Brandenburgische Technische Universität Cottbus-Senftenberg
Studiengang: Wirtschaftsingenieurwesen
Institut für Informatik, Informations- und Medientechnik sowie
Lehrgebiet Arbeitswissenschaft/Arbeitspsychologie

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Bachelorarbeit

Statistische Analyse pseudonymisierter Krankenkassenpatientendaten nach geografischen Gesichtspunkten

Florian Maria Höch

Matrikel Nr.: 3147507

2.Juli 2015

1. Gutachter: PD Dr.-Ing. habil. Thomas Hinze
2. Gutachter: Dipl.-Psych. Rico Ganßauge

Zusammenfassung

Die zentrale Aufgabe dieser Arbeit besteht in der deskriptiven Auswertung eines von der AOK-Sachsen-Anhalt zur Verfügung gestellten Datenbestandes unter Anwendung unterschiedlicher Methoden und Analysen. Dabei liegt der Schwerpunkt der Untersuchungen in der Identifizierung statistischer Auffälligkeiten, deren Entstehungsgründe in ihren geografischen Lagen zu suchen sind.
Das Datenmaterial enthält für das Bundesland Sachsen-Anhalt pseudonymisierte personenbezogene Informationen von ca. 600.000 AOK-Versicherten sowie zusätzliche Angaben über ihre ambulanten Behandlungen und verordneten Medikationen aus dem Jahr 2012.
Die Auswertung der mehreren Millionen Datensätzen wird mit Hilfe der freien Statistik-Software `R` realisiert. Die Programmiersprache `R` zeichnet sich durch eine Vielzahl an vordefinierten statistischen und grafischen Funktionen, sowie durch die effiziente Verarbeitung von Massendaten für die visuellen Aufbereitung aus.
Zu Beginn werden innerhalb der Einleitung einzelne Fragestellungen formuliert, deren Beantwortung das Hauptziel dieser Arbeit darstellt. Des Weiteren werden in den nachfolgenden Kapiteln einfache statistische Grundlagen vermittelt, sowie eine Einführung in die wichtigsten Befehle und Datenstrukturen der Programmiersprache `R` gegeben. Anschließend folgt eine detaillierte Beschreibung und Voranalyse des Datenmaterials, bevor mit den auswertenden und explorativen Studien begonnen wird.

Selbstständigkeitserklärung

Der Verfasser erklärt, dass er die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Datenschutzerklärung

Weiterhin erklärt der Verfasser, dass die Veröffentlichung dieser Arbeit im Einklang mit der Datenschutzvereinbarung gegenüber der AOK Sachsen-Anhalt vom 07.01.2015 und mit dem Kooperationsvertrag zwischen der AOK Sachsen-Anhalt und dem Lehrkrankenhaus Bergmannstrost vom 17.09.2013 steht und alle getroffenen Vereinbarungen, insbesondere Paragraph 5 des Bundesdatenschutzgesetzes, eingehalten werden.

Florian Höch

Cottbus, 02. Juli 2015

Danksagung

Mit dieser Bachelorarbeit habe ich einen wesentlichen Teil meines Studiums abgeschlossen und möchte mich deshalb bei allen bedanken, die mich bei der Erstellung tatkräftig unterstützten.

Speziell gilt mein Dank Dr. Hinze, der mich und meine Arbeit betreut hat. Vielen Dank für die Bereitstellung des Themas, die kontinuierliche Motivation und den moralischen Beistand. Mit seinem Fachwissen konnte er mir in engagierten Gesprächen hilfreiche Tipps und nützliche Anregungen für meine Abschlussarbeit geben. Zudem gilt mein Dank Herrn Ganßauge, der mich, als Zweitgutachter, mit wertvollen und konstruktiven Ratschlägen förderte.

Ganz herzlich bedanke ich mich ebenfalls bei allen Projektbeteiligten für Ihre Mithilfe. Durch den regen Kommunikationsaustausch und persönliche Dialoge sind zahlreiche Ideen für verschiedene Analysen entstanden.

Ein herzlicher Dank geht weiterhin an Johanna Schwerdtfeger, die viele Stunden in die Korrektur meiner Arbeit investiert hat. Auch möchte ich mich bei meinen Eltern bedanken, die mich stets unterstützen und an meiner Seite stehen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AMTS Arzneimitteltherapiesicherheit

AOK Allgemeine Ortskrankenkasse

ATC Anatomisch-therapeutisch-chemisches Klassifikationssystem

BMG Bundesministerium für Gesundheit

bzw. beziehungsweise

ca. circa

CRAN Comprehensive R Archive Network

DDD Defined Daily Doses

d.h. das heißt

ESRI Environmental Systems Research Institute

GIS Geoinformationssystem

ICD International Statistical Classification of Diseases and Related Health Problems

MLU Martin- Luther-Universität

MF Medikationsfehler

Mio. Millionen

NA not available

UAE unerwünschte Arzneimittelereignisse

u.a. unter anderem

PLZ Postleitzahl

PZN Pharmazentralnummer

sog. sogenannte(r)

vgl. vergleiche

WHO Weltgesundheitsorganisation

z. B. zum Beispiel

Quelltextverzeichnis

1. Einführung

Basierend auf den Schätzungen der Weltgesundheitsorganisation (WHO), die zu dem Ergebnis gekommen ist, dass ca. 10% aller Krankenhausaufnahmen in den Industriestaaten auf unerwünschte Arzneimittelereignisse (UAE) zurückzuführen sind und diese zum größten Teil vermeidbar wären, wurden die eingesetzten Mittel zur Überwachung von Arzneimitteln erweitert. Die bislang zur Anwendung gekommenen Maßnahmen, die sich mit der Entwicklung, Bereitstellung und Überwachung von Medikamenten beschäftigten, wurden durch die Mitaufnahme des Medikationsprozesses ergänzt. Dadurch sollen die Gesundheitsrisiken minimiert werden, die durch Medikationsfehler (MF) zu unerwünschten Arzneimittelereignissen führen. Studien haben ergeben, dass in Deutschland etwa 5% aller Krankenhausaufenthalte auf UAE zurückzuführen sind und davon etwa 40% vermeidbar gewesen wären [3, S. 4 f.]. Interne Untersuchungen der AOK Sachsen-Anhalt ergaben zudem, dass im Jahr 2012 ca. 650 Mio. € für Arzneimittel ausgegeben wurden. Dieser Summe liegen etwa 10 Mio. Rezepte zu Grunde, die Kosten verursachten, die weit über dem Bundesdurchschnitt liegen. Die Altersstruktur, Arbeitslosigkeit oder die Stadt-Land-Relation könnten u.a. dafür verantwortlich sein. Auch wird vermutet, dass durch unerwünschte Nebenwirkungen, hervorgerufen durch Dosierungsfehler, Arzneimittelinteraktionen oder inadäquate Arzneimittelauswahl, diese hohen Ausgaben entstanden sind [27, S. 3 ff.].
Der Aktionsplan Arzneimitteltherapiesicherheit (AMTS), der vom Bundesministerium für Gesundheit (BMG) aufgestellt wurde, soll unter anderem für eine bessere Organisation des Medikationsprozesses sorgen, um in diesen Bereichen eine Verbesserung zu erzielen. In Verbindung mit diesem Plan, startete ein Forschungsprojekt mit Kooperationspartnern der Allgemeinen Ortskrankenkasse (AOK) und der Martin-Luther-Universität (MLU) in Halle/Saale am Lehrkrankenhaus Bergmannstrost.

Ziel dieses Projektes ist es, die AMTS zu verbessern, welches zum einen dem Gesundheitszustand der Versicherten zu Gute kommt und zum anderen zu einer Senkung der Kosten für Arzneimittel führen soll. Die Arzneimittelkostensenkung würde sich wiederum zu Gunsten der Versicherten auswirken durch beispielsweise eine Beitragssenkung oder Minderung der Zuzahlungen für Medikamente. Dieses Ziel im Blick, sollen Maßnahmen innerhalb des Forschungsprojektes erarbeitet werden, um die augenblickliche Ausgangslage weiter zu verbessern. Die Maßnahmen können durch den Einfluss vieler Faktoren auf die Versicherten und deren Bedürfnisse regional unterschiedlich ausfallen, um den Versicherten eine optimale Versorgung zu ermöglichen.
Unter diesem Gesichtspunkt werden die zur Verfügung stehenden Daten über das Bundesland Sachsen-Anhalt nach regionalen Gegebenheiten untersucht.
Die Regionen in Sachsen-Anhalt sind unterschiedlich geprägt. Während im Südosten ein industriell-städtischer Charakter vorzufinden ist, ist der nördliche Teil Sachsen-Anhalts eher landwirtschaftlich ausgerichtet. Die südwestliche Region ist einerseits bekannt für Ihren Untertagebau und andererseits gilt sie als Erholungsgebiet mit vorwiegend forstwirtschaftlicher Nutzung.
Diese verschiedenartig geografischen Gegebenheiten lassen fragen, inwiefern sich unterschiedliche Wohnlagen der Patienten auf deren Krankheitsbilder auswirken. Mit anderen Worten, sind Menschen, die in ländlichen Gegenden wohnen, gesünder als Menschen, die in Industriegebieten leben oder findet man hier generell andere Erkrankungen vor? In Verbindung mit den Diagnoseschlüsseln der Krankheitsbilder sollen die Medikationen auf regionale Konzentrationen hin analysiert werden. Außerdem ist es interessant zu erfahren, ob signifikante Abweichungen zwischen den Diagnoseschlüsseln und der entsprechenden Medikation regional auftreten. Es soll festgestellt werden, ob Art und Menge von verschriebenen Medikamenten im adäquaten Verhältnis zur Diagnose stehen.
Zusätzlich ist zu untersuchen, inwiefern Medikationen vorgenommen werden, die im Gegensatz der Empfehlungen bei Menschen höheren Alters stehen. Dafür liegen anerkannte Kataloge bzw. Listen vor, die eine Sammlung von Wirkstoffen klassifizieren, bei denen eine potentielle Gefährdung besteht.
Weiterhin soll untersucht werden, ob es einen Zusammenhang zwischen dem Versicherungsstatus und bestimmten Krankheitsbildern gibt. Zwar ist davon auszugehen, dass

entsprechend viele Erkrankungen hauptsächlich in der Gruppe der Rentner anzutreffen, also stark altersabhängig sind, aber es soll auch untersucht werden, ob es Konzentrationen bezüglich der anderen Versicherungsarten, wie Arbeitslose, Familienangehörige und Beschäftigte im mittleren Alter gibt. Vorstellbar wäre, dass Familienangehörige generell weniger erkranken als die Gruppe der Beschäftigten, die unter dem Einfluss ihrer Arbeitsbedingungen, wie z. B. schwere körperliche Arbeit, Lärm, dauerhaft stehende- oder sitzende Tätigkeit oder psychischen Druck durch Überlastung, leiden. Ein weitverbreitetes Vorurteil in der Bevölkerung ist, dass Arbeitslose einen größeren Alkoholkonsum haben. Eine Analyse könnte zeigen, ob bestimmte Erkrankungen, die hauptsächlich auf einen übermäßigen Alkoholgenuss zurückzuführen sind, in der Gruppe der Arbeitslosen häufiger vorkommen.
Für eine Kostenanalyse, die ebenfalls regionenbezogen durchgeführt wird, werden die Versicherten in Geschlecht und verschiedene Altersklassen eingeteilt und den einzelnen Regionen zugeordnet, wodurch ermöglicht wird, Kostenverteilungen im Land Sachsen-Anhalt zu erkennen. In Bezug auf die verschriebenen Arzneimittel kann durch die regionale Untersuchung auch festgestellt werden, ob bestimmte wirkstoffgleiche Arzneimittel auffällig häufig oder selten in bestimmten Gebieten des Landes verordnet werden. Eine mögliche Ursache dafür wäre ein größerer Einfluss der Pharmavertreter auf alle Beteiligten des medizinischen Prozesses.
Für diese Untersuchungen wird ein Datenbestand herangezogen, der von der AOK Sachsen-Anhalt zur Verfügung gestellt wird. Er beinhaltet mehrere Dateien mit Informationen über die AOK-Mitglieder.
Ziel dieser Arbeit ist es, Antworten auf oben aufgeführte Fragen zu finden, wobei der bereitgestellte Datenbestand deskriptiv ausgewertet und durch anerkannte statistische Verfahren analysiert wird. Hierzu wird die freie Programmiersprache `R` verwendet, die im speziellen den Umgang mit großen Datenmengen und deren statistischen Auswertungen und visuellen Darstellungen ermöglicht.

