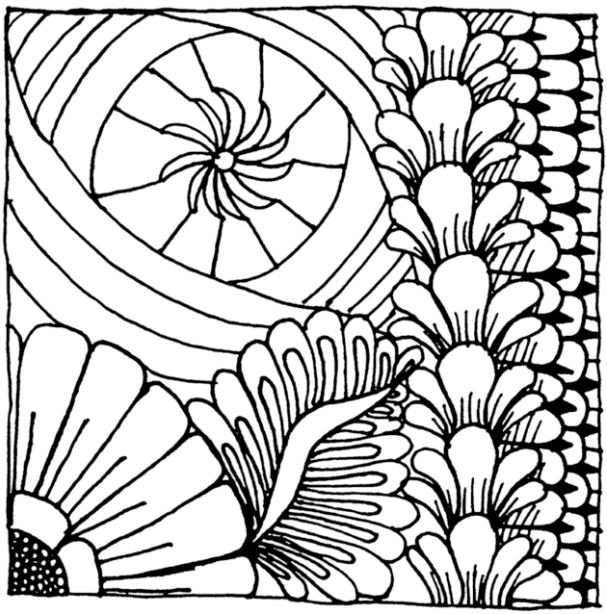
# Einleitung und Überblick

1





J. Jünger, C. Gärtner, *Computational Methods für die Sozial- und Geisteswissenschaften*, https://doi.org/10.1007/978-3-658-37747-2_1

**Zusammenfassung**

Dieses Kapitel führt Sie in die Welt der Computational Methods ein. Sie erfahren, an wen sich das Buch richtet, was unter Computational Methods verstanden wird und wofür diese in der Wissenschaft eingesetzt werden. Zudem lernen Sie, welche grundlegenden Tools für die Arbeit mit diesen Methoden benötigt werden.

Im Online-Repositorium unter https://github.com/strohne/cm finden Sie begleitend zum Kapitel weitere Materialien, auf die wir im Text mit ☛ verweisen.

**Schlüsselwörter**

Computational Methods · Computational Social Science · Digital Humanities · Kommandozeile · Texteditor

Der Bereich Computational Methods hat für die Sozial- und Geisteswissenschaften zunehmend an Bedeutung gewonnen. Das kann unter anderem darauf zurückgeführt werden, dass Kulturgüter wie historische Bücher, Daten von Organisationen oder auch die Kommunikation in journalistischen Medien in den letzten Jahrzehnten digital verfügbar wurden. Nicht nur das: Viele soziale, auch interpersonale, Prozesse laufen mittlerweile zum Beispiel auf Online-Plattformen direkt in digitalen Umgebungen ab. Computational Methods helfen dabei, umfangreiche und heterogene Datenbestände zusammenzustellen und automatisiert auszuwerten. Daraus ergeben sich auch weitere wissenschaftliche Fragestellungen und Herangehensweisen, indem beispielsweise menschliches Verhalten mit Simulationsmodellen nachgestellt wird, um aus dem Vergleich mit der sozialen Wirklichkeit grundlegende Mechanismen herauszuarbeiten. Besonders reizvoll ist dabei, dass gleichzeitig eine Vogelperspektive und eine Froschperspektive eingenommen werden können. So kann beispielsweise mit Netzwerkanalysen die Struktur hinter dem Zusammenspiel einer Vielzahl an Akteuren[1] visualisiert werden, ohne dass die einzelnen Akteure aus dem Blick geraten müssen.

Anzeichen für die zunehmende Bedeutung von Computational Methods sind auch die Gründung von spezialisierten Fachzeitschriften, Forschungseinrichtungen

[1] Sofern wir davon ausgehen, dass es sich um komplexe nichtmenschliche Akteure handelt (korporative oder kollektive Akteure sowie technische Rollen), gendern wir in diesem Buch Begriffe wie Akteur oder Anbieter nicht, auch um eine Anthropomorphisierung zu vermeiden.

und Lehrstühlen sowie Fachvertretungen. In der Kommunikationswissenschaft wird dies beispielsweise an der Gründung einer Computational Methods Division innerhalb der internationalen Fachvertretung International Communication Association (ICA) deutlich. Ausgangspunkt war die Idee, dass Computational Methods eine wichtige Rolle für die Weiterentwicklung kommunikationswissenschaftlicher Forschung spielen werden: „We believe that computational methods will be at the forefront of progress in the field in the coming decades“ (ICA CM 2017). Damit aber Computational Methods im Rahmen substanzieller Forschung eingesetzt werden können, braucht es Wissenschaftler:innen, die mit diesen Methoden umgehen können.

Das vorliegende Buch soll (bisherige) technische Laien in die Lage versetzen, selbstständig automatisierte Verfahren der Datenerhebung und Datenauswertung anzuwenden. Es ist begleitend zu Lehrveranstaltungen entstanden, in denen für Studierende der Sozial- und Geisteswissenschaften der Weg in eine zunächst fremde, dann aber sehr spannende Welt eröffnet wird. Dabei geht es darum, diese digitale Welt nicht nur besser zu verstehen, sondern auch mitgestalten zu können.

Im ersten Teil des Buches wird grundlegendes Wissen über Datenquellen, Datenformate und Verfahren zur Aufbereitung von Daten vermittelt. Im zweiten Teil finden sich kurze Einführungen in die zwei einschlägigen Programmiersprachen R und Python sowie in Konzepte zum Verwalten größerer Programmierprojekte. Im dritten Teil werden spezielle Erhebungs- und Auswertungsverfahren vorgestellt, beispielsweise Webscraping, Textanalyse und Klassifikationsverfahren. Innerhalb der Kapitel wird neben einem Überblick über die Verfahren jeweils eine Anleitung für ein ausgewähltes Beispiel gegeben. Die Beispiele beziehen sich häufig auf kommunikationswissenschaftlich relevante Bereiche, verbunden mit der Hoffnung, dass die Konzepte und Anleitungen anschlussfähig für sehr unterschiedliche Felder sind. Denn die Kommunikationswissenschaft ist eine integrative Disziplin, in der sowohl sozial- als auch geisteswissenschaftliches Denken zusammenkommt.

Die vorliegende Einführung gibt einen ersten Einblick in die vorgestellten Bereiche, kann aber nicht die vielen, sehr guten Einführungswerke in spezialisierte Verfahren ersetzen. Deshalb sind am Ende jedes Kapitels weiterführende Literaturhinweise und themenspezifische Einführungswerke zusammengestellt. Daneben finden sich nach jedem Kapitel Übungsfragen, mit denen theoretisches und praktisches Wissen überprüft werden kann. Zusätzlich zu den Inhalten in diesem Buch werden Daten und Skripte in einem Repositorium als Begleitmaterialien zur Verfügung gestellt. Wie diese beim Lernen von Computational Methods helfen können, wird in Abschn. 1.2.4 beschrieben.

## 1.1 Was sind Computational Methods?

Ausgehend von der Bezeichnung „Computational Methods“ wären darunter alle Verfahren zu verstehen, bei denen „Computation“ zur wissenschaftlichen Analyse eingesetzt wird. Versteht man unter dem Begriff „Computation“ in direkter Übersetzung allerdings ganz allgemein jede Form des Rechnens, dann umfasst das auch die quantitative Auswertung von Befragungen und Texten und jede Art von Statistik, im einfachsten Fall sogar die Bildung von Summen und Mittelwerten. Auch eine Eingrenzung auf *maschinelles* Rechnen hilft hier nicht weiter, denn Maschinen sind im Verständnis der Informatik nicht zwangsläufig elektronische oder mechanische Geräte. Ein Beispiel für eine rein konzeptionelle Maschine ist die Turing-Maschine, die ein Rechenmodell bzw. die mathematische Version eines Algorithmus darstellt. Zudem werden auch in der klassischen statistischen Analyse Computer eingesetzt. Im Film Hidden Figures (2016) findet sich ein anschauliches Beispiel dafür, dass unter Computern auch Menschen mit einer Begabung für das Rechnen verstanden werden können. Die „Computational Unit“, in der Berechnungen für die Flugbahnen der Apollo-Raumfahrtmission durchgeführt werden, besteht zunächst ausschließlich aus Frauen. Erst am Ende des Films wird dargestellt, wie diese Aufgabe von elektronischen Maschinen übernommen wird.

Es ist aber gerade dieser Aspekt, das Hinausgehen über klassische sozial- und geisteswissenschaftliche Methoden, der als Besonderheit von Computational Methods angesehen wird (siehe auch van Atteveldt und Peng 2018, S. 82). Einigkeit besteht in den Sozialwissenschaften zumindest dahingehend, dass es sich eben nicht um klassische Methoden handelt, sondern um Verfahren aus der Informatik: „Computational social sciences is a research discipline at the interface between computer science and the traditional social sciences. This interdisciplinary and emerging scientific field uses computationally methods to analyze and model social phenomena, social structures, and collective behavior“ (Amaral 2017; siehe auch Cioffi-Revilla 2010, S. 259). Doch auch hier wird schnell klar, dass es sich nicht zwangsläufig um neue Entwicklungen handelt. Im Gegenteil: Es lässt sich mittlerweile bereits eine historische Perspektive auf diesen Forschungsbereich einnehmen (Cioffi-Revilla 2017, S. 18 ff.). Bereits in den 1960er-Jahren finden sich Publikationen, in denen diskutiert wird, inwiefern computerbasierte Methoden für sozialwissenschaftliche Fragestellungen nutzbar sind (Coleman 1964).

