e6d6b61df171152829d073942b404eef84bbec0a88
15	471b27ae5cdf0d227773964c7178fb6496ef793f0f1b730426faac9d27b56bbc f62244e09d4ce2af358e7b7e1bf2838b9f27b7cd91db6c50507c75e611dde873

12 Changelog

12.1 Version 2023-10-03

- Vollständige Aktualisierung der Daten

12.2 Version 2023-07-09

- Vollständige Aktualisierung der Daten
- Konfigurations-Dateien in etc/ verschoben

12.3 Version 2023-04-07

- Vollständige Aktualisierung der Daten
- Gesamte Laufzeitumgebung nun mit Docker versionskontrolliert
- Download-Manifest wird nun spätestens nach 24h Stunden invalidiert, damit keine alten Daten aus früheren Kompilierungen den Prozess zum Absturz bringen
- ZIP-Archiv der TXT-Dateien wird nun auch gehasht
- Verbesserte Formatierung von Warnungen und Fehlermeldungen im Compilation Report
- Veränderung der Download-Reihenfolge
- Falls im ersten Download-Durchlauf Dateien fehlen werden die Folgeversuche nun korrekt durchgeführt
- Die Pipeline mit allen Zwischenergebnissen wird nun automatisch in "output/" archiviert
- Source Code ZIP-Archiv wird nun anhand des git-Manifestes generiert
- README im Hinblick auf Docker aktualisiert
- Struktur des Compilation Reports angepasst, um Warnungen und Fehler prominenter anzuzeigen
- Zusätzliche Unit Tests

12.4 Version 2023-01-05

- Vollständige Aktualisierung der Daten
- Neuer Entwurf des gesamten Source Codes im {targets} Framework
- Zusätzliche Netzwerk-Diagramme für alle Rechtsakte: Sunburst und Circlepacking
- Reguläre Netzwerk-Diagramme nun in blau auf schwarzem Hintergrund
- Manche finale Dateinamen nun mit Trennstrichen statt Pascal Case
- TXT-Konvertierung bricht bei Fehler nicht ab, dokumentiert aber fehlende TXT-Dateien
- Einführung eines separaten Berichts für Robustness Checks

12.5 Version 2022-08-05

- Vollständige Aktualisierung der Daten
- Wenn der Download einer Datei scheitert wird der Kompilierungs-Prozess nicht mehr abgebrochen; Kontrolle über Datenabgleich im Compilation Report
- Diagramme für Norm/Rechtsakt/Metadaten je Periodikum sind nun logarithmisch skaliert

- Technischer Bugfix bei der Berechnung von Netzwerkdiagrammen
- Neuer Unit Test um identische Länge von HTML-Links und extrahierten PDF- und EPUB-Dateinamen
- Fehlende PDF- oder EPUB-Dateien führen nun nicht mehr zu Fehlern in der Pipeline
- Unterscheidung zwischen VBVG 2005 und VBVG 2023

12.6 Version 2022-05-22

- Vollständige Aktualisierung der Daten
- README und CHANGELOG sind nun externe Dateien die bei der Kompilierung automatisch eingebunden werden
- Das für *renv* notwendige Skript activate.R ist im ZIP-Archiv in den Ordner "renv" sortiert

12.7 Version 2022-01-12

- Vollständige Aktualisierung der Daten
- Strenge Versionskontrolle aller R packages
- Der Prozess der Kompilierung ist jetzt detailliert konfigurierbar, insbesondere die Parallelisierung
- Parallelisierung der XML-Parser deaktivert, weil instabil
- Parallelisierung nun vollständig mit *future* statt mit *foreach* und *doParallel*
- Fehlerhafte Kompilierungen werden beim vor der nächsten Kompilierung vollautomatisch aufgeräumt
- Alle Ergebnisse werden automatisch fertig verpackt in den Ordner »output« sortiert
- Source Code des Changelogs zu Markdown konvertiert
- Einführung eines Debugging-Modus um die Entwicklung zu beschleunigen