2. Statistische Grundlagen und Methoden

In diesem Kapitel wird eine kurze geschichtliche Einordnung und Begriffsbestimmung der verschiedenen angewendeten Statistikverfahren vorgenommen. Im Anschluss daran werden die häufig zur Anwendung kommenden mathematischen Methoden erläutert.

Die Statistik kann unterschieden werden zwischen theoretischer und angewandter Statistik. Die angewandte Statistik ist ca. viereinhalb Jahrtausende alt und hat ihren Ursprung wahrscheinlich im alten Ägypten. Zum Zwecke der Besteuerung wurde dort das Gold und die Felder gezählt. Für den Pyramidenbau wurden Volkszählungen durchgeführt, um eine Übersicht über die Zahl der möglichen Arbeitskräfte zu bekommen. Im alten Rom wurden Zählungen und Vermögensschätzungen vorgenommen, um diese in den sog. Steuerlisten aufzuführen. Diese Art der Statistik bezieht sich auf Tätigkeiten, die sich mit der Erfassung und Aufbereitung von Massendaten befassen.
Die Grundlage der theoretischen Statistik bilden mathematische Regeln und Verfahren wie erfasste und aufbereitete Massendaten sinnvoll, in Verbindung mit Wahrscheinlichkeitstheorien, ausgewertet werden können. Diese theoretische Statistik ist ca. drei Jahrhunderte alt. Die Veröffentlichungen der beiden Mathematiker Pierre-Simon de Laplace und Jean Baptiste Joseph Fourier um 1800, die sich mit der Bevölkerungsstatistik beschäftigten, hatten einen wesentlichen Einfluss auf die weitere Entwicklung der theoretischen Statistik [17, S. 1-6].
In Zusammenhang mit diesen beiden Arten sprechen wir auch von der Methodenlehre. Die angewandte Statistik ist die konkrete Umsetzung statistischer Methoden wie sie in der Wirtschafts- und Bevölkerungsstatistik vorkommen. Die theoretische Statistik umfasst die Methoden, die unabhängig von der konkreten Anwendung herangezogen werden,

also die mathematischen Regeln und Verfahren. Eine neuere Form der Statistik ist die explorative Statistik, die erst durch eine moderne Rechentechnik in Form von Computern ermöglicht wurde, da sie sich mit der Auswertung von Massendaten beschäftigt und somit eine Brücke zum sog. Data-Mining schlägt [10, S. 3].
Unter Data-Mining versteht man die Gewinnung von neuen Erkenntnissen und Mustern, die ausgehend von einem großen Datenbestand und unter der Annahme, dass darin wertvolle Informationen verborgen sind, mit Hilfe rechnergestützter Analyseverfahren aufgedeckt und gezielt extrahiert werden können [19, S. 8].

Unabhängig von der Art oder der Methode der Statistik definiert Prof. Peter P. Eckstein die Statistik wie folgt [10, S. 2]:

> Statistik ist die Bezeichnung für die Gesamtheit von Verfahren und Methoden zur Gewinnung, Erfassung, Aufbereitung, Analyse, Abbildung, Nachbildung und Vorhersage von (möglichst) massenhaften , zähl-, mess- und/oder systematisch beobachtbaren Daten über reale Sachverhalte zum Zwecke der Erkenntnisgewinnung und Entscheidungsfindung (meist unter Ungewissheit).

Unter diese Definition fallen auch die deskriptive und die induktive Statistik, welche zwei Teilgebiete der allgemeinen Statistik beschreiben. Die deskriptive Statistik befasst sich mit der Sammlung, Auswertung und Analyse von ermittelten Daten, während die induktive Statistik unter Zuhilfenahme von Wahrscheinlichkeitstheorien eine Datenanalyse durchführt, auf der Grundlage eines kleinen und bekannten Datenbestandes (Stichprobe), um damit Rückschlüsse auf eine unbekannte und große Grundgesamtheit zu ziehen. Beispiel für eine induktive Auswertung ist die Hochrechnung bei Bundestags- oder Landtagswahlen. Aus der Grundgesamtheit der Wähler wird eine repräsentative Auswahl getroffen, ausgewertet und anschließend extrapoliert auf alle Wahlberechtigten.

Die deskriptive Statistik, auch beschreibende Statistik genannt, benutzt als Grundlage für die Auswertung immer eine Grundgesamtheit. Ihr Ziel ist es, gesammelte Massendaten durch geeignete Visualisierungswerkzeuge in Form von Diagrammen, Tabellen, Grafiken und Heatmaps darzustellen. Die Menge der Informationen, die diese Daten

enthalten, wären ohne eine vernünftige Aufbereitung, Zusammenfassung und Strukturierung kaum zu erfassen.

Eine statistische Untersuchung, egal welcher Art, kann in vier Hauptpunkte gegliedert werden [10, S. 3].

Die Planung: Sie umfasst die Festlegung des zu erreichenden Untersuchungszieles.

Die Datenerhebung: In diesem Schritt wird das zu untersuchende Datenmaterial erfasst. Die Art, Menge und Güte des Materials bestimmt letztendlich die Qualität des Ergebnisses der Untersuchung.

Die Datenaufbereitung: In diesem Schritt werden die Daten geordnet, zusammengefasst und dargestellt. Die Darstellung erfolgt in Tabellen, Dateien und Grafiken.

Die Datenanalyse: Mit Hilfe verschiedener statistischer Verfahren werden die Daten zur Erkenntnisgewinnung analysiert, um in weiteren Schritten entsprechend den Erkenntnissen Entscheidungen zu treffen.

Im Folgenden werden einige verschiedene Verfahren, die bei der Datenanalyse zur Anwendung kommen, vorgestellt.

Das arithmetische Mittel

Das arithmetische Mittel, oder auch Mittelwert genannt, gibt den Durchschnittswert einer endlichen Menge reeller Zahlen an. Er wird berechnet, indem die Summe aller gegebenen Werte durch die Anzahl der Werte dividiert wird. Der errechnete Wert muss kritisch betrachtet werden, da größere Abweichungen zwischen den einzelnen Zahlen bei der gegebenen Datenmenge, nach oben oder unten, das Ergebnis verzerren und zu falschen Schlussfolgerungen führen kann [26, S. 59 ff.].

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2.1}$$

Absolute Häufigkeit

Die absolute Häufigkeit gibt die Anzahl eines Ereignisses oder einer Wertgröße an, bezogen auf ein Experiment bzw. auf eine Datenmenge. Dieser Wert ist in Bezug auf seine Größe schwer zu beurteilen, solange man nicht die Größe der Datenmenge kennt, auf die sich der Wert bezieht [26, S. 20 f.].

$$H(x_i) \tag{2.2}$$

Relative Häufigkeit

Die relative Häufigkeit wird aus der absoluten Häufigkeit, dividiert durch die Gesamtzahl aller Werte, berechnet. Dadurch wird sie in ein Verhältnis zur Grundgesamtheit gesetzt, wodurch das Ergebnis besser beurteilbar wird. Die aufsummierten relativen Häufigkeiten müssen stets den Wert 1 ergeben. Wird die relative Häufigkeit noch mit dem Wert 100 multipliziert, erhält man die prozentuale relative Häufigkeit [26, S. 21 f.].

$$h(x_i) = \frac{1}{n} * H(x_i) \tag{2.3}$$

Die Varianz und Standardabweichung

Um die Varianz für eine Reihe von Daten zu berechnen wird der Mittelwert (siehe Formel 2.1) benötigt. Die Varianz gibt an, wie stark die betrachteten Werte um den Mittelwert der Datenreihe "streuen". Aus diesem Grund wird dieser Wert auch häufig Streuung genannt. Je dichter die betrachteten Werte um den Mittelwert liegen, desto geringer ist die Varianz. Wird die Wurzel aus dem Varianzwert (s^2 bzw. $\bar{s}^2$) gezogen, erhält man die Standardabweichung (σ bzw. $\bar{\sigma}$). Diese gibt konkret an, wie groß die durchschnittliche Abweichung eines jeden Messwertes vom arithmetischen Mittel ist [26, S. 75 ff.].
Man unterscheidet zwischen der korrigierten und unkorrigierten Stichprobenvarianz.

1. **Die korrigierte Varianz und Standardabweichung**
   Die korrigierte Varianz wird benutzt, wenn die zu untersuchenden Daten eine Stichprobe aus der Grundgesamtheit bilden. Hier wird der Mittelwert der Stichprobe ($\bar{x}$) zur weiteren Berechnung herangezogen. Je nach Auswahl der Stichprobendaten wird der Mittelwert eine andere Größe annehmen.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \qquad \sigma = \sqrt{s^2} \tag{2.4}$$

2. **Die unkorrigierte Varianz und Standardabweichung**
   Die unkorrigierte Varianz, auch empirische Varianz genannt, kommt zur Anwendung, wenn es sich bei dem Datenbestand um eine Grundgesamtheit handelt. Diese Varianz basiert auf dem wahren Mittelwert ($\hat{x}$), der berechnet werden kann, wenn alle Daten zur Verfügung stehen. Dieser Wert ist konstant und ändert sich für die Grundgesamtheit niemals.

$$\bar{s}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x})^2 \qquad \bar{\sigma} = \sqrt{\bar{s}^2} \tag{2.5}$$

3. Das freie Statistikpaket R und ausgewählte Funktionen

Dieses Kapitel soll den Umgang mit R verdeutlichen und eine Einführung in die Programmoberfläche geben. Zudem werden die grundlegenden Datentypen, Strukturen und Funktionen erläutert, die im Rahmen dieses Projektes zur Anwendung kommen. Am Ende dieses Kapitels wird ein Einblick in die Paketinstallation, die Paketverwaltung sowie die Paketnutzung gegeben, die für einige Analysen dieser Arbeit notwendig sind.