Im Zusammenhang mit Computational Methods haben sich in verschiedenen Disziplinen Bezeichnungen herausgebildet, mit denen die Verbindung zu compu-

terbasierten Methoden angezeigt wird (Tab. 1.1; siehe auch Welker 2019). Ganz allgemein wird in den Sozialwissenschaften von Computational Social Science (CSS) und in den Geisteswissenschaften von Digital Humanities (DH) gesprochen[2]. Daneben finden sich fachspezifische Bezeichnungen wie Digital Sociology, Computational Communication Science, Digital History oder Computational Linguistics. Die englischen Bezeichnungen machen deutlich, dass es sich um globale Entwicklungen handelt.

**Tab. 1.1** Forschungsrichtungen, in denen Computational Methods eingesetzt werden

| Begriff | Definition (Beispiel) |
|---|---|
| Computational Social Science (CSS) | „The new field of Computational Social Science can be defined as the interdisciplinary investigation of the social universe on many scales, ranging from individual actors to the largest groupings, through the medium of computation" (Cioffi-Revilla 2017, S. 2). |
| Digital Sociology | „The ‚digital' in digital sociology may denote at least three different things: it may refer to (1) the *topics* of social enquiry; (2) the *instruments and methods* of social research; (3) the *platforms* for engaging with the audiences and publics of sociology. Depending on which of these aspects of the ‚digital' we consider, we arrive at a very different understanding of what digital sociology is" (Marres 2017, S. 24). |
| Computational Communication Science (CCS) | „Computational communication science studies generally involve: (1) large and complex data sets; (2) consisting of digital traces and other ‚naturally occurring' data; (3) requiring algorithmic solutions to analyze; and (4) allowing the study of human communication by applying and testing communication theory" (van Atteveldt und Peng 2018, S. 82; in Anlehnung an Shah et al. 2015). |
| Digital Humanities (DH) | „‚Digital Humanities' covers a great number of activities in research, teaching, and cultural production, using computers and web-based platforms as new technologies for both scholars and the general public. New modes of digital collaboration and sharing are re-defining the lines between academic and cultural production" (DHI 2020).<br>„[D]ie Summe aller Versuche, die Informationstechniken auf den Gegenstandsbereich der Geisteswissenschaften anzuwenden" (Jannidis et al. 2017, S. 13). |

(Fortsetzung)

[2] Zur Bedeutung des Begriffs Digital Humanities gibt es eine umfangreiche fachliche Debatte, siehe die Beiträge in Terras et al. (2013).

**Tab. 1.1** (Fortsetzung)

| Begriff | Definition (Beispiel) |
|---|---|
| Humanities Computing | „Humanities computing is precisely the automation of every possible analysis of human expression (therefore, it is exquisitely a ‚humanistic' activity), in the widest sense of the word, from music to the theater, from design and painting to phonetics, but whose nucleus remains the discourse of written texts" (Busa 2004, S. xvi). |
| Digital History | „[D]igital history is the process by which historians are able to use computers to do history in ways impossible without the computer" (Burton 2005, S. 207).<br>„The most important weapon for building the digital future we want is to take an active hand in creating digital history in the present" (Cohen und Rosenzweig 2006). |
| Computational Linguistics | „Computational linguistics is the scientific and engineering discipline concerned with understanding written and spoken language from a computational perspective, and building artifacts that usefully process and produce language, either in bulk or in a dialogue setting" (Schubert 2020). |
| Digital Methods | „[T]he repurposing of methods in media for social and cultural research" (Rogers 2010, S. 243). |
| Webometrics | „Webometrics […] covers research of all network-based communication using informetric or other quantitative measures" (Almind und Ingwersen 1997, S. 404).<br>„The study of the quantitative aspects of the construction and use of information resources, structures and technologies on the Web drawing on bibliometric and informetric approaches" (Björneborn 2004, S. 12; siehe auch Björneborn und Ingwersen 2004).<br>„[T]he study of web-based content with primarily quantitative methods for social science research goals using techniques that are not specific to one field of study" (Thelwall 2009, S. 6). |
| Data Mining | „Data mining is the application of specific algorithms for extracting patterns from data" (Fayyad et al. 1996, S. 39). |
| Knowledge Discovery in Databases (KDD) | „KDD is the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" (Fayyad et al. 1996, S. 40). |
| Data Science | „[D]ata science is a new interdisciplinary field that synthesizes and builds on statistics, informatics, computing, communication, management, and sociology to study data and its environments (including domains and other contextual aspects, such as organizational and social aspects) in order to transform data to insights and decisions by following a data-to-knowledge-to-wisdom thinking and methodology" (Cao 2017, S. 8). |

Quelle: eigene Darstellung

Weitere Bezeichnungen stammen stärker aus der Tradition der Informatik und fokussieren spezifische Aspekte der Datenerhebung, wie etwa die Begriffe Data Mining oder Knowledge Discovery in Databases. Diese Begriffe findet man auch außerhalb der und an der Schnittstelle zur Wissenschaft. Als Bezeichnung für in der Wirtschaft verbreitete Tätigkeitsfelder haben sich die Begriffe Data Engineer und Data Scientist herausgebildet, inklusive entsprechender Studiengänge. Während erstere vor allem mit der Entwicklung von Software für die Datenaufbereitung beschäftigt sind, widmen sich letztere der Analyse. Hinzu kommen in größeren Organisationen Data Architects, die vor allem die Infrastruktur und Koordination der Teilaufgaben im Blick haben.

Dass sich in diesem Bereich verschiedene Traditionen verbinden, erkennt man auch daran, dass je nach Perspektive unterschiedliche Terminologien für die gleichen Sachverhalte verwendet werden.[3] Wo in der klassischen Statistik von unabhängigen und abhängigen Variablen die Rede ist, sprechen Data Scientists von Features und Labels. Beide Varianten umfassen die Merkmalsausprägungen der untersuchten Fälle, etwa das Erstelldatum von Kommentaren auf einer Webseite. Dabei geht es jedoch um mehr als nur um alternative Bezeichnungen, häufig unterscheiden sich auch die Zielstellungen. Während aus anwendungsorientierter Sicht vor allem die optimale Vorhersage von Verhalten relevant ist, suchen Wissenschaftler:innen nach Erklärungen. Beide Perspektiven modellieren die Wirklichkeit mit statistischen Mitteln. Erstere streben dabei allerdings eine möglichst hohe Modellgüte an, während für letztere die einzelnen Parameter der Modelle wichtiger sind.[4] Auch innerhalb der Wissenschaft kommen verschiedene erkenntnistheoretische Grundpositionen zusammen. Der Begriff Digital Methods kommt beispielsweise aus einer stärker interpretativen oder sogar ethnografischen Perspektive, wenn darunter „the repurposing of methods in media for social and cultural research“ (Rogers 2010, S. 243) verstanden wird. Damit ist etwa gemeint, dass für die wissenschaftliche Analyse von Online-Kommunikation die Mittel der Online-Kommunikation verwendet werden sollten. So würde man Suchmaschinen untersuchen, indem man Suchmaschinen verwendet. Hinzu kommt vor allem im Bereich der Digital Humanities noch eine ganz praktische Perspektive, wenn darunter auch künstlerisches oder literarisches

[3] Zur unterschiedlichen Terminologie siehe die Einführung von Attewell und Monaghan (2015, S. 8 ff.).

[4] Die unterschiedlichen Perspektiven lassen sich gut am Umgang mit Regressionsmodellen verdeutlichen. Während Data Scientists den Determinationskoeffizienten $R^2$ optimieren und die ohnehin unübersichtliche Anzahl an Variablen bzw. Einflussfaktoren hintenanstellen, sind Sozial- und Geisteswissenschaftler:innen vorrangig an den Beta-Koeffizienten der einzelnen Variablen interessiert und akzeptieren dabei eine geringe Erklärungskraft des Gesamtmodells.

Schaffen verstanden wird, bei dem computergestützt kulturelle Artefakte nicht nur beschrieben oder analysiert, sondern selbst produziert werden.

Die Kombinationen mit dem Wort „digital“ verweisen bereits auf den Gegenstandsbereich digitaler Kommunikation, der allerdings sehr umfassend und damit schwer fassbar ist.[5] Etwas engere Fokussierungen nehmen die Begriffe Internet Research oder noch enger Web Science und Web Mining vor.[6] Noch spezifischer sind Felder wie Webometrics, in denen in der Tradition bibliografischer Zitationsanalysen unter anderem netzwerkanalytische Strukturen zwischen Webseiten im Vordergrund stehen. In der vorliegenden Einführung wird der nicht auf eine Disziplin festgelegte Begriff Computational Methods verwendet, da die Methoden im Vordergrund stehen sollen. Diese Methoden können in kreativer Weise für sehr unterschiedliche Fragestellungen fruchtbar gemacht werden.