12.8 Version 2021-09-16

- Vollständige Aktualisierung der Daten
- Einfügung von Kurzbezeichnungen der Rechtsakte in die Dateinamen der Netzwerkanalysen
- Einfügung der ID der Rechtsakte in die CSV-Tabelle aller Kurz- und Langtitel

12.9 Version 2021-07-30

- Vollständige Aktualisierung der Daten
- Einführung von neuen Variablen für letzte Änderung (Datum), Neufassung (Datum), Aufhebung (Datum jeweils für Verkündung und Wirkung), Lizenz und hierarchische Ketten von Gliederungsbezeichnungen und -titeln
- Parallelisierung der Downloads um Kompilierung des Korpus zu beschleunigen
- Korrektur bei den Dateinamen der Allgemeinen Eisenbahngesetze: GII weist zwei gleichnamige Rechtsakte (»Allgemeines Eisenbahngesetz«) nach. Beide werden nun mit dem Jahr ihrer Ausfertigung 1951 und 1993 im Langtitel differenziert. In der Vorversion wurde das neuere AEG noch mit dem Jahr 1994 (Inkrafttreten) beschriftet und das andere AEG ohne Jahreszahl.
- Einführung von Netzwerkanalysen (experimentell!)
- Variablen in CSV-Dateien sind nun semantisch sortiert

- Neues Diagramm für Verteilung von Zeichen
- Falls die XML-Datei mehrere Bemerkungen für Hinweise, Änderung, Neufassung, den Stand oder sonstige Angaben aufweist werden diese nun durch einen vertikalen Strich getrennt (vorher nur mehrere Leerzeichen).
- Kleinere Korrekturen und Ergänzungen im Codebook

12.10 Version 2021-01-05

- Vollständige Aktualisierung der Daten
- Komplette Überarbeitung des Source Codes
- Erstveröffentlichung eines Codebooks
- Einführung der vollautomatischen Erstellung von Datensatz und Codebook
- Einführung von Compilation Reports um den Erstellungsprozess exakt zu dokumentieren
- CSV-Dateien werden durch Parsing der XML-Dateien erstellt
- Automatisierung und deutliche Erweiterung der Qualitätskontrolle
- Einführung von Diagrammen zur Visualisierung von Prüfergebnissen
- Einführung kryptographischer Signaturen

12.11 Version 2020-10-09

- Vollständige Aktualisierung der Daten
- Erstveröffentlichung des Source Codes
- XML-Daten nun fehlerfrei. In Version 2020-07-08 waren XML-Dateien mit Anhängen fehlerhaft.

12.12 Version 2020-07-08

- Vollständige Aktualisierung der Daten

12.13 Version 2020-05-18

- Erstveröffentlichung

13 Abschluss

```
## Datumsstempel
print(datestamp)
#> [1] "2023-10-03"

## Datum und Uhrzeit (Anfang)
print(begin.script)
#> [1] "2023-10-03 12:28:55 CEST"


## Datum und Uhrzeit (Ende)
end.script <- Sys.time()
print(end.script)
#> [1] "2023-10-03 13:00:55 CEST"


## Laufzeit des gesamten Skriptes
print(end.script - begin.script)
#> Time difference of 31.99528 mins
```