3.1. Die Programmiersprache R

Die Programmiersprache R wurde Anfang der neunziger Jahre von Ross Ihaka und Robert Gentleman in Auckland, Neuseeland, entwickelt. Basierend auf der Programmiersprache S, wurde R als sog. frei verfügbare Weiterentwicklung entworfen. R unterliegt seit 1995 der *GNU General Public License*. Dies bedeutet, dass der Quellcode von R für jeden frei zugänglich, nutzbar und veränderbar ist. Ein weiterer wichtiger Punkt ist die plattformübergreifende Kompatibilität. R ist auf allen gängigen Betriebssystemen, wie *Windows*, *Mac OSX* und *UNIX*-basierenden Systemen, lauffähig.
Der Vorzug der Sprache R liegt in der Vielzahl integrierter statistischer und stochastischer Methoden und Analysen, sowie der Möglichkeit, diese auf unterschiedlichste Art grafisch zu visualisieren [9, S. 1] [24].
R wird nicht nur als Programmiersprache, sondern auch als Umgebung bzw. Entwicklungssystem bezeichnet. Innerhalb dieser Umgebung haben Nutzer die Möglichkeit, sog. Pakete nachzuladen; das heißt, R ist einfach erweiterbar und kann durch über 6500 verfügbare Pakete, die offiziell über das sog. *Comprehensive R Archive Network* (CRAN) veröffentlich werden, für die entsprechende Aufgabe bzw. Problemstellung den Funk-

tionsumfang ergänzen. Auf der offiziellen Webseite findet man unter `Task Views` eine Einteilung der Pakete nach Themengebieten, um die Suche zu erleichtern und erhält eine Übersicht der Pakete, die für das jeweilige Themengebiet verfügbar sind. Zurzeit gibt es 33 Themengebiete wie z. B. Klinische Studien, Wahrscheinlichkeitsverteilungen, Finanzen und weitere wissenschaftliche Bereiche [24] [25].

Da mittels `R` fast jede statistische Problemstellung bewältigt werden kann, gewinnt es zunehmend Anwendung im wirtschaftlichen und wissenschaftlichen Bereich.

3.2. Entwicklungsumgebung

`R` ist grundsätzlich von den Compilersprachen wie `C` und `Java` zu unterscheiden. `R` zählt zu den sog. Interpretersprachen; das bedeutet, sie ist kommandozeilenorientiert und führt die eingegebenen Anweisungen unmittelbar nach dem Betätigen der *<ENTER>*-Taste zeilenweise von oben nach unten aus. Ruft man `R` nach der Installation auf, erscheint dessen Konsole. Unterhalb der Allgemeinen Informationen wie der Versionsnummer, Copyright-Rechte, Plattformangaben und Hilfehinweise für `R` wird der Kommando-*Prompt (>)* angezeigt. Nach dem *Prompt* erfolgt der eigentliche Befehlsaufruf bzw. die Eingabe des Benutzers. Hier können alle gewünschten Rechenoperationen oder auch, wie im Kapitel 3.3 beschrieben wird, Variablen zugewiesen werden. In `R` ist es nicht zwingend erforderlich, jeder Operation eine Variable zuzuweisen. Beispielsweise kann die Konsole wie ein Taschenrechner genutzt werden [4, S. 3].

Hierzu ein Beispiel: Tippt man nach dem *Prompt (>)* eine Additionsoperation ein und betätigt die *<ENTER>*-Taste, erscheint folgendes:

```
1 > 1+1
2 [1] 2
```

Das Ergebnis wird in der Form `[1] 2` dargestellt.
Die `2` bezeichnet hier das eigentliche Ergebnis der Operation, wobei die geklammerte `1` auf das erste Element des Ergebnisses hinweist. In diesem Beispiel ist das von geringerer Bedeutung, da es hier nur ein einzelnes Ergebnis gibt. In späteren Berechnungen, bei

denen viele Elemente zusammengefasst betrachtet werden, ist der angegebene Index des Elementes hilfreich [4, S. 3].
Auch können die Rechenoperationen beliebig kombiniert werden. Im nächsten Beispiel wird der Mittelwert der Zahlenreihe 1 bis 10 ermittelt:

```
1 (1+2+3+4+5+6+7+8+9+10) / 10
2 [1] 5.5
```

Das Ergebnis lautet: `5.5`

Bei einer größeren Anzahl von Anweisungen ist diese Art der Verarbeitung jedoch nicht effizient. Hierfür bietet die R-Umgebung über den Menü-Punkt *Ablage -> Neues Dokument* einen Skripteditor an, in dem eine Folge von Anweisungen abgelegt und gespeichert werden kann, um die wiederholte Verarbeitung zu einem beliebigen Zeitpunkt zu ermöglichen. Der R-Skript-Editor ist nicht zwingend erforderlich und könnte durch andere Text-Editoren ersetzt werden, jedoch unterstützt er den Nutzer durch unterschiedliche Farbgebungen des Quellcodes und ermöglicht eine einfache Autovervollständigung für eine korrekte Syntax.
Des Weiteren umfasst die R-Entwicklungsumgebung unter dem Menüpunkt *Pakete und Datensätze* zwei wichtige Unterpunkte, die das Paketmanagement steuern. Einer dieser Punkte umfasst die *Paketinstallation*. Sie ermöglicht es, Pakete in R zu installieren und zu integrieren. Die Pakete können aus verschiedenen *Paket-Repositories* (Quellen) bezogen werden. Die größte Quelle an Zusatzpaketen bietet das offizielle CRAN, aber auch andere Quellen aus dem Internet oder vom lokalen Rechner können mit einbezogen werden. Aufgrund der Menge an Paketen bietet R die Möglichkeit, über eine Suchmaske das benötigte Paket schnell zu finden und auszuwählen.
Der zweite Unterpunkt ist die *Paketverwaltung*. Die Paketverwaltung enthält eine tabellarische Auflistung aller lokal verfügbaren und installierten Pakete. Zusätzlich wird der aktuelle Status des Paketes und eine Kurzbeschreibung angezeigt. Der Status gibt Auskunft, ob das Paket *geladen* oder *nicht geladen* ist und kann durch eine Checkbox gesteuert werden. Wird ein Paket in der Auflistung markiert, wird die entsprechende Dokumentation des Paketes angezeigt. Die Dokumentation beinhaltet mindestens Informationen über die Versionsnummer und eine detaillierte Beschreibung jeder implemen-

tierten Funktion. Teilweise sind Beispiele zu einzelnen Funktionen abrufbar.
Weiterhin erlaubt die Umgebung die Speicherung der bisher getätigten Zu- und Anweisungen. Sämtliche Zuweisungen werden in dem sog. Arbeitsbereich und die Anweisungen im Verlauf gesichert. Beim Beenden der `R` Umgebung kann der Nutzer entscheiden, ob alle bisherigen Objekte gespeichert werden sollen und somit bei der nächsten Sitzung wieder zur Verfügung stehen. Der Verlauf aller Eingaben über das Kommando-*Prompt* wird in jedem Fall gesichert [9, S. 39].

3.3. Datentypen und Strukturen

Um Daten in `R` abspeichern zu können, müssen sie einer Variablen zugeordnet werden, wobei diese immer genau einen bestimmten Datentyp repräsentiert. Eine Variable setzt sich zusammen aus einem Namen und einem zugewiesenen Wert. Für die spätere Verarbeitung der Daten ist es wichtig, die entsprechenden Datentypen zu verwenden, um die gewollten Rechenoperationen durchführen zu können.

3.3.1. Einfache Datentypen

Die drei einfachsten genutzten Datentypen in `R` sind:

Numeric : Ganzzahl, ein sog. *integer* oder eine reelle Zahl *double*.

Character : Zeichen oder eine Folge von beliebigen Zeichen.

Logical : Boolescher Wert, der die Werte *TRUE* oder *FALSE* annehmen kann.

`R` besitzt noch weitere Datentypen, auf die hier nicht näher eingegangen wird, da sie kein Gegenstand der Untersuchungen sind [12, S. 28 f.]. Die Zuweisung einer Variablen erfolgt mittels `<-` oder auch `=`. Bei der Zuweisung ist darauf zu achten, dass zwischen Groß- und Kleinschreibung unterschieden wird und der Variablenname nicht mit einer Ziffer beginnen darf [28, S. 19, 21].
Beispielsweise lässt sich der Wert `1` der Variablen `x` wie folgt zuweisen:

```
1 > x <- 1
```

Dabei wird die Variable `x` automatisch zum Datentyp *numeric* deklariert. Das kann in `R` abgerufen werden:

```
> class(x)
[1] "numeric"
```

Auch ein Zeichen oder eine Zeichenkette lässt sich zuweisen. Hier ist zu beachten, dass ein Zeichen oder eine Zeichenkette immer in hochgestellten Anführungszeichen liegen muss:

```
> X <- "Hallo Welt"
```

Wie oben beschrieben, ist `R` case-sensitive, das heißt `x` und `X` sind verschiedene Variablen. Abrufen lässt sich der Wert einer Variablen durch Eingabe des Variablennamens und anschließendem *<ENTER>* auf der Tastatur. Das Doppelkreuz (#) in `R` muss einem folgenden Kommentar vorangestellt werden [9, S. 10-12]. Ein Beispiel:

```
> # Kommentar: X eingeben und <ENTER> drücken
> X
[1] "Hallo Welt"
```

3.3.2. Komplexe Datentypen

`R` besitzt eine Reihe von komplexen Datentypen, auch Strukturen genannt, die aus den einfachen Datentypen zu Objekten zusammengesetzt werden können. Nachfolgend werden *Vektoren*, *Matrizen*, *Listen* und *Data.Frames* beschrieben [28, S. 19].

Vektoren

Vektoren können einen der einfachen Datentypen darstellen, nicht aber mehrere zugleich. Das bedeutet, ein Vektor beinhaltet entweder nur numerische, zeichenketten-basierte (alphanumerische) oder logische Werte. Er wird durch die concatenate-Funktion `c()` gebildet [4, S. 14] [28, S. 31].
Ein Beispiel:

```
> num_vektor <- c(1,2,3,4,5,6,7,8,9,10)
> anum_vektor <- c("a","b","c")
> log_vektor <- c(TRUE,TRUE,FALSE)
```

Unzulässig wäre ein Vektor `vektor <- c(1,"a",TRUE)`, da hier verschiedene Datentypen benutzt werden.
Der Abruf eines Vektors geschieht, wie oben beschrieben, indem der Variablenname gefolgt von einem *<ENTER>* eingegeben wird. Wird nur ein bestimmtes Element aus einem Vektor benötigt, muss der Variablenname mit dem Index des Elementes in eckigen Klammern kombiniert werden [9, S. 20]. Beispielsweise soll aus dem Vektor `anum_vektor` nur das dritte Element abrufen werden:

```
1 > anum_vektor[3]
2 [1] "c"
```

Das Ergebnis ist `"c"`.

Matrizen

Matrizen in `R` sind, ähnlich wie Vektoren, zusammengefasste Elemente eines gleichen Datentyps. Sie sind im Unterschied zu Vektoren zweidimensionale Objekte, die wie eine Tabelle ihre Daten nach Spalten und Zeilen anordnen. Daher eignen sich Matrizen gut zur Darstellung von tabellarischen Datensätzen.
In `R` können Matrizen mit der Funktion `matrix()` gebildet werden. Um eine Matrix den erforderlichen Bedürfnissen anzupassen, können der Funktion mehrere Argumente übergeben werden [28, S. 75 ff.].
Die folgende Auflistung zeigt wichtige und häufig benutzte Argumente:

data = Nutzdaten (einfacher Datentyp oder Datenstruktur (z. B. Vektor))

nrow = Anzahl an Zeilen der Matrix.

ncol = Anzahl an Spalten der Matrix.

dimnames = Liste der Zeilen- und Spaltenüberschriften der Matrix

Der unten aufgeführte Aufruf erstellt in `R` eine 5x5 Matrix und füllt diese mit den Werten 1 bis 25:

```
> num_matrix <- matrix(data=c(1:25),nrow=5,ncol=5)
> num_matrix

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
```

Eine Kurzschreibweise, ohne die Argumentbezeichnungen ist ebenfalls möglich, jedoch wird die Lesbarkeit des Aufrufes gemindert: `num_matrix <- matrix(c(1:25),5,5)`

Im obigen Beispiel wird durch die Kurzschreibweise `c(1:25)` eine in der Vektorfunktion `c()` fortlaufende Zahlenfolge generiert. Der Doppelpunkt kommt in der Sprache `R` häufig zum Einsatz und dient als Separator zweier Werte, aus denen eine Sequenz gebildet werden soll. Ohne den Parameter `dimnames` werden die Zeilen und Spalten fortlaufend nummeriert; Zeilenbezeichnungen haben die Form [Zahl,] und Spaltenbezeichnungen [,Zahl]. Diese Art der Identifikation wird auch für den Zugriff auf einzelne Elemente genutzt. Der Abruf erfolgt nach dem Schema: `Matrixname[Zeilenindex,Spaltenindex]`, wobei die Indizes entweder Zahlen oder Vektoren sein können [9, S. 20] [28, S. 78 ff.].