Wenn auch die inhaltliche Begriffsbestimmung nicht trivial ist, so sind Computational Methods faszinierend. Ein Teil der Faszination ergibt sich vermutlich daraus, dass Computer (als technische Maschinen) dabei Aufgaben übernehmen, die im ersten Moment menschliche Fähigkeiten voraussetzen. Deutlich wird dies im Bereich der Sprach- und Bilderkennung: Suchmaschinen können Bilder danach sortieren, ob sie Katzen oder Menschen enthalten. Smartphones reagieren auf die verbale Anweisung, einen Timer zu stellen. Und in menschlichen Gesichtern wird versucht, automatisch Emotionen zu erkennen. In diesen Beispielen werden maschinell komplexe, aber sehr spezifische Aufgaben gelöst, die man herkömmlich nur Menschen zutraut. Deshalb wird auch von künstlicher Intelligenz gesprochen, ohne dass dabei tatsächlich Intelligenz im Spiel wäre (Russell und Norvig 2012, S. 22).

Interessant für die wissenschaftliche Analyse werden entsprechende Verfahren unter anderem dadurch, dass sie automatisiert auf einer großen Datenmenge durchgeführt werden können. Hierin liegt ein entscheidender Vorteil gegenüber menschlichen Analysen, wenn eine kleine Stichprobe für eine Fragestellung nicht ausreicht. Zudem versprechen Computational Methods eine hohe Reliabilität – das heißt, sie führen zumindest dem ersten Anschein nach zu den immer gleichen Ergebnissen.[7]

---

[5] Für kritische Anmerkungen zur Verwendung des Digitalisierungsbegriffs siehe Jünger und Schade (2018).

[6] Das Internet und das Web unterscheiden sich sowohl in technischer als auch in organisatorischer Hinsicht (Beck 2006, S. 30), die Begriffe sollten deshalb nicht verwechselt werden.

[7] Tatsächlich ist Reliabilität im Sinne von Reproduzierbarkeit häufig nicht gegeben, da sich die untersuchten Gegenstände und die Methoden schnell wandeln (Jünger 2018, S. 122). Selbst wenn im Sinne von Open Science die Quelltexte von Programmen veröffentlicht werden, lassen sich Skripte nach einigen Jahren aufgrund veränderter technischer (z. B. Betriebssysteme) und organisatorischer (z. B. Plattformen) Rahmenbedingungen häufig nicht mehr ausführen.

Theoretisch besteht die Besonderheit von Automatisierung darin, dass mit einem vergleichsweise hohen Anfangsaufwand ein System aufgesetzt wird, das anschließend ohne menschliche Eingriffe läuft und so den Aufwand reduziert. Automatisierung bedeutet zum Beispiel bei der Analyse von Texten, dass der Zusatzaufwand mit jedem zusätzlichen Dokument sinkt. Aus ökonomischer Sicht sinken die Grenzkosten: „As a rule-of-thumb, we consider a system fully automated if the marginal cost of analyzing additional texts goes to zero as the size of the corpus being analyzed increases, and the coding is completely replicable given a set of software, dictionaries, and so forth“ (Monroe und Schrodt 2008, S. 352; siehe auch Scharkow 2011, S. 547). Automatisierung ist allerdings in der Praxis nicht zwangsläufig effektiv oder effizient (Abb. 1.1). Das Lernen der Verfahren, die

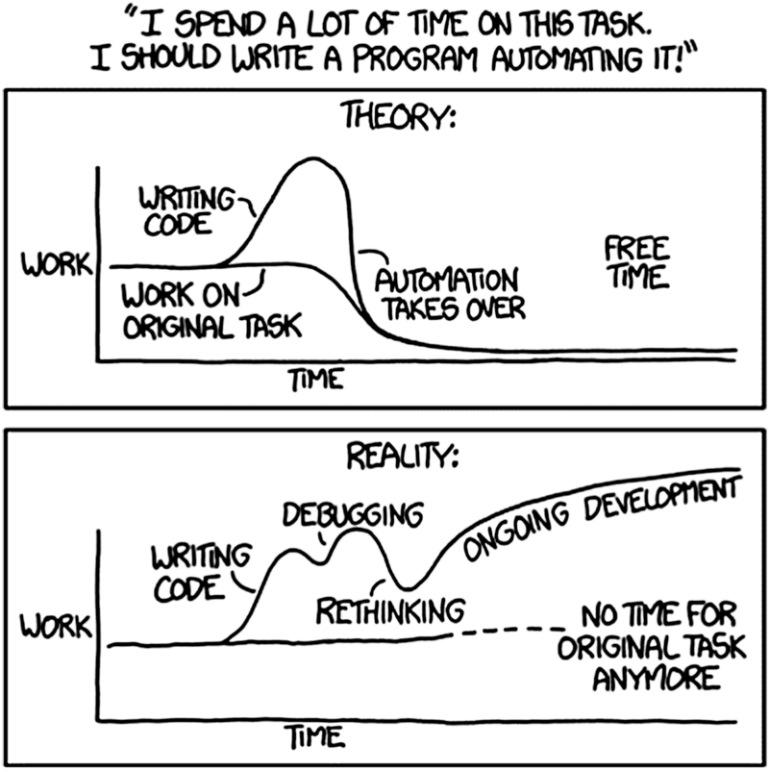



**Abb. 1.1** Der Fluch der Automatisierung. (Quelle: Munroe (2013; https://xkcd.com/1319/))

Vorbereitung der Infrastruktur und die Behebung von Fehlern führen häufig dazu, dass kontinuierlicher Aufwand betrieben werden muss. Gleichzeitig verändern sich dabei die Fragestellungen und Probleme, an denen gearbeitet wird. Insofern ist die Beschäftigung mit Computational Methods trotz oder vielleicht gerade aufgrund von Bemühungen zur maschinellen Automatisierung ein kreativer und inspirierender Prozess.

Automatisierung kann in allen Phasen des Forschungsprozesses eine Rolle spielen:

- Mittels automatisierter **Datenerhebung** können Inhalte wie etwa Kommentare auf Social-Media-Plattformen oder Webseiten in großem Umfang erschlossen werden. Zu diesem Zweck wird Webscraping eingesetzt, bei dem im Prinzip der Browser so automatisiert wird, dass die Klicks eines Menschen simuliert werden. Aus den Quelltexten der angesurften Webseiten werden dann die gewünschten Daten extrahiert. Einige Anbieter stellen auch Application Programming Interfaces (APIs) bereit, über die vorstrukturierte Daten abgerufen werden können.
- Automatisierte **Datenaufbereitung** meint die Umwandlung unstrukturierter Inhalte in strukturierte Daten. Unstrukturierte Inhalte zeichnen sich dadurch aus, dass die Eigenschaften der Fälle nicht bereits in standardisierter Form vorliegen. Bei der automatisierten Textanalyse werden beispielsweise aus Kommentaren die Sätze und Wörter extrahiert und in Datensätze umgeformt, sodass Wörter zu Variablen werden (Document-Term-Matrix). Bei der Datenaufbereitung geht es auch darum, nicht benötigte Daten zu entfernen (engl. *boilerplate removal*).
- Der Übergang zwischen automatisierter Aufbereitung und **Datenanalyse** ist fließend. Auf Grundlage strukturierter Textdaten lassen sich beispielsweise Kommentare danach klassifizieren, ob sie eher positive oder eher negative Aussagen enthalten. Hier unterscheidet man überwachte und unüberwachte Lernverfahren. Erstere zeichnen sich dadurch aus, dass die Zielkategorien durch menschlich erstelltes Trainingsmaterial vorgegeben werden. Letztere sind explorativ angelegt und gruppieren Fälle nach Ähnlichkeit, um zum Beispiel Themen zu bestimmen. In das Feld automatisierter Datenanalyse fallen darüber hinaus Netzwerkanalysen, Zeitreihenanalysen, geografische Analysen und Computersimulationen.[8]

[8] Für den Bereich Computational Social Science unterscheidet Cioffi-Revilla (2010, S. 260) automatische Informationsextraktion, Netzwerkanalysen, geografische Informationssysteme, Modellierung von Komplexität und soziale Simulationsmodelle.

- Bei der **Darstellung** von Ergebnissen können ebenfalls computerbasierte Methoden zum Einsatz kommen. Beispielsweise werden große Netzwerke häufig mit Algorithmen visualisiert, bei denen die Elemente schrittweise in eine zweidimensionale Anordnung gebracht werden, sodass verbundene Elemente möglichst nah beieinanderstehen. Darüber hinaus lassen sich Ergebnisse interaktiv aufbereiten und online veröffentlichen, damit einzelne Datenkategorien oder Parameter von den Nutzer:innen nachträglich angepasst werden können.

In diesem Sinne wird der Begriff im vorliegenden Buch verwendet: Der Bereich Computational Methods umfasst alle Verfahren der *automatisierten* Datenerhebung, -aufbereitung, -analyse und -darstellung.[9] Im Buch werden einerseits konzeptionelle Grundlagen vermittelt und andererseits praktische Anleitungen gegeben. Eine lineare Lektüre ist dabei nicht unbedingt nötig. Sie können beispielsweise direkt in die Kapitel zum Webscraping einsteigen. Je nach Vorwissen wird es hilfreich sein, von dort gezielt in die vorangegangenen Kapitel zu springen, vor allem, wenn Ihnen die verwendeten Begriffe, Datenformate und Programmiertechniken unbekannt sind. In Kap. 12 am Ende des Buchs finden Sie weitere Hinweise zu möglichen Leserichtungen.