14 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] ggraph_2.1.0      ggplot2_3.4.1     igraph_1.4.1      kableExtra_1.3.4
#>  [5] knitr_1.42        quanteda_3.2.4    data.table_1.14.8 future_1.32.0
#>  [9] RcppTOML_0.2.2    tarchetypes_0.7.5 targets_0.14.3
#>
#> loaded via a namespace (and not attached):
#>  [1] viridis_0.6.2       httr_1.4.5          tidyr_1.3.0
#>  [4] tidygraph_1.2.3     viridisLite_0.4.1   RcppParallel_5.1.7
#>  [7] highr_0.10          future.callr_0.8.1  base64url_1.4
#> [10] renv_0.17.0         yaml_2.3.7          ggrepel_0.9.3
#> [13] globals_0.16.2      pillar_1.8.1        backports_1.4.1
#> [16] lattice_0.20-45     glue_1.6.2          digest_0.6.31
#> [19] polyclip_1.10-4     rvest_1.0.3         stringfish_0.15.7
#> [22] colorspace_2.1-0    htmltools_0.5.4     Matrix_1.5-1
#> [25] pkgconfig_2.0.3     listenv_0.9.0       purrr_1.0.1
#> [28] scales_1.2.1        webshot_0.5.4       processx_3.8.0
#> [31] svglite_2.1.1       tweenr_2.0.2        RApiSerialize_0.1.2
#> [34] ggforce_0.4.1       tibble_3.2.0        generics_0.1.3
#> [37] farver_2.1.1        withr_2.5.0         furrr_0.3.1
#> [40] cli_3.6.0           magrittr_2.0.3      evaluate_0.20
#> [43] ps_1.7.2            stopwords_2.3       fs_1.6.1
#> [46] fansi_1.0.4         parallelly_1.34.0   MASS_7.3-58.1
#> [49] xml2_1.3.3          tools_4.2.2         lifecycle_1.0.3
#> [52] stringr_1.5.0       munsell_0.5.0       callr_3.7.3
#> [55] compiler_4.2.2      qs_0.25.5           systemfonts_1.0.4
#> [58] rlang_1.0.6         grid_4.2.2          rstudioapi_0.14
#> [61] labeling_0.4.2      rmarkdown_2.20      gtable_0.3.1
#> [64] codetools_0.2-18    graphlayouts_0.8.4  R6_2.5.1
#> [67] gridExtra_2.3       dplyr_1.1.0         fastmap_1.1.1
#> [70] utf8_1.2.3          fastmatch_1.1-3     stringi_1.7.12
```

```
#> [73] parallel_4.2.2      Rcpp_1.0.10         vctrs_0.5.2
#> [76] tidyselect_1.2.0    xfun_0.37
```

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, and Adam Obeng. 2021. *Readtext: Import and Handling for Plain and Formatted Text Files.* https://github.com/quanteda/readtext.

Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. 2018. "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.

Csardi, Gabor, and Tamas Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal* Complex Systems: 1695. https://igraph.org.

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.

file., See AUTHORS. 2023. *Igraph: Network Analysis and Visualization.*

Landau, William Michael. 2021a. *Tarchetypes: Archetypes for Targets.*

———. 2021b. "The Targets R Package: A Dynamic Make-Like Function-Oriented Pipeline Toolkit for Reproducibility and High-Performance Computing." *Journal of Open Source Software* 6 (57): 2959. https://doi.org/10.21105/joss.02959.

———. 2023a. *Tarchetypes: Archetypes for Targets.*

———. 2023b. *Targets: Dynamic Function-Oriented Make-Like Declarative Pipelines.*

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

———. 2023b. *Openssl: Toolkit for Encryption, Signatures and Certificates Based on Openssl.* https://github.com/jeroen/openssl.

———. 2023c. *Pdftools: Text Extraction, Rendering and Converting of Pdf Documents.*

Pedersen, Thomas Lin. 2022. *Ggraph: An Implementation of Grammar of Graphics for Graphs and Networks.*

Ushey, Kevin. 2023. *Renv: Project Environments.* https://rstudio.github.io/renv/.

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

———. 2022. *Rvest: Easily Harvest (Scrape) Web Pages*.

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*.

Wickham, Hadley, Jim Hester, and Jeroen Ooms. 2021. *Xml2: Parse Xml*.

Wickham, Hadley, and Dana Seidel. 2022. *Scales: Scale Functions for Visualization*.

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*.