Um dies zu verdeutlichen, greifen wir auf die obige Matrix `num_matrix` zurück und wollen auf das Element 20, welches in der Zeile 5 und Spalte 4 zu finden ist, zugreifen.

```
> num_matrix[5,4]
[1] 20
```

Wird eine ganze Zeile bzw. Spalte benötigt, gibt man nur den jeweiligen Zeilenindex bzw. Spaltenindex an. Zeile 1 der Matrix `num_matrix` erhält man demnach durch folgende Eingabe:

```
> num_matrix[1,]
[1]  1  6  11  16  21
```

Mehrere Zeilen oder Spalten gleichzeitig abzurufen, ist möglich, indem die entsprechenden Zeilen- oder Spaltenindizes in einem Vektor zusammenfasst werden. Beispielsweise lassen sich die Spalten 1 und 5 wie folgt ausgeben:

```
> num_matrix[,c(1,5)]

     [,1] [,2]
[1,]    1   21
[2,]    2   22
[3,]    3   23
[4,]    4   24
[5,]    5   25
```

Eine Matrix kann auch mit den beiden integrierten Funktionen `cbind()` bzw. `rbind()` erzeugt werden. Die `cbind()`-Funktion fügt Vektoren (oder andere Objekte) spaltenweise zusammen. `rbind()` hingegen, fügt sie zeilenweise zusammen. Dabei müssen die zu kombinierenden Vektoren/Objekte dieselbe Länge aufweisen. Um dies an einem Beispiel zu verdeutlichen, bedienen wir uns unseres bekannten Vektors `num_vektor`.
Aus dem eindimensionalen Vektor `num_vektor`

```
> num_vektor
[1]  1  2  3  4  5  6  7  8  9  10
```

wird eine 3x3 Matrix mit Hilfe der `cbind()`-Funktion erstellt:

```
> matrix_cbind <- cbind(num_vektor[c(1:3)],num_vektor[c(4:6)],
    num_vektor[c(7:9)])
> matrix_cbind

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Der `cbind()`-Funktion werden drei Vektoren/Objekte übergeben, wobei die Indexangabe `[c(1:3)]` nur die Elemente 1 bis 3 aus dem Vektor `num_vektor` auswählt. Die Angabe

`[c(4:6)]` bzw. `[c(7:9)]` filtern jeweils die Elemente 4 bis 6 bzw. 7 bis 9 heraus und fügen sie als Spalten der Matrix `matrix_cbind` an.
Mit `rbind()` erhält man dieselbe Matrix, mit dem Unterschied, dass die Elemente an der Hauptdiagonalen gespiegelt werden:

```
> matrix_rbind <- rbind(num_vektor[c(1:3)],num_vektor[c(4:6)],
   num_vektor[c(7:9)])
> matrix_rbind

     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Listen

Listen können im Gegensatz zu den oben genannten Strukturen (Vektoren, Matrizen) verschiedene Datentypen vereinen. Um eine Liste zu erstellen, wird in `R` die `list()`-Funktion verwendet. Die Objekte, die in der Liste zusammengefasst werden sollen, werden nacheinander der Funktion übergeben:

```
list(Objekt1,Objekt2,...)
```

Dabei kann optional ein Name für das Objekt mit angegeben werden:

```
list("Name1"=Objekt1,"Name2"=Objekt2)
```

Des Weiteren können Listen beliebig viele Objekte, unterschiedlicher Länge, enthalten [28, S. 117 ff.]. Als Grundlage für das nächste Beispiel, in der eine Liste erstellt wird, dienen die bereits zugewiesenen Vektoren `num_vektor`, `anum_vektor` und `log_vektor`.

```
> liste <- list("numerisch"=num_vektor,"alphanumerisch"=
   anum_vektor,"logisch"=log_vektor)
> liste

$numerisch
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
$alphanumerisch
[1] "a" "b" "c"

$logisch
[1]  TRUE  TRUE FALSE
```

Das Ergebnis ist eine Liste mit drei Objekten (siehe Zeile 3).

Der Zugriff auf ein Objekt erfolgt über doppelte eckige Klammern `[[...]]`.
Mit der Eingabe `liste[[1]]` erhält man die Elemente des ersten Objektes der Liste, hier der Vektor `num_vektor`:

```
1 > liste[[1]]
2 [1]  1  2  3  4  5  6  7  8  9  10
```

Falls ein Name innerhalb der Liste für das Objekt angegeben wurde, kann mit diesem ebenfalls auf das Objekt zugegriffen werden. Dies erfolgt in `R`, indem der Listenname vom Objektnamen durch ein Dollarzeichen ($) getrennt wird. Das gleiche Ergebnis in der Konsolenausgabe wird durch die Eingabe `liste$numerisch` realisiert [28, S. 119].

```
1 > liste$numerisch
2 [1]  1  2  3  4  5  6  7  8  9  10
```

Ein Element aus dem Objekt erhält man durch die bekannte Notation der eckigen Klammern `[...]`.

```
1 > liste$numerisch[1]
2 [1]  1
```

Data.Frames

Data.frames sind eines der wichtigsten Datenstrukturen in `R`. Sie haben einen tabellarischen Aufbau und kombinieren die Elemente von Listen und Vektoren. Dabei können unterschiedliche Datentypen gleicher Elementanzahl zu einem *data.frame* vereint werden. Sie stellen somit eine Verallgemeinerung von Matrizen dar. Die Kombination unterschiedlicher Datentypen in tabellarischer Auflistung findet häufig Verwendung in der

Datenerhebung für statistische Untersuchungen, die in R mittels *data.frames* optimal abgebildet werden können. Die Erstellung eines *data.frames* in R ähnelt der Syntax der Listen, mit dem Unterschied, dass der Funktionsname `data.frame()` genutzt wird [28, S. 122 f.].
Somit kann ein *data.frame* aus den oben definierten numerischen, alphanumerischen und logischem Vektor erstellt werden. Dabei ist zu beachten, dass der numerische Vektor aus zehn Elementen besteht und die beiden anderen jeweils aus drei Elementen. Um die Bedingungen zu erfüllen, muss man den numerischen Vektor verkürzen bzw. dem *data.frame* nur drei Elemente des Vektors übergeben, da die Länge aller Objekte stets einheitlich sein muss.

```
1 > datenframe <- data.frame("numerisch"=num_vektor[c(5,6,7)],"alphanumerisch"=
      anum_vektor,"logisch"=log_vektor)
2 > datenframe
3
```

	numerisch	alphanumerisch	logisch
1	5	a	TRUE
2	6	b	TRUE
3	7	c	FALSE

Unter dem Variablennamen `datenframe` ist der erstellte *data.frame* anzusprechen. Die drei eingefügten Vektoren sind in Spalten überführt worden. Die erste Zeile ist der sog. *header* und beinhaltet die Spaltenüberschriften. Jede nachfolgende Zeile definiert eine Datenzeile, beginnend mit der Zeilennummer, gefolgt von den eigentlichen Daten. Der Zugriff auf Zeilen, Spalten und einzelne Elemente erfolgt wie bei den Matrizen (`datenframe[Zeilennummer,Spaltennummer]`) oder durch Angabe der Spaltenüberschrift wie bei den Listen mit dem Dollarzeichen als Präfix (`datenframe$logisch`).

3.4. Funktionen

Funktionen bilden eine Programmcode-Einheit, die einen bestimmten Zweck erfüllen. Dabei besitzen Funktionen meist einen oder mehrere Übergabeparameter, führen damit Berechnungen aus und geben diese als Ausgabeparameter oder grafische Darstellungen zurück.

Um beispielsweise den Mittelwert wie im Kapitel 2 (Formel 2.1) mit bereits vorhandenen Funktionen zu berechnen, bietet R die Möglichkeit einzelne Elemente mittels der `sum()`-Funktion zu summieren und anschließend durch die Anzahl der Elemente mit Hilfe der `length()`-Funktion zu dividieren:

```
> sum(num_vektor) / length(num_vektor)
[1]  5.5
```

Vereinfachen lässt sich dies durch eine weitere integrierte Funktion. Man erhält den Mittelwert der Elemente 1 2 3 4 5 6 7 8 9 10 durch die Funktion `mean()`:

```
> mean(c(1:10))    # oder mean(num_vektor)
[1]  5.5
```

Hier wurde mit `c(1:10)` ein Vektor mit den Elementen 1 bis 10 erstellt und an die Funktion `mean()` übergeben, die den Mittelwert berechnet.

Wie dieses Beispiel zeigt, wurde hier eine Funktionskomposition angewandt, d.h. eine Verkettung von mehreren Funktionen `mean( c(...) )`.
Im Folgenden wird die Summe der Elemente 10 20 30 40 50 60 70 80 90 100 bestimmt.

```
> sum(seq(from=10,to=100,by=10))
[1]  550
```

Die innere Funktion `seq()` erzeugt eine Folge von Werten, hier beginnend (`from=`) mit der Zahl 10 und einer Schrittweite (`by=`) von 10 bis zum Maximalwert (`to=`) 100. Dieser Vektor wird von der äußeren Funktion `sum()` übernommen und summiert.

Eine weitere wichtige Funktion, um Datenanalysen durchzuführen, ist die `table()`-Funktion. Die Funktion zählt die Anzahl identisch vorkommender Elemente eines Datensatzes und erstellt eine Häufigkeitstabelle. Mit folgendem Befehl wird die absolute Häufigkeit der Elemente des Vektors `log_vektor` ermittelt.

```
> table(log_vektor)
log_vektor
   FALSE   TRUE
   1       2
```

Das Ergebnis liefert für das Element `FALSE` eine 1 und für `TRUE` den Wert 2 zurück. Diese beiden Zahlenwerte stellen die absolute Häufigkeit für das Vorkommen im Vektor `log_vektor` dar.

In Verbindung mit den vorgestellten *data.frames* kommt im Zuge der Untersuchungen die `subset()`-Funktion häufig zum Einsatz. Sie ermöglicht durch ihre Filterfunktion, basierend auf einer oder mehrerer Bedingungen, Teilmengen aus Datenstrukturen zu extrahieren. Angenommen für Analysezwecke werden nur Datensätze aus dem oben definierten *data.frame* `datenframe` benötigt, die den Wert `TRUE` besitzen. Dies kann erreicht werden, indem die `subset()`-Funktion mit einer entsprechenden Bedingung kombiniert wird.
Nachfolgende Zeilen zeigen den gefilterten *data.frame* `datenframe` an.

```
1 > subset(x=datenframe,subset=datenframe$logisch=="TRUE")
2
```

```
  numerisch alphanumerisch logisch
1         1              a    TRUE
2         2              b    TRUE
```

Zeile 1 enthält den `subset()`-Aufruf. Der Funktion werden hier zwei Argumente übergeben. Die erste Argumentenbezeichnung `x=` erwartet den Namen des zu filternden Objektes und der zweite Bezeichnung `subset=` enthält die Bedingung(en) für den Filter. In der Regel werden sämtliche Spalten des Quell-*data.frames* ausgegeben. Durch die Übergabe eines dritten Argumentes `select=`, könnte eine bestimmte Spalte für die Rückgabe bestimmt werden.