## 1.2 Der Werkzeugkoffer

Zwei Werkzeuge sind nicht nur für die folgenden Kapitel, sondern auch sonst für die Arbeit mit Computational Methods unverzichtbar: die Kommandozeile und ein guter Texteditor. Beides wird im Folgenden kurz eingeführt. Sie können die Ausführungen gegebenenfalls zunächst überfliegen und später nachschlagen. Zumindest die im ersten Abschnitt genannten Grundregeln für den Umgang mit Dateien sollten Sie aber möglichst frühzeitig umsetzen. Im letzten Abschnitt zu den Begleitmaterialien finden Sie außerdem Hinweise zu Daten und Skripten, die ergänzend zu den einzelnen Kapiteln als Hilfsmittel zur Verfügung stehen.

Dieses Buch führt damit in die Tiefe der Computational Methods ein und soll dazu beitragen, dass Sie entsprechende Werkzeuge irgendwann auch selbst entwickeln können. Alternativ finden sich mittlerweile unter dem Stichwort Forschungssoftware (engl. *research software*) viele leistungsfähige Tools mit grafischen Be-

[9] Die Division der ICA definiert ähnlich: „Computational methods cover computerized tools and algorithms for collecting, processing, analyzing, and visualizing data such as social media data, news sites, and other forms of communication“ (ICA CM 2017).

nutzeroberflächen.[10] Diese sind sehr hilfreich für einen schnellen Einstieg und als Inspiration für mögliche Analyseverfahren. Wenn Sie sich in einen Bereich neu einarbeiten – beispielsweise in die Netzwerkanalyse –, dann lohnt es sich, zunächst nach passender Software zu recherchieren. Für viele Bereiche haben andere Wissenschaftler:innen bereits Materiallisten zusammengestellt, die häufig als Awesome List bezeichnet werden.[11] Allerdings sind die Möglichkeiten dann auf die vorgegebenen Funktionen begrenzt und auch nicht immer im Detail transparent. Diese Beschränkungen können Sie überwinden, indem Sie selbst die Grundtechniken von Computational Methods erlernen.

### 1.2.1 Grundregeln für Ordner und Dateien

Wichtig ist für die Arbeit mit Daten, dass Sie sich mit dem Dateisystem Ihres Computers auskennen. Einige Grundregeln erleichtern die Arbeit ungemein:

1. Erstellen Sie ein **Arbeitsverzeichnis**, das Sie leicht erreichen können, beispielsweise in dem Dokumente-Ordner Ihres Computers. Unter einem Verzeichnis versteht man einen Ordner, in dem Unterordner und Dateien abgelegt werden. Die Anzahl der Dateien wächst in Programmierprojekten schnell an. Nutzen Sie deshalb zur Organisation auch Unterverzeichnisse. Arbeiten Sie immer in diesem Verzeichnis, beispielsweise wenn Sie die Kommandozeile starten.

   Zu Ordnern, Unterordnern und Dateien können Sie mit der Maus navigieren, indem Sie im Explorer (Windows) bzw. dem Finder (Mac)[12] die entsprechenden Symbole anklicken. Es ist allerdings auch möglich, Dateien oder Verzeichnisse gezielt über den sogenannten Pfad zu adressieren. Der Aufbau von Pfaden unterscheidet sich zwischen den Betriebssystemen. Unter Windows (Abb. 1.2) beginnt ein Pfad mit dem Buchstaben des Laufwerks gefolgt von einem Doppelpunkt (meistens *C:*). Danach folgen getrennt mit Backslashs[13] die Unterord-

[10] Siehe zum Beispiel die Sammlungen von Social Media Data Stewardship (2021; https://socialmediadata.org/social-media-research-toolkit/) der Deutsche Gesellschaft für Publizistik- und Kommunikationswissenschaft (2022; https://www.dgpuk.de/de/forschungssoftware.html). Meine Forschungssoftware. Zugriff am 16.05.2022.

[11] Zum Beispiel: Briatte (2021; https://github.com/briatte/awesome-network-analysis).

[12] Der Explorer bzw. Finder sind die Programme, mit denen man auf die Laufwerke und die eigenen Dateien zugreift.

[13] Unter Windows werden Pfadelemente mit dem Backslash \ getrennt, unter Linux und Mac dagegen mit einfachem Slash / . Der Vorwärtsslash / funktioniert in der Regel aber auch unter Windows.

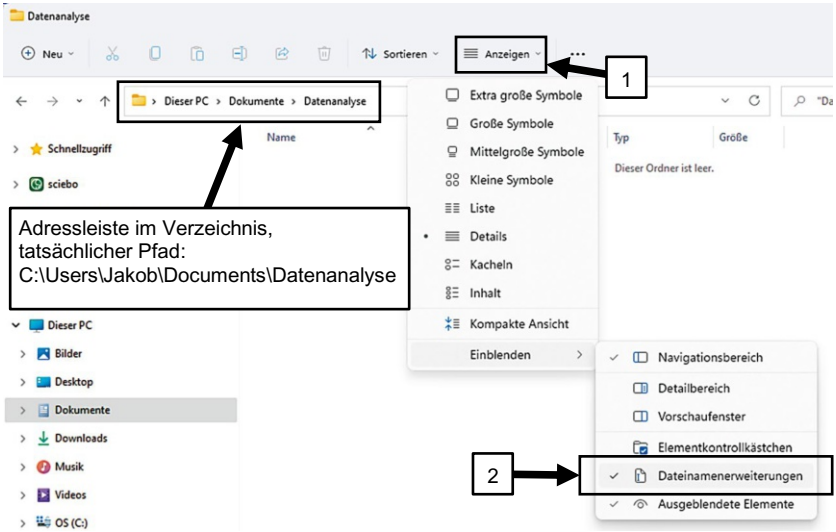



**Abb. 1.2** Die Adressleiste und das Einblenden von Dateierweiterungen unter Windows 11. Hinweis: Beachten Sie den Unterschied zwischen dem angezeigten Pfad in der Adressleiste und dem tatsächlichen Pfad. (Quelle: eigene Darstellung)

ner. Der Pfad ist im Explorer in der Adressleiste erkennbar. Zu beachten ist, dass die Anzeige häufig gekürzt ist und englische Ordnernamen übersetzt sind. Der tatsächliche Pfad kann herausgefunden werden, indem man mit der Maus innerhalb eines Unterordners in die Adressleiste klickt. Unter Mac und Linux beginnt ein Pfad mit einem Slash, danach folgen getrennt mit weiteren Slashs die Unterordner. Eine vollständige Angabe ist hier aber nur selten nötig, denn mit einer Tilde ~ kann direkt auf das aktuelle Benutzerverzeichnis verwiesen werden (Abb. 1.3).

Wenn Sie bereits im gewünschten Verzeichnis sind, können Sie in der Kommandozeile oder beim Programmieren relative Pfade einsetzen. Relative Pfade beginnen im aktuellen Verzeichnis. Aus dem aktuellen Arbeitsverzeichnis und dem relativen Pfad wird automatisch der absolute Pfad zusammengesetzt. Wenn beispielsweise der Ordner *C:\Users\Jakob\Documents* Ihr aktuelles Arbeitsverzeichnis ist, dann reicht die Angabe *Datenanalyse*, um das entsprechende Unterverzeichnis zu benennen.

2. Verwenden Sie in **Dateinamen** keine Leerzeichen, Umlaute und Sonderzeichen. Anstelle von Leerzeichen können Sie Unterstriche verwenden. Unter Mac

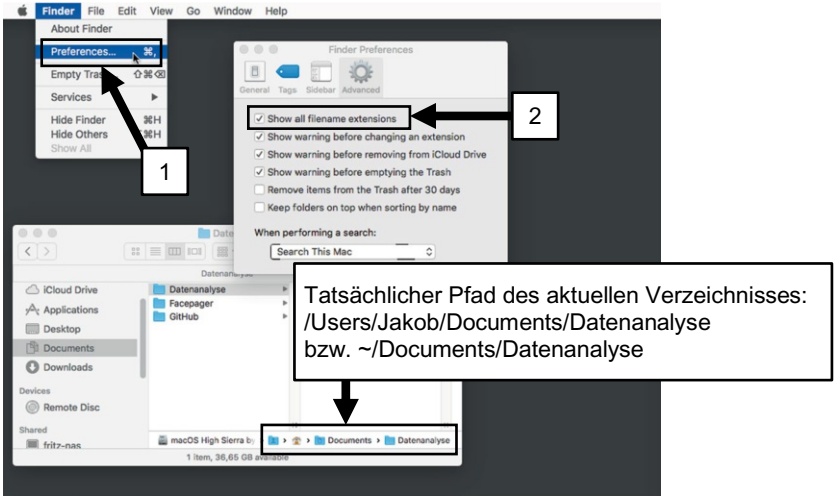



**Abb. 1.3** Dateierweiterungen unter macOS im Finder einblenden. Hinweis: Beachten Sie, dass sich die Tilde im tatsächlichen Pfad auf das Benutzerverzeichnis bezieht. (Quelle: eigene Darstellung)

und Linux wird Groß- und Kleinschreibung unterschieden. Deshalb ist es günstig, alle Namen in Kleinschreibung zu halten.

Dateien setzen sich aus einem Namen und einer Endung zusammen. Die Dateiendung umfasst den letzten Teil nach dem Punkt; darüber ist der Dateityp erkennbar. Bilder haben beispielsweise die Endung *.png* oder *.jpg*. Quelltexte enden mit *.R* oder mit *.py*. Daten werden häufig in Dateien mit den Endungen *.csv* und *.json* abgelegt. Textdateien erkennen Sie an der Endung *.txt*. Auch Markdown-Dateien mit der Endung *.md* werden Ihnen begegnen. Diese enthalten Text, der mit einfachen Konventionen so strukturiert und formatiert ist, dass er zum Beispiel auf Webseiten angezeigt werden kann (siehe Kap. 3).