Nachfolgend sind wichtige, vordefinierte Funktionen aufgelistet, die im weiteren Verlauf dieser Arbeit genutzt wurden.

Funktion	Beschreibung
c()	Erstellung eines Vektors
t()	Transponieren einer Matrix bzw. eines data.frames
matrix()	Erstellung einer Matrix
rownames()	Abrufen oder Setzen der Zeilenbeschriftung eines Objektes
colnames()	Abrufen oder Setzen der Spaltenbeschriftung eines Objektes
data.frame()	Erstellung eines data.frames
subset()	Rückgabe einer Teilmenge eines Datentyps, die die Bed. erfüllen
unique()	Entfernung von doppelten/mehrfachen Elementen
table()	Erstellung einer (absoluten) Häufigkeitstabelle
length()	Ermittlung der Länge
sum()	Summierung aller Elemente des Datentyps
seq()	Generierung einer gleichmäßigen Sequenz
min()	Rückgabe des minimalen Wertes einer Zahlenreihe
max()	Rückgabe des maximalen Wertes einer Zahlenreihe
mean()	Bildet das arithmetische Mittel einer Zahlenreihe
var()	Ermittlung der Varianz einer Zahlenreihe
sd()	Bildet die korrigierte Standardabweichung einer Zahlenreihe
read.table()	Einlesen einer Datei in ein data.frame

Tab. 3.1.: Wichtige Funktionen in R

3.4.1. Grafikfunktionen

`R` besitzt eine große Anzahl bereits implementierter Befehle, um entsprechend aufbereitete Daten anschaulich zu präsentieren. Ergebnisse können u.a. in Form von Balkendiagrammen, Boxplots, Histogrammen, Karten oder Kreisdiagrammen veranschaulicht werden. In den Auswertungen dieser Arbeit kommt überwiegend die Grafikfunktion für Balkendiagramme und die allgemeine Plot-Funktion für die Kartenvisualisierung zum Einsatz. Zusätzlich bietet `R` weitere nützliche Befehle, die die Grafiken durch Textzusätze, Linien, Legenden, Farben und anderen Attributen vervollständigen. Im Folgenden wird auf einige dieser Funktionen, am Beispiel des Vektors `num_vektor`, eingegangen.

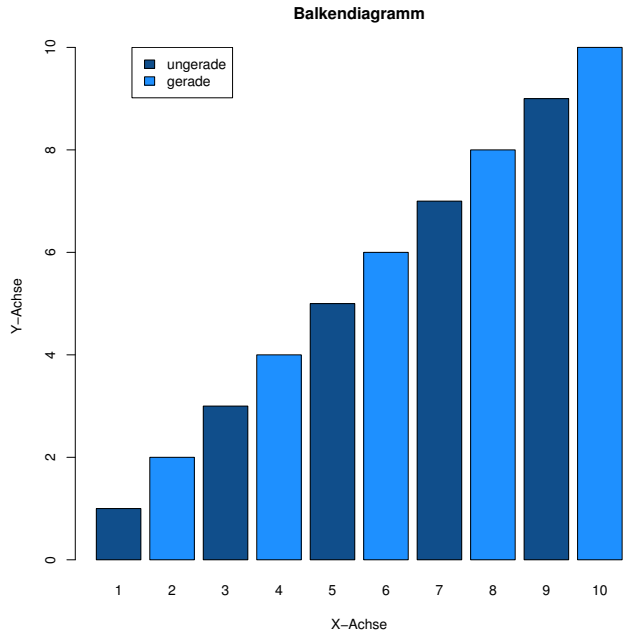

Abb. 3.1.: Die barplot()-Funktion

Das in Abbildung 3.1 dargestellte Balkendiagramm ist mit dem `barplot()`-Befehl und dessen Parametern erzeugt worden. Zur Vervollständigung der Grafik ist eine Legende mit Hilfe der `legend()`-Funktion eingefügt.

```
1 > barplot(height=num_vektor,main="Balkendiagramm",axes=TRUE,
    horiz=FALSE,col=rep(c("dodgerblue4","dodgerblue")),xlab=
    "X-Achse",ylab="Y-Achse",names.arg=num_vektor)
2 > legend(x=1,y=10,legend=c("ungerade","gerade"),
    fil=c("dodgerblue4","dodgerblue"))
```

Zeile 1 enthält den Aufruf der `barplot()`-Funktion und dessen Argumente. Das Argument `height=` enthält die Daten eines Vektors oder einer Matrix, die die Größe der Balken beeinflussen. `main=` legt den Titel des Diagramms fest, `horiz=` ist ein boolescher Wert und bestimmt ob die Balken horizontal oder vertikal ausgegeben werden. `axes=` ist ebenfalls vom Typ Bool und beeinflusst die Anzeige der Achsen. Das Argument `col=` wird benutzt um die einzelnen Balken farblich zu kennzeichnen. In diesem Beispiel werden mit der repeat-Funktion `rep()` die beiden Farben `dodgerblue4` und `dodgerblue`

alternierend wiederholt. `xlab=` bzw. `ylab=` bezeichnen jeweils die Achsenbeschriftung und das Argument `names.arg=` setzt einen Text unter jeden Balken. In diesem Fall wird der `num_vektor` angegeben, wodurch die Zahlen 1 bis 10 für die Beschriftung genutzt werden. Mit der zweiten Zeile wird an die vorhandene Darstellung eine Legende hinzugefügt. `x=` bzw. `y=` sind die Koordinatenangaben für die Legende auf der Darstellung. Dem Argument `legend=` wird ein String-Vektor mit den Namen `ungerade`/`gerade` und `fil=` ein Vektor mit den Farben `dodgerblue4`/`dodgerblue` übergeben.

Funktion	Beschreibung
barplot()	Erzeugung eines Balkendiagramms
plot()	Funktion um Objekte in `R` darzustellen
hist()	Erzeugung eines Histogramms
pie()	Erzeugung eines Kreisdiagramms
axis()	Fügt eine Achse der aktuellen Darstellung hinzu
text()	Fügt einen Text der aktuellen Darstellung hinzu
lines()	Fügt eine Linie der aktuellen Darstellung hinzu
par()	Setzen von grafischen Parametern
legend()	Erstellung einer Legende

Tab. 3.2.: Wichtige Grafikfunktionen in R

3.4.2. Eigene Funktionen

Für bestimmte Untersuchungen werden eigene Funktionen in `R` definiert und eingesetzt. Funktionen verkürzen den Quellcode, da wiederkehrende Aufgaben mittels eines einfachen Funktionsaufrufes ausgeführt werden können. Weiterhin werden Redundanzen im Quelltext vermieden und die Übersichtlichkeit wird durch die Modularisierung erhöht. Neben diesen Vorteilen bieten Funktionen auch eine bessere Wartbarkeit, d.h. bei etwaigen Änderungen müssen Programmteile nur an einer Stelle geändert werden und der restliche Quellcode bleibt davon unberührt. Gleichzeitig verringert sich das Fehlerrisiko bei Änderungen.

In R können eigene Funktionen mit dem Befehl `function() { }` definiert werden:

```
Funktionsname <- function(Argument1,Argument2,...) {
  Anweisung1
  Anweisung2
  ...
  return(ergebnis)
}
```

Um die allgemeine Schreibweise zu verdeutlichen, wird im Folgenden eine Funktion in R geschrieben, die von einem bestimmten Wert den prozentualen Anteil berechnet:

```
1 > prozente <- function(basis,prozentzahl) {
2       ergebnis <- basis * prozentzahl / 100
3       return(ergebnis)
4 }
```

Zeile 1 beinhaltet den Funktionsnamen `prozente()`, das Schlüsselwort `function`, um R mitzuteilen, dass eine eigene Funktion definiert wird, sowie die Übergabeparameter `basis` und `prozentzahl`. Mit der geöffneten, geschweiften Klammer beginnt der Funktionsrumpf, in der die Anweisungen stehen.
Zeile 2 enthält die eigentliche Berechnung. Der Variablen `ergebnis` wird der berechnete Wert zugewiesen, der sich aus

$$\frac{basis * prozentzahl}{100}$$

ergibt.
Zeile 3 gibt das Ergebnis mittels `return()` zurück.
Zeile 4 schließt mit der geschweiften Klammer den Funktionsrumpf und beendet die Funktion.
Die eigene Funktion wird in der R-Konsole aufgerufen mit:

```
1 > prozente(200,25)
2 [1] 50
```

Als Ergebnis liefert die Funktion den Wert 50 zurück, da 25% von 200 = 50 entsprechen.

3.5. Datenimport

Ein Datenbestand, der mit Hilfe von R untersucht werden soll, kann in Form von externen Dateien vorliegen. Dieser muss vor Beginn der Datenanalyse korrekt eingelesen werden. Je nach Datenaufbereitung können die Datensätze verschiedene Strukturen und Dateiformate aufweisen. Für häufig verwendete Formate wie Text- oder CSV-Dateien bietet R die mitgelieferten Funktionen `read.table()` bzw. `read.csv()`. Diese Funktionen ermöglichen den Datenimport in die richtigen Datenstrukturen. Der folgende R-Befehl ist ein typischer Aufruf mit vier Argumenten um eine Text-Datei zu importieren.

```
1 > datensatz <- read.table(file="/PFAD/DATEINAME.txt",header=
    TRUE,sep=" ",fileEncoding="latin1")
```

Das erste Argument `file=` bezeichnet den vollständigen Pfad und Namen der einzulesenden Datei. `header=` ist ein boolescher Wert, der angibt, ob die erste Zeile Spaltenbeschriftungen enthält. Das Argument `sep=` übermittelt R, welches Zeichen die einzelnen Werte innerhalb einer einzulesenden Zeile trennt. In diesem Beispiel ist das Leerzeichen als Separator bestimmt worden. Häufig verwendete Trennzeichen sind Semikolon (`;`), Kommata (`,`) oder Tabulatoren (`\t`).

3.6. Pakete

Wie in Kapitel 3.1 erwähnt, werden mit den Zusatzpaketen neue Funktionen in die R-Umgebung nachgeladen. Alle Zusatzpakete, d.h. Nicht-Standard-Pakete, muss der Benutzer menügeführt in der R-Umgebung oder im Kommando-*Prompt* manuell herunterladen und aktivieren [28, S. 16 f.].
Im Kommando-*Prompt* kann dies wie folgt durchgeführt werden:

```
1 # Pakete aus dem CRAN herunterladen:
2 > Install.packages("Paketname")
3 # Pakete aktivieren:
4 > Library(Paketname)
```

Im Rahmen dieses Projektes werden zwei Zusatzpakete für Analysezwecke benötigt. Zum einen das Paket *stringr*, welches spezielle Routinen zur String-Bearbeitung zur Verfü-

gung stellt und zum anderen das Paket *maptools*. Das Paket *maptools* enthält Befehle, um in R den Umgang mit vektorbasierten geografischen Daten wie Punkten, Linien und Polygonen zu ermöglichen. Speziell das Einlesen und Schreiben von *ESRI shapefiles* wird durch dieses Paket unterstützt [1, S. 1].
Shapefile ist ein Datenformat, welches von der Firma *ESRI* entwickelt wurde, zur Aufbereitung von Geodaten, die in den verschiedensten GIS-Systemen zur Anwendung kommen [11]. Das *Shape*-Format besteht aus mindestens drei Einzeldateien:

Dateiname	Beschreibung
.shp	Mainfile (Geometriedaten)
.dbf	dBASE table (Attribute/Sachdaten)
.idx	Index file (Verknüpfung der Attribute mit den Geometriedaten)

Das Paket *maptools* besitzt Paketabhängigkeiten zu anderen Paketen. Um *maptools* aktivieren zu können wird das Paket *sp*, ein Paket welches Klassen und Methoden für räumliche- bzw. geografische Daten liefert [18, S. 1], sowie das Paket *rgeos*, welches ein Interface für die Geometry Engine Open Source in R bereitstellt [2, S. 1], benötigt. Ob die beiden Pakete automatisch oder manuell installiert werden, hängt von der Betriebssystem-Plattform ab. Beide Pakete sind ebenfalls über das *CRAN* verfügbar.

In den folgenden Kapiteln wird *maptools* benutzt, um Auswertungsergebnisse in Form einer geografischen Landkarte, basierend auf den deutschen Postleitzahlen, übersichtlich darzustellen und einprägsamer zu machen, als es zum Beispiel Tabellen ermöglichen. Dabei werden die Auswertungsergebnisse auf der geografischen Karte farblich codiert, sodass ein Farbspektrum entsteht. Eine Landkarte dieser Art wird auch als *Heatmap* bezeichnet. Markante Ergebnisse, aus großen Datenmengen, werden dem Betrachter anschaulich und schnell vermittelt. In R wird mit Hilfe der *maptools*-Funktion `readShapeSpatial()` ein *shapefile* eingelesen und mit der anschließenden R-Funktion `plot()` in einem Fenster ausgegeben:

```
1 > shapefile <- readShapeSpatial(fn="deutschland.shp")
2 > farben <- rep("white",16); farben[14] <- "dodgerblue"
3 > plot(x=shapefile,col=farben)
```

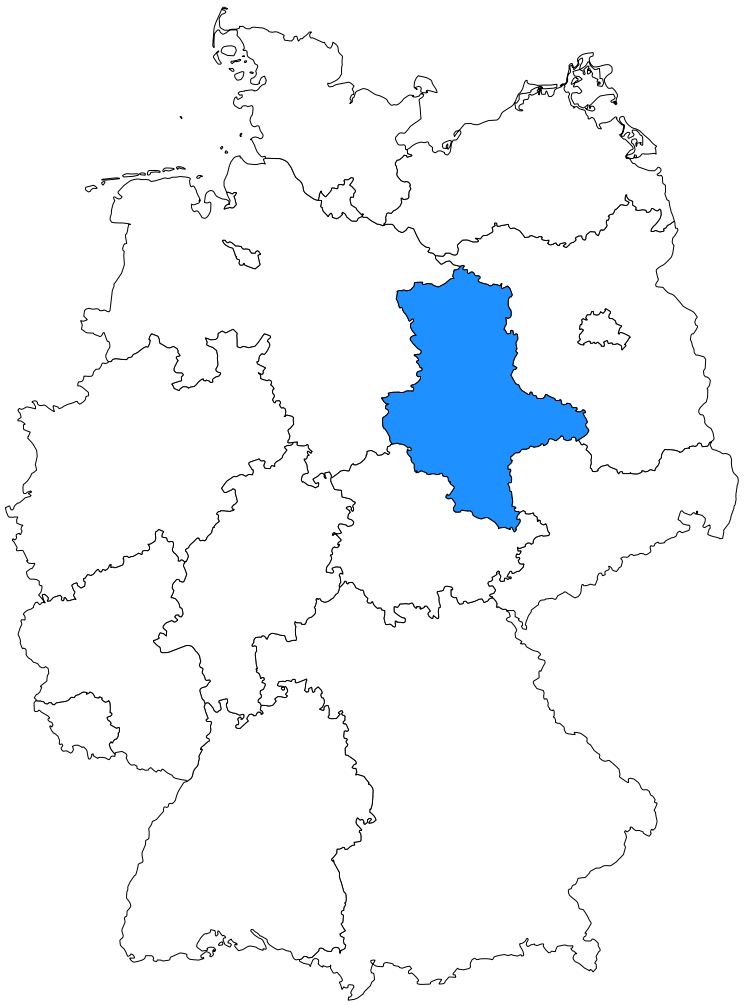

Abb. 3.2.: Bundesländer von Deutschland

Das Ergebnis ist eine Deutschlandkarte, deren Polygone die Bundesländergebiete begrenzen. Durch die Datenbankdatei (*dBASE Table*), deren Inhalt mit den Polygonen verknüpft ist, kann `R` auf die Polygone der einzelnen Bundesländergebiete zugreifen und sie farblich kennzeichnen. In diesem Fall wurde das vierzehnte Polygon, welches dem Bundesland Sachsen-Anhalt entspricht, blau eingefärbt.

4. Datenbestand der AOK-Sachsen-Anhalt

4.1. Struktur und Charakteristik

Grundlage und Basis der folgenden Analysen ist der von der Allgemeinen Ortskrankenkasse in Sachsen-Anhalt zur Verfügung gestellte Datenbestand. Der Datenbestand beinhaltet Daten der AOK-Versicherten für das Jahr 2012 und ist eine Zusammenstellung ambulanter Behandlungen, festgestellter Diagnosen und verordneter Medikamente. Weiterhin umfasst der Datenbestand spezifische Informationen zum Alter, Geschlecht, Wohnort und dem Versicherungsstatus. Um die Datenschutzrichtlinien bezüglich der Anonymität einzuhalten, wurde den einzelnen Versicherten und den behandelnden Ärzten eine eindeutige und zufällig vergebene Pseudonymnummer zugeordnet, die keine Rückschlüsse auf deren tatsächliche Identität zulässt.
Festgestellte Diagnosen vom behandelnden Arzt werden dem Patienten durch einheitlich festgelegte Diagnoseschlüssel (ICD-10-Codes) zugeordnet. Das ICD-10-System ist ein internationales statistisches Klassifikationssystem von Krankheiten und verwandter Gesundheitsproblemen, in dem jede bekannte Erkrankung einem unären Code aus der Kombination von Buchstaben und Zahlen zugewiesen ist [6].
Die verschiedenen Medikationen sind im Datenbestand durch die Pharmazentralnummer (PZN) und die darin enthaltenen Arzneistoffe durch das sog. Anatomisch-Therapeutisch-Chemische Klassifikationssystem (ATC) festgehalten [5] [29]. Die Pharmazentralnummer ist eine Kennziffer zur Identifizierung von Arzneiprodukten bestimmter Bezeichnung, Packungsgröße und Darreichungsform eines bestimmten Anbieters [16]. Im ATC-Klassifikationssystem werden im Gegensatz zum PZN-System nicht die Arzneimittel, sondern die einzelnen Arzneiwirkstoffe klassifiziert.

Im Detail bestehen die Daten aus vier Textdateien und zwei Excel-Tabellen, die für die Verarbeitung in `R` vorher in das CSV-Format konvertiert wurden. Eine Verknüpfung zwischen den Textdateien kann durch die Pseudonymnummer der Versicherten hergestellt werden.
Im Folgenden wird auf jede Datei eingegangen und deren Inhalt beschrieben:

1. `Wohlfahrt_Stammdaten.txt`

PSEUDONYM	TODESTAG	PLZ	STADT	VERSICHERUNGSART
1000008245	-	6184	Kabelsketal	arbeitslos
1000050389	-	39106	Magdeburg	beschäftigt
1000010776	-	39576	Stendal	Familienangehörige
1000040684	-	39326	Wolmirstedt	freiwillig versichert
1002065958	-	6124	Halle	nicht_GKV_Versichert
1002145759	-	29410	Salzwedel	Rehabilitanten
1010846848	-	6369	Zabitz	Rentenantragsteller
1000996630	23.11.2012	6120	Halle	Rentner
1105049551	-	7318	Saalfeld	unbekannt

Die Datei `Wohlfahrt_Stammdaten.txt` enthält fünf Spalten und 605.924 Zeilen, wobei in jeder Zeile mehrere Attribute zu einem AOK-Versicherten gespeichert sind. Spalte eins enthält in der Regel eine 10-stellige Pseudonymnummer, gefolgt von der Spalte *TODESTAG*, die generell ein Minuszeichen enthält oder, falls der Versicherte verstorben ist, das Sterbedatum. Die Felder *PLZ* und *STADT* enthalten Angaben über den Wohnort der Versicherten, mit welchem ein regionaler Bezug für sämtliche Analysen möglich ist. Die letzte Spalte gibt den Versicherungsstatus des Versicherten an. In obiger Tabelle sind alle vorkommenden Versicherungsarten gelistet.

2. `Wohlfahrt_ambuDaten.txt`

PSEUDONYM	QUARTAL_AMBULANT	ICD	ARZT	ARZTGRUPPE	AGE	GENDER
1000008245	20122	E6609	3545239554	01	31	F
1000010776	20123	Z309	2117519261	15	49	F
1000018755	20124	M779	193719836	10	76	M
1000011946	20123	B99	813953709	02	25	M
1836801090	20121	A099	1500531565	01		?

Die `Wohlfahrt_ambuDaten.txt`-Datei umfasst sieben Spalten und insgesamt 27.446.591 Zeilen, wobei die erste Spalte wieder den einzelnen Versicherten durch die Pseudonymnummer repräsentiert. Im Unterschied zu den oben beschriebenen Stammdaten, können hier die Pseudonymnummern mehrfach vorkommen, da jede einzelne ambulante Behandlung eines Versicherten im Jahr 2012 protokolliert wurde. Die folgenden Spalten geben Auskunft über das Quartal, in dem die ambulante Behandlung stattgefunden hat, den Diagnoseschlüssel der Erkrankung, die pseudonymisierte Arztkennziffer, die Gruppenzugehörigkeit des Arztes (siehe `Daten_wohlfahrt.txt`), das Alter des Patienten im Jahr 2012 und sein Geschlecht.

3. `Wohlfahrt_Arzneimittel.txt`

PSEUDONYM	ARZT	VDATUM	ATC	PZN	PZN_BEZEICHNUNG	PACKUNGEN	AUTIDEM
1000018755	2871553790	14.12.12	M01AH05	02760985	Arcoxia 90mg	0,1	1
100001485	1755068488	16.04.12	B01AC06	07402210	Ass 100 Hexal	0,1	1
1000098913	1487560204	05.04.12	A02BC01	00233052	Omep 20mg	0,1	0
1000010776	2117519261	20.09.12	P01AB01	06155525	Vagimid 500	0,1	1

In dieser Datei sind alle verordneten Arzneimittel jedes Versicherten aufgelistet. Sie beinhaltet acht Spalten und besitzt 9.281.867 Einträge. Beginnend mit der Versicherten- und Arztpseudonymnummer, folgt das Verordnungsdatum, der ATC-Code, die PZN-Nummer und deren Bezeichnung. Die Spalte *PACKUNGEN* gibt die Packungsgröße des verordneten Medikamentes an, wobei der Wert mit 10 multipliziert werden muss. Die letzte Spalte *AUTIDEM* legt fest, ob das vom Arzt verschriebene Medikament durch ein gleichwertiges Arzneimittel ersetzt werden darf. Dabei steht die 1 für ersetzungsfähig.