Je nach Voreinstellung Ihres Betriebssystems müssen Sie die Dateiendungen erst einblenden, das sollten Sie am besten jetzt sofort tun! Unter Windows 11 (Abb. 1.2) öffnen Sie dazu den Explorer, wechseln in den Reiter ANZEIGEN und wählen im Bereich EINBLENDEN die Option DATEINAMENERWEITERUNGEN. Unter macOS (Abb. 1.3) öffnen Sie im Finder die Voreinstellungen und wählen die Option ALLE DATEINAMENSUFFIXE EINBLENDEN. Unter Linux-Systemen werden die Erweiterungen normalerweise standardmäßig angezeigt.

### 1.2.2 Die Kommandozeile

Mit der Kommandozeile, auch Eingabeaufforderung, Konsole, Terminal oder Shell genannt, schauen Sie hinter den Vorhang der grafischen Oberfläche von Windows, Mac oder Linux. Hier werden über die Tastatur Befehle eingegeben, mit denen zum Beispiel Programme installiert oder gestartet werden. Die Handhabung und die Befehle unterscheiden sich etwas zwischen den Betriebssystemen. Auch wenn Sie normalerweise unter Windows oder Mac arbeiten, kommt vermutlich irgendwann der Zeitpunkt, zu dem Sie in ein Linux-Terminal wechseln müssen. Denn insbesondere Cloud-Computing, das heißt die Nutzung von Servern statt des eigenen Computers, setzt üblicherweise auf einer Infrastruktur mit Linux-Systemen auf.

Die klassische Kommandozeile[14] öffnen Sie unter **Windows** zum Beispiel, indem Sie die Windows-Taste drücken und die Buchstaben `cmd` (= command) eingeben. Nach dem Starten wird das aktuelle Verzeichnis angezeigt (Abb. 1.4). Hinter dem aktuellen Verzeichnis blinkt ein Cursor und Sie können an dieser Stelle Be-

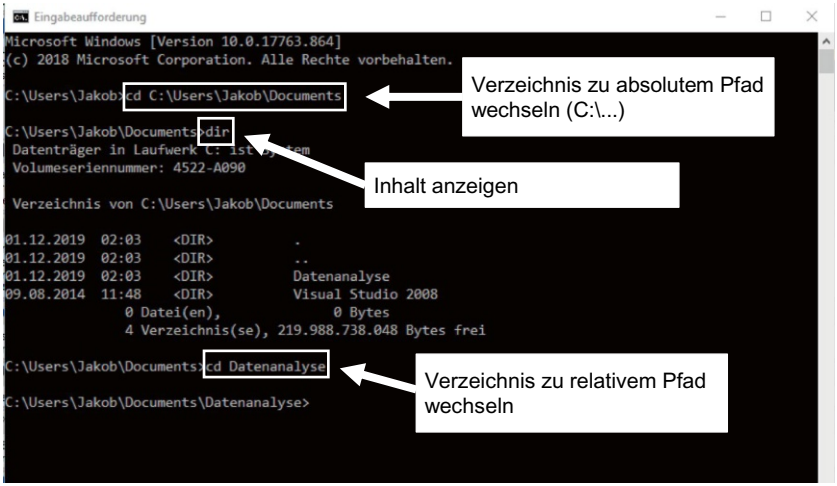



**Abb. 1.4** Die Eingabeaufforderung unter Windows 10. (Quelle: eigene Darstellung)

[14] Unter Windows steht außerdem die mächtigere PowerShell zur Verfügung, die Befehle unterscheiden sich teilweise von der Eingabeaufforderung. Darüber hinaus können Shells installiert werden, die der Linux- und Mac-Shell gleichen. Wenn Sie die Versionsverwaltung Git (siehe unten) installieren, steht anschließend eine Linux-ähnliche Shell zur Verfügung.

fehle eingeben, um beispielsweise Verzeichnisse zu wechseln oder Programme zu installieren. Ein Befehl wird ausgeführt, sobald er mit der Enter-Taste bestätigt wird.

Das Verzeichnis wechseln Sie mit dem Befehl `cd` (= change directory) und der Angabe eines absoluten oder relativen Pfads.[15] Der Befehl `dir` zeigt den Inhalt des Verzeichnisses an, also die enthaltenen Dateien und Ordner. Um vom aktuellen Verzeichnis eine Ebene nach oben zu wechseln, wird der Befehl `cd ..` verwendet. Längere Pfade oder Befehle können auch über die Zwischenablage in die Kommandozeile eingefügt werden.

Den Umweg über den `cd`-Befehl kann man sich aber sparen, wenn man das Verzeichnis zunächst im Explorer öffnet. Gibt man anschließend oben in die Adressleiste `cmd` ein und bestätigt mit der Entertaste, dann wird die Kommandozeile in diesem Verzeichnis geöffnet.

Auf dem **Mac** lässt sich die Kommandozeile öffnen, indem in der Spotlight-Suche (Lupen-Symbol) „Terminal“ eingegeben wird. Im Terminal wird zu Beginn einer Zeile der Name des Computers angezeigt, nach einem Doppelpunkt folgen der Name des aktuellen Verzeichnisses, dann ein Leerzeichnen und der Benutzername (Abb. 1.5). Die Befehle werden hinter dem Dollarzeichen eingegeben. Auch hier werden Verzeichnisse durch den Befehl `cd` gefolgt von einem relativen oder absoluten Pfad gewechselt. Der Inhalt des aktuellen Verzeichnisses wird mit dem Befehl `ls` angezeigt, wobei der Parameter `-l` für eine kompakte Darstellung sorgt: `ls -l`.

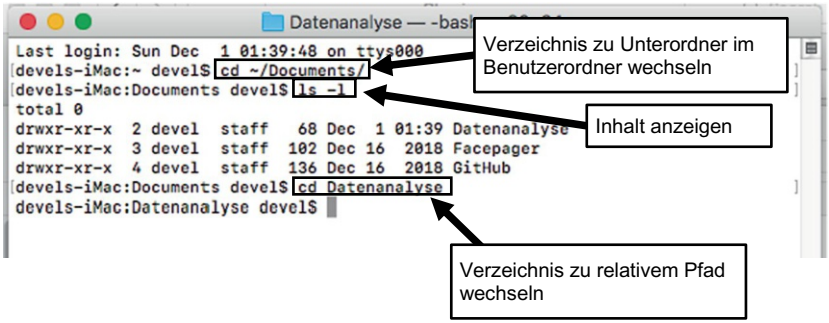



**Abb. 1.5** Das Terminal unter macOS High Sierra. (Quelle: eigene Darstellung)

[15] Das Laufwerk können Sie nicht über den Befehl `cd` wechseln, stattdessen muss der Buchstabe mit Doppelpunk ohne weiteren Befehl eingegeben werden.

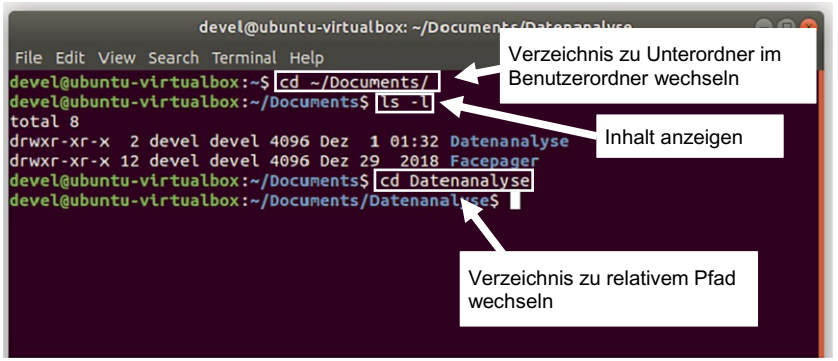



**Abb. 1.6** Das Terminal unter Ubuntu 18. (Quelle: eigene Darstellung)

Unter einem **Linux**-System wie Ubuntu erreichen Sie die Kommandozeile über die Command-Taste und Eingabe von „Terminal“. Da sowohl macOS als auch Linux zu den Unix-Systemen zählen, ist die Bedienung identisch (Abb. 1.6).

Egal auf welchem System Sie arbeiten, einige Funktionen sind in nahezu jeder Kommandozeile enthalten. Mit den Pfeiltasten (hoch/runter) können Sie etwa vorherige Befehle aufrufen. Bei langen Verzeichnisnamen hilft auch die Autovervollständigung: Tippen Sie die ersten Buchstaben ein und drücken Sie dann die Tabulatortaste!

Die Kommandozeile können Sie schließen, indem Sie das Fenster schließen. Wenn Sie innerhalb der Kommandozeile in einem Befehl festhängen, dann können Sie in der Regel über die Tastenkombination Strg + Pause oder Strg + C (Windows) bzw. Command + Pause oder Command + C (Mac) entkommen.[16]

### 1.2.3 Texteditoren

Ein weiteres wichtiges Werkzeug sind Texteditoren. Denn viele Datenformate, Skripte und Quelltexte sind Textformate. Vor allem wenn man den Inhalt einer Datei oder das Dateiformat nicht kennt, sollte man die Datei zunächst mit einem Texteditor erkunden. Auf der Kommandozeile stehen unter Unix-Systemen häufig die Editoren `vim` und `nano` zur Verfügung.