4. `Daten_wohlfahrt.txt`

AGS	NAME
00	unbekannte/keine Fachgruppe
01	Allgemeinmediziner (Hausarzt)
02	Arzt/Praktischer Arzt (Hausarzt)
03	Internist (Hausarzt)
04	Anästhesiologie
05	Augenheilkunde
06	Chirurgie

Die Arztgruppe, die in der Datei `Wohlfahrt_ambudaten.txt` für jeden Arzt eingetragen ist, wird mit Hilfe dieser Datei entschlüsselt, wodurch der Arzt einem bestimmten Fachbereich zugeordnet werden kann. Die Datei enthält 100 Schlüsselnummern, von 00 bis 99.

5. `pzn-kosten-tabelle.xlsx`

Abrechnungspositionsnummer	Preis
01379177	5,34 €
03800770	6,25 €
01379208	6,65 €
04978607	6,97 €
02477924	9,96 €
02477930	11,01 €
02477953	11,97 €
04978613	14,79 €

In der Excel-Tabelle `pzn-kosten-tabelle.xlsx` befinden sich 40.162 Zeilen mit den Spalten *Abrechnungspositionsnummer* und *Preis*, wobei jede *Abrechnungspositionsnummer* einem PZN-Code entspricht. Jeder dieser Codenummern ist ein eindeutiger Preis zugeordnet, wodurch die verordneten Arznei- und Hilfsmittel preislich erfasst sind.

6. `Priscus-Liste_Joerg.xlsx`

Wirkstoff	ATC	Information
Indometacin	M01AB01	
Indometacin (am Auge)	S01BC01	nicht mehr verwendet
Indometacin (Gabe auf die Haut)	M02AA23	Schmerzgel
Acemetacin	M01AB11	
Ketoprofen	M01AE03	
Dexketoprofen	M01AE17	Schmerzmittel (systemisch angewendet)
Ketoprofen (Gabe auf die Haut)	M02AA10	Schmerzgel
Phenylbutazon	M01AA01	keine arzneiliche Verwendung

Diese Datei beinhaltet 97 verschiedene ATC-Codes mit Angabe der Wirkstoffbezeichnungen. Die dritte Spalte enthält eventuelle Zusatzinformationen zu den Wirkstoffen. Die in dieser Tabelle gelisteten Medikamente bilden die sog. *PRISCUS*-Liste und umfassen potenziell inadäquate Medikationen für ältere Menschen [13].

4.2. Implementierung und Bereinigung

In Vorbereitung auf das nächste Kapitel widmet sich dieser Abschnitt mit dem Import und der Aufbereitung des Datenmaterials. Ausgehend von der Annahme, dass größere Datenbestände partiell Fehler bzw. Inkonsistenzen aufweisen, sind die Eingabedateien nach dem Import in *R* zu untersuchen [21, S. 2]. Das Ziel dieser Maßnahme soll, nach Abschluss der Bereinigung, einen konsistenten Datenbestand ergeben.
Abhängig vom Inhalt der Dateien werden nach dem Einlesen in ein *data.frame*, entsprechend dem Kapitel 3.5 mit der `read.table()`-Funktion, verschiedene Untersuchungen durchgeführt.
Beginnend mit der `Wohlfahrt_Stammdaten.txt`-Datei, sind die AOK-Versicherten, die durch das Feld *PSEUDONYM* abgebildet werden, auf Inkonsistenzen zu analysieren. Die Überprüfung ergibt, dass 41 der über 600.000 Pseudonymnummern doppelt vorkommen. Aufgrund der Tatsache, dass bei diesen Nummern sich die restlichen Spalteneinträge un-

terscheiden, wird der Datenbestand um alle 82 Pseudonyme bereinigt, da für diese keine Eindeutigkeit vorliegt. Ein Beispiel für die Pseudonymnummer "941508714":

PSEUDONYM	TODESTAG	PLZ	STADT	VERSICHERUNGSART
941508714	-	6847	Dessau-Roßlau	Familienangehörige
941508714	-	6618	Naumburg	beschäftigt

Die zweite Untersuchung an den Stammdaten bezieht sich auf das Feld *VERSICHERUNGSART*. Bei der Überprüfung mit Hilfe der `table()`- und `length()`-Funktion wird festgestellt, dass alle Einträge eine der neun zulässigen Arten entsprechen. Somit besteht der *data.frame* nach der Bereinigung aus 605.842 Zeilen. Die Untersuchung der Postleitzahlen ergibt, dass nicht alle vom Typ *numeric* sind, da auch teilweise ausländische Postleitzahlen mit Leerzeichen und Buchstaben enthalten sind.

```
1 # Einlesen der Wohlfarth_Stammdaten.txt in R
2 W_Sd <- read.table(file="Wohlfarth_Stammdaten.txt",stringsAsFactors=FALSE,
      header=TRUE,sep="\t",fileEncoding="latin1")
3 # Den Spalten den korrekten Datentyp zuweisen
4 W_Sd$VERSICHERUNGSART <- as.character(W_Sd$VERSICHERUNGSART)
5 W_Sd$PLZ <- as.numeric(as.character(W_Sd$PLZ))
6 # Doppelte Pseudonyme ermitteln
7 doppel_pseudo <- W_Sd[which(duplicated (W_Sd[,1])),1]
8 # Doppelte Pseudonyme entfernen
9 stammdaten <- W_Sd[!W_Sd$PSEUDONYM %in% doppel_pseudo,]
```

Für die Kontrolle der `Wohlfahrt_ambuDaten.txt` werden die Felder *PSEUDONYM*, *AGE* und *GENDER* herangezogen. Da in dieser Datei der Versicherte, dessen Alter und Geschlecht erfasst sind, wird überprüft, ob zu einer bestimmten Pseudonymnummer divergierende Angaben auftreten. Die Kombination der drei Felder in Verbindung mit der bereits bekannten `table()`-Funktion und der `unique()`-Funktion, die mehrfach vorkommende Datenzeilen entfernt, ist folgendes Resultat vorzufinden: Insgesamt 117 AOK-Versicherte weisen unterschiedliche Angaben in den Feldern *AGE* und *GENDER* auf. Im unteren Auszug wird ein(e) Versicherte(r) irrtümlich als Frau und Mann; eine weitere Person mit zwei verschiedenen Altersangaben geführt.

PSEUDONYM	AGE	GENDER
845317006	37	F

845317006	37	M
3809132080	11	F
3809132080	6	F

Um einen möglichst konsistenten Datenbestand zu erhalten, werden die insgesamt 117 Pseudonymnummern entfernt. Wie im vorherigen Kapitel beschrieben, können die Pseudonymnummern in dieser Datei wiederholt auftreten, sodass der Datenbestand um 9629 Zeilen reduziert wird.
Weitere Auffälligkeiten, die sich im Feld *AGE* durch eine fehlende Altersangabe und im Feld *GENDER* durch den Eintrag eines "?" zeigen, wirken sich nicht auf die Aufbereitung der Daten aus, da es sich hier nicht um eine Doppeldeutigkeit handelt. Die letzte Änderung, die für die Analysen notwendig ist, betrifft die Spalte *ICD*. Hier fällt auf, dass der ICD-Code teilweise gekürzt angegeben ist, wenn es sich bei den letzten Stellen um Nullwerte handelt. Um im weiteren Verlauf mit dem Feld *ICD* arbeiten zu können, ist es notwendig, dass jeder ICD-Code 5-stellig ist. Mit den String-Funktionen `substr()` und `paste()` wird in `R` jeder Eintrag ergänzt.

```
# Einlesen der Wohlfarth_ambuDaten.txt in R
ambudaten_laden <- function() {
W_Ambu <- read.table(file="Wohlfahrt_ambuDaten.txt",header=TRUE,
    stringsAsFactors=FALSE,sep="\t",dec=",")
# Die Spalten PSEUDONYM, AGE, GENDER werden extrahiert
tmp_ambu <- W_Ambu[c(1,6,7)]
# Divergierendes Alter und Geschlecht für ein Pseudonym ermitteln
tmp_doppel1 <- data.frame(table("PSEUDONYM"=unique(tmp_ambu)[,1]))
tmp_doppel2 <- subset(tmp_doppel1,tmp_doppel1$Freq > 1)
# Löschen der doppeldeutigen Pseudonym-Nummern
ambudaten <- W_Ambu[!W_Ambu$PSEUDONYM %in% tmp_doppel2$PSEUDONYM,]
# ICD Codes müssen 5-stellig sein
ambudaten$ICD <- substr(x=paste(ambudaten$ICD,"0000",sep=""),start=1,stop=5)
# Rückgabe der bereinigten AmbuDaten
return(ambudaten)
}
```

Eine Bereinigung der restlichen Dateien (`Wohlfahrt_Arzneimittel.txt`, `Daten_wohlfahrt.txt`, `pzn-kosten-tabelle.xlsx`, `Priscus-Liste_Joerg.xlsx`) ist nicht notwendig. Diese können direkt importiert und durch einen Funktionsaufruf geladen werden.

```
# Einlesen der Wohlfarth_Arzneimittel.txt in R
arzneimittel_laden <- function() {
  arznei <- read.table(file="Wohlfarth_Arzneimittel.txt",stringsAsFactors=
      FALSE,header=TRUE,sep="\t",dec=",")
  return(arznei)
}

# Einlesen der Priscus-Liste_Joerg.csv in R
priscus_laden <- function() {
  Priscus_CSV <- read.table(file="Priscus-Liste_Joerg.csv",stringsAsFactors=
      FALSE,header=TRUE,sep=";")
  # Spalte ATC Codes auswählen
  priscus_liste <- data.frame("ATC"=Priscus_CSV$ATC)
  # Rückgabe der ATC-Codes der Priscus-Liste
  return(priscus_liste)
}

# Einlesen der pzn-kosten-tabelle.csv in R
pznkosten_laden <- function() {
  PZN_Preise <- read.table(file="pzn-kosten-tabelle.csv",stringsAsFactors=
      FALSE,header=TRUE,sep=";",dec=",")
  # Spaltenbeschriftung hinzufügen
  colnames(PZN_Preise) <- c("PZN","Preis")
  # Rückgabe der PZN_Preise
  return(PZN_Preise)
}
```

4.3. Voranalyse

In diesem Abschnitt wird erläutert, wie auf Grundlage des Datenbestandes ein regionaler Bezug hergestellt werden kann. Eine Möglichkeit besteht darin, die Spalte *STADT* in den Stammdaten als Ausgangspunkt zu nehmen und Stadt/Dorf-basierend die Analysen durchzuführen. Eine andere, mit *maptools* grafisch eleganter zu visualisierende Methode, sind die Postleitzahlen, die ebenfalls in den Stammdaten enthalten sind. Diese Variante wurde bevorzugt, sodass sie das Fundament für alle Auswertungen bilden. Auch ist eine Kombination aus Postleitzahlen und Städten möglich, allerdings wären nur mehrere Postleitzahlen zu einer Stadt zusammengefasst, welches nicht zu einem wesentlich besserem Resultat führen würde.