[16] Das Pluszeichen bei der Angabe von Tastenkombinationen bedeutet, dass die ersten Tasten gehalten werden, es wird nicht mit eingegeben.

Für Einsteiger ist die Arbeit auf der Kommandozeile aber häufig etwas umständlich. Ein für alle Betriebssysteme geeigneter Open-Source-Texteditor ist Atom.[17] Nach dem ersten Starten wird eine Einführung präsentiert. Eine Stärke dieses Editors ist die Paketverwaltung (Menüpunkt EINSTELLUNGEN), über die viele Erweiterungen nachinstalliert werden können. Unter Windows ist Notepad++ empfehlenswert und die meisten Beispiele in diesem Buch werden mit diesem Editor illustriert.[18] Ein Texteditor, der speziell für MacOS entwickelt wurde, ist Textmate.[19] Eine weitere betriebssystemübergreifende Alternative bietet der Editor VS Code.[20] Installieren Sie sich am besten jetzt gleich einen solchen Texteditor!

Um Dateien in einem Texteditor zu öffnen, gibt es in der Regel zwei Wege. Entweder starten Sie zuerst den Editor und öffnen die Datei über das entsprechende Menü. Oder Sie suchen im Dateimanager (Explorer, Finder bzw. Files) das Verzeichnis und öffnen die Datei von dort aus über das Kontextmenü der Datei. Dieses Kontextmenü wird in der Regel über die rechte Maustaste erreicht. Dort finden Sie beispielsweise unter Windows einen Punkt ÖFFNEN MIT und können dann den Texteditor auswählen. Erscheint der Texteditor nicht bereits bei den vorgeschlagenen Programmen, können Sie ihn dort hinzufügen. Dafür suchen Sie den Ordner, in welchem der Editor installiert ist und wählen in diesem unter Windows die Datei mit der Endung *.exe* aus.

### 1.2.4 Begleitmaterialien zum Buch

Ein weiteres Hilfsmittel für den Einstieg in die Welt der Computational Methods können die Begleitmaterialien zu diesem Lehrbuch sein. Da Computational Methods sehr praktisch sind, sollen die vorbereiteten Beispiele, Skripte und Datensätze die einzelnen Kapitel dieses Buchs ergänzen und dabei helfen, schrittweise eigene praktische Kompetenzen aufzubauen. Besonders bei komplexen Verfahren kann es hilfreich sein, zunächst vorbereitete Skripte Schritt für Schritt nachzuvollziehen, bevor man selbst Anpassungen vornimmt und schlussendlich eigene Skripte schreibt.

Die Begleitmaterialien befinden sich in einem GitHub-Repositorium,[21] das zu Beginn eines jeden Kapitels verlinkt ist und auf das wir im Text mit ☛ *Reposito-*

[17] Siehe GitHub (2022a; https://atom.io/).

[18] Siehe Ho (2022; https://notepad-plus-plus.org/).

[19] Siehe MacroMates (2021; https://macromates.com/).

[20] Siehe Microsoft (2022b; https://code.visualstudio.com/).

[21] GitHub ist eine Plattform für Entwickler:innen, die dort ihren Code teilen, die Entwicklungsarbeit dokumentieren und koordinieren (siehe Abschn. 6.1).

*rium* verweisen. Die Inhalte des Repositoriums können Sie entweder im Browser öffnen oder lokal auf Ihrem Computer speichern und bearbeiten. Um die Dateien gesammelt herunterzuladen, finden Sie unter der Bezeichnung CODE einen Link zu einer Zip-Datei. Da Repositorien auf GitHub unter Versionsverwaltung stehen, können diese auch über Befehle der Versionsverwaltung heruntergeladen werden. Nutzen Sie die Gelegenheit, um das Zusammenspiel von Kommandozeile, Versionsverwaltung und Texteditor auszuprobieren! Dazu installieren Sie zunächst die für Ihr Betriebssystem passende Git-Version.[22] Anschließend können Sie über die folgenden Schritte das Verzeichnis herunterladen:

1. Legen Sie ein Arbeitsverzeichnis auf Ihrem Computer an und öffnen Sie dort die Kommandozeile.
2. Laden Sie das Repositorium über die Kommandozeile mit folgendem Befehl herunter: `git clone https://github.com/strohne/cm`
3. Öffnen Sie in einem der Verzeichnisse des heruntergeladenen Repositoriums die Datei *readme.md* mit Notepad++, Atom oder einem anderen Texteditor.

Einen Einstieg in Versionsverwaltungen bietet Abschn. 6.1.

## 1.3 Mit Fehlern umgehen

Fehler zu machen, zu erkennen und zu beheben ist ein ganz wesentlicher Bestandteil von Computational Methods. Schließlich handelt es sich um ein interdisziplinäres Feld, das Wissen und Methoden aus unterschiedlichen Bereichen umfasst, in denen man nicht zwangsweise bereits Expert:in ist. Außerdem lassen sich kaum neue Daten erschließen, Programme erkunden oder Methoden ausprobieren, ohne dabei auch einmal festzustecken oder zeitweise in die falsche Richtung zu laufen.

Während das Lösen von Problemen durchaus viel Spaß bereiten kann, wenn man an neuen Ideen knobelt und dabei über sich hinauswächst, kann es gleichzeitig frustrierend und zeitintensiv sein. Um möglicher Frustration vorzubeugen, sind nachfolgend einige häufige Fehler und Tipps aufgeführt:

- Manchmal werden Dateien oder Verzeichnisse nicht gefunden. Hier hilft es, systematisch zu überprüfen, ob sich der angegebene Pfad auch auf das richtige Arbeitsverzeichnis bezieht, die Datei auch wirklich in diesem Ordner liegt und das Datenformat korrekt ist.

[22] Siehe Git (2022a; https://git-scm.com/downloads).

- Häufige Fehlerquellen sind außerdem Schreibfehler. Diese sind leicht zu übersehen, da sie in einigen Programmen nicht optisch hervorgehoben werden. Auch Leerzeichen, Groß- und Kleinschreibung machen in den meisten Programmiersprachen einen Unterschied. Es hilft oft, mehrfach zu prüfen, ob Verzeichnisse, Variablen oder Befehle richtig geschrieben sind. Einige Wörter wie „for“ oder „in“ sind in Programmiersprachen mit Funktionen belegt und sollten nur dafür verwendet werden.
- Wenn Dateien nicht korrekt eingelesen werden, kann dies an Steuerzeichen oder nicht sichtbaren Zeichen liegen. In CSV-Dateien signalisieren zum Beispiel Zeilenumbrüche, wann eine neue Zeile in einer Tabelle beginnt und diese Markierungen können je nach Betriebssystem unterschiedlich formatiert sein. Hier hilft es, die Datei im Texteditor zu öffnen, nicht sichtbare Zeichen einzublenden und zum Beispiel Zeilenumbrüche auszutauschen (siehe Abschn. 3.1).
- Bei der Arbeit mit R oder Python verwendet man häufig Funktionen, also definierte Abläufe von Befehlen, die andere Entwickler:innen in sogenannten Packages bereitstellen. Packages werden meist laufend weiterentwickelt. Deshalb kommt es vor, dass Befehle nach einiger Zeit veralten und nicht mehr funktionieren. Hinweise dazu, wie Sie die Version der Packages überprüfen und aktualisieren, finden Sie in den entsprechenden Kapiteln (siehe Kap. 5). Mitunter werden Funktionen sogar abgeschafft (engl. *deprecated*), dann muss man sie durch Alternativen ersetzen.
- Über viele Fehlermeldungen haben sich meistens bereits andere geärgert. Deswegen hilft es häufig, angezeigte Fehlermeldungen in eine Suchmaschine einzugeben. Eine Plattform, auf der viele Problemlösungen dokumentiert sind und über die man Hilfe zu speziellen Programmierfragen bekommt, ist Stack Overflow.[23] Dahinter steht eine aktive Community, die sich gegenseitig bei Problemen rund um das Programmieren hilft.
- Eine gute Inspirationsquelle sind Cheatsheets, die im Internet zu allen möglichen Themen und Programmiersprachen oder -paketen zu finden sind. Auf nur ein bis zwei Seiten werden übersichtlich die entsprechenden Hilfsmittel zusammengefasst und es lässt sich schnell ein Überblick über wichtige Funktionen gewinnen.

Die aufgeführten Punkte erscheinen vielleicht im ersten Moment trivial. Dennoch tappen selbst ausgewiesene Expert:innen immer wieder in die gleichen Fallen. Die Fehlerbehebung bleibt stets eine der wichtigsten Tätigkeiten und es ist hilfreich, dafür nach und nach eigene Routinen auszubilden.

[23] Siehe Stack Overflow (2022; https://stackoverflow.com).

## 1.4 Überblick über das Buch

Computational Methods, wie sie hier verstanden werden, schlagen eine Brücke zwischen automatisierten Verfahren und inhaltlichen Fragestellungen. Fängt man an damit zu arbeiten, geht ein Großteil der Zeit in die Aufbereitung von Daten, die Erkundung von Methoden und natürlich in die Behebung von Fehlern. Allein die Auseinandersetzung mit den Methoden ist inspirierend für den Forschungsprozess, am Ende geht es aber darum, Phänomene in der geistigen, kulturellen und sozialen Welt zu verstehen und zu erklären. Mit Computational Methods werden zum einen altbekannte Fragestellungen adressiert, etwa wie sich Öffentlichkeit über Diskurse im Zeitverlauf entfaltet. Sie werfen neues Licht auf diese Phänomene, indem beispielsweise größere Textkorpora analysiert werden können oder Diskurse als Netzwerke von Akteuren und Texten verstanden werden. Zum anderen ergeben sich in unseren Lebenswelten allein durch das Aufkommen von Online-Plattformen und den Einsatz sogenannter Künstlicher Intelligenz – dieser schillernde Begriff meint nichts anderes als automatisierte Verfahren – auch neue Forschungsfragen. In Bezug auf Öffentlichkeit stellt sich beispielsweise die Frage, inwiefern Empfehlungssysteme zu einer Fragmentierung von Öffentlichkeit führen, wenn Menschen personalisierten Inhalten ausgesetzt werden. Und der Einsatz von Bots führt auch zu grundsätzlichen theoretischen Fragen, etwa zu der Frage, inwiefern Programme und Algorithmen als handelnde Akteure begriffen werden können und wem eigentlich die Verantwortung für automatisiertes Verhalten zugeschrieben wird.

Computational Methods sind somit in der Lebenswelt anzutreffen, gleichzeitig aber auch Analyseinstrumente für wissenschaftliche Fragestellungen. Die Welt dieser Methoden entwickelt sich ständig weiter. Damit Sie für unterschiedliche Szenarien gewappnet sind, werden im ersten Teil des Buchs konzeptionelle Grundlagen eingeführt. Unabhängig von konkreter Software oder bestimmten Tools werden Denkmuster vermittelt – insbesondere geht es darum, die Welt durch eine eckige Brille als Ansammlung von Matrizen und Tabellen zu betrachten. Denn dies ist eine grundlegende Datenstruktur, mit der Daten erfasst, transformiert und analysiert werden können.

- **Kap. 2** führt in Zugänge **zur automatisierten Datenerhebung** ein und benennt exemplarische Datenquellen: Unstrukturierte Inhalte können aus Webseiten ausgelesen werden, vorstrukturierte Daten werden über Application Programming Interfaces (APIs) bereitgestellt und schließlich findet sich mittlerweile eine Vielzahl von Datenbanken mit fertig aufbereiteten Datensätzen.
- In **Kap. 3** werden **Datenformate** vorgestellt, denen man in der Welt der Computational Methods begegnet. Grundlegend ist die Unterscheidung von Daten-

typen – beispielsweise von Zahlen oder Buchstaben. Diese Daten können tabellarisch zusammengestellt oder durch Auszeichnungssprachen und Objektdatenformate strukturiert sein. Auch für die Zusammenstellung mehrerer Tabellen zu Datenbanken gibt es etablierte Verfahren. Dahinter liegen Datenmodelle, mit denen die Abbildung der Wirklichkeit auf Datenstrukturen und damit die Bedeutung der Daten festgelegt werden.
- Verfahren zur **Datenextraktion** werden in **Kap. 4** beleuchtet. Mit Selektionsverfahren und -sprachen wie regulären Ausdrücken, XPath oder SQL lassen sich unstrukturierte in strukturierte Daten transformieren oder Teildatensätze auswählen, um sie schließlich für die Datenanalyse in Tabellen- oder Matrizenform umzuformen, zusammenzuführen und zu aggregieren.

Im zweiten Teil des Buchs findet sich jeweils eine kurze Einführung in die Programmierung mit R und Python, zwei der wichtigsten Sprachen für die Anwendung von Computational Methods. Diese Sprachen verweisen auf unterschiedliche Traditionen. Während R näher an der Welt der Statistik ist, wird Python insbesondere von Wissenschaftler:innen mit Informatikhintergrund eingesetzt. Beide Sprachen sind sehr gut dazu geeignet, Daten zu erheben, aufzubereiten und auszuwerten. Je nach Anwendungsgebiet können Sie persönliche Vorlieben ausbilden, sodass beispielsweise die Datenerhebung mit Python und die Datenanalyse mit R schneller von der Hand gehen.

- **Abschn. 5.1** führt kurz und knapp in die **Programmierung mit R** ein. Hier lernen Sie zunächst, wie die Entwicklungsumgebung RStudio aufgebaut ist, wie Befehle formuliert und Skripte entwickelt werden. Dabei werden grundlegende Funktionen, unter anderem aus dem Tidyverse, besprochen, um Datensätze einzulesen, zu filtern und zu analysieren. Auch das Erstellen von Grafiken ist ein wesentlicher Bestandteil der Datenanalyse mit R.
- **Abschn. 5.2** bietet eine praxisorientierte Einführung in die **Programmierung mit Python**. Zu Beginn lernen Sie, wie Sie Jupyter-Notebooks einrichten und nutzen. Anschließend wird in Basisbefehle, Datenstrukturen und Funktionen eingeführt. Für die Datenanalyse lernen Sie Funktionen aus der weit verbreiteten Programmbibliothek *pandas* kennen.
- In **Kap. 6** finden Sie Hilfestellungen, sobald Programmierprojekte größer werden, etwa wenn mehrere Personen gleichzeitig an einem Projekt arbeiten oder die Datenmengen sehr umfangreich werden. Dabei erfahren Sie, wann sich die Arbeit mit Versionsverwaltungen lohnt, wie Computer virtualisiert werden können und wie Datenanalysen auf ein Cluster für High-Performance-Computing ausgelagert werden.

Der dritte Teil des Buchs beschäftigt sich schließlich mit konkreten Anwendungsfeldern von Computational Methods, in denen die bis dahin thematisierten Grundtechniken eingesetzt werden. Die Beispiele werden in einer der beiden Programmiersprachen R oder Python angeleitet und gegebenenfalls um Hinweise zur Umsetzung in der jeweils anderen Sprache ergänzt. Jedes der Kapitel fasst die jeweiligen Verfahren kurz zusammen und führt schrittweise durch ein praktisches Beispiel. Zunächst werden Verfahren automatisierter Datenerhebung thematisiert:

- Eine Einführung in **Webscraping**, um Inhalte aus Webseiten auszulesen, bietet **Abschn. 7.1**. Dabei lernen Sie zunächst, wie Sie mit Python einzelne HTML-Dokumente herunterladen, Daten aus dem Quelltext extrahieren und abspeichern. Um mehrere Webseiten abzufragen, kann der Webbrowser mithilfe von Selenium automatisiert werden und Sie finden Hinweise auf Programme und Plattformen, die beim Webscraping unterstützen.
- Wie Sie mit **Application Programming Interfaces (APIs)** arbeiten, lernen Sie in **Abschn. 7.2**. Über APIs lassen sich vorstrukturierte Daten erheben, die von den Plattformbetreibern bereitgestellt werden. Zwei Anwendungsfälle verdeutlichen den Nutzen von APIs: Zum einen werden Social-Media-Daten mithilfe von Facepager über die Twitter-API erhoben und zum anderen wird automatische Bilderkennung über Googles Cloud-Vision-API vorgestellt.

Sobald die Daten vorliegen, kommt es zur Datenanalyse, um inhaltliche Fragestellungen zu beantworten:

- **Vorhersagen** und **Klassifikationen** haben eine lange Tradition in der Statistik, zum Beispiel in der Regressionsanalyse. Sie werden aus Sicht von Computational Methods häufig als Probleme des **Machine Learnings** begriffen und in **Kap. 8** behandelt. Zunächst wird in grundlegende Konzepte des maschinellen Lernens eingeführt. Als überwachtes Lernverfahren wird in Abschn. 8.1 ein künstliches neuronales Netz trainiert, um damit Bilder automatisiert vorgegebenen Kategorien zuzuordnen. In Abschn. 8.2 wird als unüberwachtes Lernverfahren Topic Modelling angewendet, mit dem Texte ohne vorab bekannte Kategorien sortiert werden.
- **Kap. 9** beschäftigt sich mit der automatisierten **Textanalyse**, die Texte als Daten begreift, indem sie diese in ihre Bestandteile zerlegt und in Variablen überführt. Das Kapitel behandelt Grundtechniken zum Auszählen von Wörtern, die diktionärsbasierte Inhaltsanalyse und gibt einen Ausblick auf die Analyse von Syntax und Semantik.

- Die Beziehungen zwischen Akteuren, aber auch Konzepten und Ereignissen, können über **Netzwerkanalysen** betrachtet werden. Aus netzwerkanalytischer Sicht interessiert beispielsweise, wie sich Informationen verbreiten oder wie sich die Ressourcen eines Akteurs durch die Beziehungen zu anderen Akteuren erklären lassen. Das **Kap. 10** beinhaltet eine Einführung in die grundlegenden Konzepte der Netzwerkanalyse sowie ein praktisches Beispiel zur Erhebung, Analyse und Visualisierung eines Netzwerkes.
- **Kap. 11** führt in **Simulationsverfahren** ein, bei welchen hypothetische Welten erschaffen und mit empirisch vorgefundenen Welten verglichen werden. Dadurch lässt sich zum einen nachvollziehen, wie aus dem Verhalten einzelner Akteure auf der Ebene von Gesamtsystemen komplexe Effekte emergieren – etwa inwiefern es zur Fragmentierung von Öffentlichkeit kommt, wenn vor allem personalisierte Inhalte konsumiert werden. Zum anderen kann überprüft werden, inwiefern die in einem Datensatz vorgefundenen Zusammenhänge überzufällig oder auffällig erscheinen, wenn sie mit kontrafaktischen Welten verglichen werden.

Auch wenn die Reihenfolge der drei Teile eine bestimmte Leserichtung nahelegt, sind alle Kapitel so konzipiert, dass sie losgelöst von den anderen gelesen werden können. Wir hoffen, dass sie gleichzeitig als Inspiration und als Nachschlagewerk dienen können. Für einen schnellen Einstieg kopieren Sie die Skriptschnipsel aus dem Buch oder Repositorium und wandeln Sie diese für eigene Zwecke ab. Fehler sind selbstverständlich vorprogrammiert.

### Übungsfragen

1. Was versteht man unter Computational Methods?
2. Wählen Sie eine wissenschaftliche Disziplin und finden Sie heraus, wie die Verbindung mit Computational Methods in dieser Disziplin bezeichnet wird! Wie heißt dieser Bereich zum Beispiel in der Musik oder in der Physik?
3. Was sind forschungspraktische Konsequenzen von Automatisierung?
4. Was sind absolute und relative Pfade?
5. Wie öffnen Sie die Kommandozeile?

### Weiterführende Literatur

Attewell, P. A. & Monaghan, D. B. (2015). *Data mining for the social sciences. An introduction.* Oakland: University of California Press.

Cioffi-Revilla, C. (2017). *Introduction to computational social science. Principles and applications* (2. Aufl.). London: Springer.

Jannidis, F., Kohle, H. & Rehbein, M. (2017). *Digital Humanities. Eine Einführung*. Stuttgart: J.B. Metzler.

Rogers, R. (2013). *Digital Methods.* Cambridge: The MIT Press.

Sloan, L. & Quan-Haase, A. (Hrsg.). (2017). *The SAGE handbook of social media research methods.* Los Angeles: SAGE reference.

## Literatur

Almind, T. C. & Ingwersen, P. (1997). Informetric analyses on the world wide web: methodological approaches to 'webometrics'. *Journal of Documentation*, *53*(4), 404–426. https://doi.org/10.1108/EUM0000000007205

Amaral, I. (2017). Computational Social Sciences. In L. A. Schintler & C. L. McNeely (Hrsg.), *Encyclopedia of Big Data* (S. 1–3). Cham: Springer. https://doi.org/10.1007/978-3-319-32010-6_41

Attewell, P. A. & Monaghan, D. B. (2015). *Data mining for the social sciences. An introduction*. Oakland: University of California Press.

Beck, K. (2006). *Computervermittelte Kommunikation im Internet*. München: Oldenbourg. https://doi.org/10.1524/9783486839203

Björneborn, L. (2004). *Small-world link structures across an academic web space. A library and information science approach*. Copenhagen: Royal School of Library and Information Science.

Björneborn, L. & Ingwersen, P. (2004). Toward a basic framework for webometrics. *Journal of the American Society for Information Science and Technology*, *55*(14), 1216–1227. https://doi.org/10.1002/asi.20077

Briatte. (2021). *Awesome Network Analysis. An awesome list of resources to construct, analyze and visualize network data*. Zugriff am 19.04.2022. https://github.com/briatte/awesome-network-analysis

Burton, O. V. (2005). American Digital History. *Social Science Computer Review*, *23*(2), 206–220. https://doi.org/10.1177/0894439304273317

Busa, R. A. (2004). Perspectives on the Digital Humanities. In S. Schreibman, R. G. Siemens & J. Unsworth (Hrsg.), *A companion to digital humanities* (S. xvi–xxi). Oxford: Blackwell.

Cao, L. (2017). Data Science. *ACM Computing Surveys*, *50*(3), 1–42. https://doi.org/10.1145/3076253

Cioffi-Revilla, C. (2010). Computational social science. *Wiley Interdisciplinary Reviews: Computational Statistics*, *2*(3), 259–271. https://doi.org/10.1002/wics.95

Cioffi-Revilla, C. (2017). *Introduction to computational social science. Principles and applications* (2. Aufl.). London: Springer.

CLARIAH-DE. (2022). *Willkommen bei CLARIAH-DE*. Zugriff am 16.05.2022. https://www.clariah.de/

Cohen, D. J. & Rosenzweig, R. (2006). *Digital history. A guide to gathering, preserving, and presenting the past on the Web*. Philadelphia: University of Pennsylvania Press.

Coleman, J. S. (1964). *An Introduction to Mathematical Sociology*. New York: Free Press.

Deutsche Gesellschaft für Publizistik- und Kommunikationswissenschaft (2022). https://www.dgpuk.de/de/forschungssoftware.html

DHI. (2020). *The DHI and Digital Humanities*. Zugriff am 08.05.2020. https://www.ceu.edu/dhi/what-is-digital-humanities

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, *17*(3), 37–54. https://doi.org/10.1609/aimag.v17i3.1230

Git. (2022a). Git. Fast-version-control (Version 2.36.0) [Computer software]. https://git-scm.com/downloads

GitHub. (2022a). Atom (Version 1.60.0) [Computer software]. https://atom.io/

Ho, D. (2022). Notepad++ (Version 8.4.1) [Computer software]. https://notepad-plus-plus.org/

ICA CM. (2017). *About the ICA Computational Methods Interest Group*. Zugriff am 29.11.2019. http://ica-cm.org

Jannidis, F., Kohle, H. & Rehbein, M. (2017). *Digital Humanities. Eine Einführung*. Stuttgart: J.B. Metzler.

Jünger, J. (2018). Mapping the field of automated data collection on the web. Data types, collection approaches and their research logic. In C. M. Stützer, M. Welker & M. Egger (Hrsg.), *Computational social science in the age of big data. Concepts, methodologies, tools, and applications* (S. 104–130). Köln: Halem.

Jünger, J. & Schade, H. (2018). Liegt die Zukunft der Kommunikationswissenschaft in der Vergangenheit? Ein Plädoyer für Kontinuität statt Veränderung bei der Analyse von Digitalisierung. *Publizistik*, *63*(4), 497–512. https://doi.org/10.1007/s11616-018-0457-6

MacroMates. (2021). TextMate (Version 2.0) [Computer software]. https://macromates.com/

Marres, N. (2017). *Digital sociology. The reinvention of social research*. Cambridge: Polity.

Microsoft. (2022b). Visual Studio Code (Version 1.67.1) [Computer software]. https://code.visualstudio.com/

Monroe, B. L. & Schrodt, P. A. (2008). Introduction to the special issue: The statistical analysis of political text. *Political Analysis*, *16*(4), 351–355. https://doi.org/10.1093/pan/mpn017

Munroe, R. (2013). *Automation. xkdc: A webcomic of romance, sarcasm, math, and language.* Zugriff am 16.05.2022. https://xkcd.com/1319/

Rogers, R. (2010). Internet Research. The Question of Method. A Keynote Address from the YouTube and the 2008 Election Cycle in the United States Conference. *Journal of Information Technology & Politics*, *7*(2–3), 241–260. https://doi.org/10.1080/19331681003753438

Russell, S. J. & Norvig, P. (2012). *Künstliche Intelligenz. Ein moderner Ansatz* (3., aktual. Aufl.). München: Pearson.

Scharkow, M. (2011). Zur Verknüpfung manueller und automatischer Inhaltsanalyse durch maschinelles Lernen. *Medien & Kommunikationswissenschaft*, *59*(4), 545–562. https://doi.org/10.5771/1615-634x-2011-4-545

Schubert, L. (2020). Computational Linguistics. In E. N. Zalta (Hrsg.), *The Stanford Encyclopedia of Philosophy.* Zugriff am 01.04.2020. https://plato.stanford.edu/archives/spr2020/entries/computational-linguistics/

Shah, D. V., Cappella, J. N. & Neuman, W. R. (2015). Big Data, Digital Media, and Computational Social Science. *The ANNALS of the American Academy of Political and Social Science*, *659*(1), 6–13. https://doi.org/10.1177/0002716215572084

Social Media Data Stewardship. (2021). *Social Media Research Toolkit*. Zugriff am 16.05.2022. https://socialmediadata.org/social-media-research-toolkit/

Stack Overflow. (2022). *A public platform building the definitive collection of coding questions & answers*. Zugriff am 24.04.2022. https://stackoverflow.com

Terras, M., Nyhan, J. & Vanhoutte, E. (Hrsg.). (2013). *Defining digital humanities. A reader*. Farnham: Ashgate.

Thelwall, M. [Michael]. (2009). *Introduction to Webometrics. Quantitative Web Research for the Social Sciences*. San Rafael: Morgan & Claypool. https://doi.org/10.2200/S00176ED1V01Y200903ICR004

Van Atteveldt, W. & Peng, T.-Q. (2018). When Communication Meets Computation: Opportunities, Challenges, and Pitfalls in Computational Communication Science. *Communication Methods and Measures*, *12*(2–3), 81–92. https://doi.org/10.1080/19312458.2018.1458084

Welker, M. (2019). Computer- und onlinegestützte Methoden für die Untersuchung digitaler Kommunikation. In W. Schweiger & K. Beck (Hrsg.), *Handbuch Online-Kommunikation* (S. 531–572). Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-18016-4_21