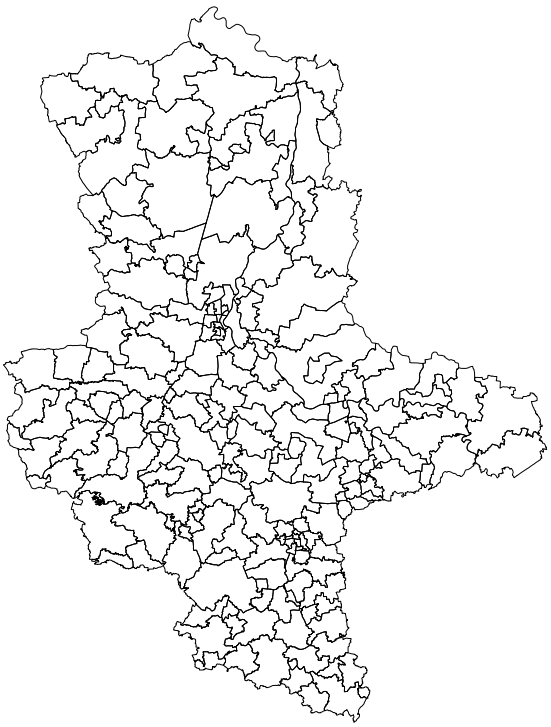

Abb. 4.1.: Postleitzahlengebiete von Sachsen-Anhalt

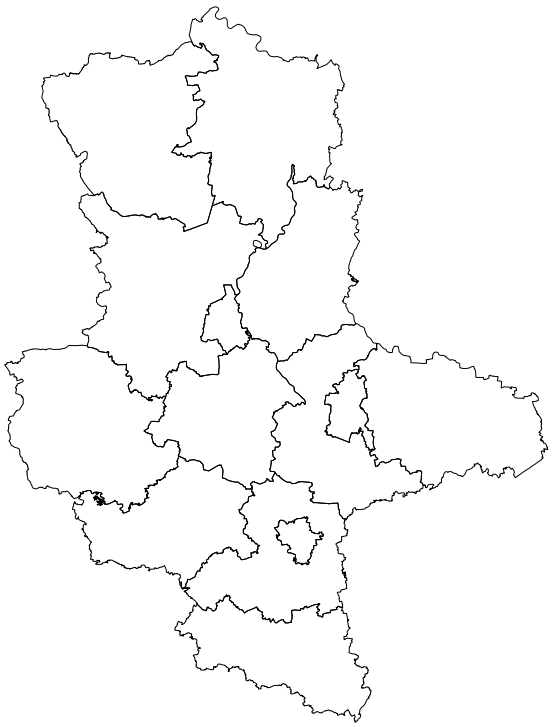

Abb. 4.2.: Landkreise/kreisfreie Städte von Sachsen-Anhalt

Für die im nächsten Kapitel erstellten Analysen wird Sachsen-Anhalt entweder auf Basis der einzelnen Postleitzahlengebiete (Abbildung 4.1) oder anhand der elf Landkreise und drei kreisfreien Städte (Abbildung 4.2) gegliedert. Je nach Untersuchung wird eine der beiden Varianten bevorzugt. Zum Beispiel kann es sinnvoll sein, die zu untersuchenden Datensätze nach den Landkreisen zu gruppieren, wenn festgestellt wird, dass eine zu geringe Anzahl an Datensätzen pro Postleitzahlgebiet entsteht und somit das Untersuchungsergebnis kein aussagekräftiges Resultat liefern würde. Dadurch ist sichergestellt, dass der Analyse immer eine gewisse Grundmenge pro Region zu Grunde liegt.

Es folgt der Quellcode für die Gruppierung der Regionen nach den Landkreisen/kreisfreien Städten, die in R als String-Vektoren mit ihren Postleitzahlbereichen abgebildet werden.

```
# Postleitzahlen für Landkreise
plz_magdeburg              <- c("39104-39130")
plz_halle                  <- c("06108-06132")
plz_dessau_rosslau         <- c("06842-06844","06846-06853","06855-06862")
plz_altmarkkreis_salzwedel <- c("29410-29416","38486-38489","39619","39624",
                                "39638","39649")
plz_stendal                <- c("14715","39517-39615","39629")
plz_boerde                 <- c("39164-39171","39179","39326-39397","39646")
plz_jerichower_land        <- c("39175","39245","39279","39288-39319")
plz_harz                   <- c("06458","06484-06495","06497-06502",
                                "38820-38880","38883-38899")
plz_salzlandkreis          <- c("06406-06430","06433-06457","06459-06469",
                                "39217-39240","39249","39418-39448")
plz_anhalt_bitterfeld      <- c("06366-06388","06749-06753","06755-06758",
                                "06760-06766","06768-06771","06774-06780",
                                "06792-06809","39261-39264")
plz_wittenberg             <- c("06772-06773","06785","06868-06870",
                                "06872-06895","06897-06917","06919-06920",
                                "06923-06925")
plz_mansfeld_suedharz      <- c("06295-06299","06301-06320","06322-06329",
                                "06331-06347","06526-06543")
plz_saalekreis             <- c("06179-06202","06204-06253","06255-06279")
plz_burgenlandkreis        <- c("06618-06629","06631-06650","06654-06693",
                                "06695-06723","06728-06729")
```

Die Datenbestandsbereinigung, die Landkreisdefinitionen auf Basis der Postleitzahlen und andere wichtige, oft wiederkehrende Befehle und Funktionen sind in einer R-Datei zusammengefasst. Diese Datei, mit dem Namen `initial.R`, wird für jede Auswertung benötigt und automatisch geladen. Nachfolgend wird auf die noch nicht erwähnten Bereiche dieser Datei eingegangen. Die komplette `initial.R` kann im Quelltext-Anhang (Seite 170) nachgeschlagen werden.

```
# String-Vektor mit den Landkreisnamen
landkreise <-c("magdeburg","halle","dessau_rosslau","altmarkkreis_salzwedel",
    "stendal","boerde","jerichower_land","harz","salzlandkreis","anhalt_bit
    terfeld","wittenberg","mansfeld_suedharz","saalekreis","burgenlandkreis")
```

```
4  # Farben für Balkendiagramme (Blau)
5  farben <- c("#0a519c","#3182bd","#6baed5","#b8d5e7","#e4ebfa")
6  # Farben für Balkendiagramme (Regenbogen)
7  farben_2 <- c("#A50026","#D73027","#F46D43","#FDAE61","#FEE090","#FFFFBF","#
       BFF0B9","#74C476","#37924F","#E0F3F8","#74ADD1","#4575B4","#373ca1","#080
       e72","black")
8  # Farben für die Shape-Datei / Heatmap
9  farben_shape <- c("#F2F2F2","#fff5f0","#fee0d2","#fcbba1","#fdd0a2","#fdae6b"
       ,"#fc9272","#fd8d3c","#fb6a4a","#ef3b2c","#cb181d","#a50f15","#7f2704","
       #67000d")
10 # Funktion um aus einer Tabelle einen bestimmten Bereich zu extrahieren
11 # tabelle -> übergebene Tabelle (Typ: data.frame)
12 # gruppenvektor -> Vektor nach dem gruppiert wird (Typ: vector)
13 # typ -> "c"=character und "n"=numeric
14 func_bereich <- function(tabelle,spalte,gruppenvektor,typ) {
15   df <- NULL
16   splitvektor <- NULL
17   i <- NULL
18   for (i in gruppenvektor) {
19     if (grepl("-",i) == TRUE) {
20       splitvektor <- unlist(strsplit(i,"-"))
21     }
22     else {
23       splitvektor[1] <- i
24       splitvektor[2] <- i
25     }
26     if (typ == "n") {
27       von <- as.numeric(splitvektor[1])
28       bis <- as.numeric(splitvektor[2])
29     }
30     else {
31       von <- as.character(splitvektor[1])
32       bis <- as.character(splitvektor[2])
33     }
34     if (von == bis) {
35       df <- rbind(df,subset(tabelle,tabelle[,spalte] == von))
36     }
37     else {
38       df <- rbind(df,subset(tabelle,tabelle[,spalte] >= von & tabelle[,spalte
           ] <= bis))
39     }
40   }
41   return(df)
42 }
```

```
43 # Aufteilen der Stammdaten nach Landkreisen
44 # Die Spalten "PSEUDONYM", "PLZ", "VERSICHERUNGSART" werden extrahiert
45 stammdaten_landkreise <- stammdaten[c(1,3,5)]
46 # Landkreise in Sachsen-Anhalt definieren
47 halle <- func_bereich(stammdaten_landkreise,2,plz_halle,"n")
48 saalekreis <- func_bereich(stammdaten_landkreise,2,plz_saalekreis,"n")
49 mansfeld_suedharz <- func_bereich(stammdaten_landkreise,2,plz_mansfeld_
                                    suedharz,"n")
50 harz <- func_bereich(stammdaten_landkreise,2,plz_harz,"n")
51 anhalt_bitterfeld <- func_bereich(stammdaten_landkreise,2,plz_anhalt_
                                    bitterfeld,"n")
52 salzlandkreis <- func_bereich(stammdaten_landkreise,2,plz_salzlandkreis,"n")
53 burgenlandkreis <- func_bereich(stammdaten_landkreise,2,plz_burgenlandkreis,
                                   "n")
54 wittenberg <- func_bereich(stammdaten_landkreise,2,plz_wittenberg,"n")
55 dessau_rosslau <- func_bereich(stammdaten_landkreise,2,plz_dessau_rosslau,
                                  "n")
56 stendal <- func_bereich(stammdaten_landkreise,2,plz_stendal,"n")
57 altmarkkreis_salzwedel <- func_bereich(stammdaten_landkreise,2,
                                          plz_altmarkkreis_salzwedel,"n")
58 magdeburg <- func_bereich(stammdaten_landkreise,2,plz_magdeburg,"n")
59 boerde <- func_bereich(stammdaten_landkreise,2,plz_boerde,"n")
60 jerichower_land <- func_bereich(stammdaten_landkreise,2,plz_jerichower_land,
                                   "n")
61 # Alle Landkreise in Sachsen-Anhalt zusammenfassen
62 alle_kreise <- rbind (magdeburg,halle,dessau_rosslau,altmarkkreis_salzwedel,
63                       stendal,boerde,jerichower_land,harz,salzlandkreis,
64                       anhalt_bitterfeld,wittenberg,mansfeld_suedharz,
65                       saalekreis,burgenlandkreis)
66 # Ermittlung der NICHT in Sachsen-Anhalt gemeldeten Versicherten
67 alle_anderen <- stammdaten_landkreise[!stammdaten_landkreise$PSEUDONYM %in%
       alle_kreise$PSEUDONYM,]

69 # Einlesen der shape-Datei in R
70 shape_laden <- function(karte) {
71   # Benötigte Module/Plugins laden
72   library(sp)
73   library(rgeos)
74   library(stringr)
75   library(maptools)
76   # Falls "plz", wird Sachsen-Anhalt unterteilt nach PLZ-Gebieten geladen
77   if (karte=="plz") {
78     shapefile <- readShapeSpatial(fn="/shapefiles/sachsen_anhalt/sa_plz.shp")
79   }
```

```
80    # Falls "lk", wird Sachsen-Anhalt unterteilt nach Landkreisen geladen
81    if (karte=="lk") {
82      shapefile <- readShapeSpatial(fn="/shapefiles/sachsen_anhalt/sa_lk.shp")
83    }
84    return(shapefile)
85  }
86  # unkorrigierte Standardabweichung
87  sd2 <- function(x) {sqrt(sum((x - mean(x))^2) / length(x))}
```

In Zeile 2 wird ein String-Vektor `landkreise` mit allen Landkreisen/kreisfreien Städten definiert, der zur Beschriftung von Tabellen und Balkendiagrammen dient. Zeile 4-8 bestimmen die Farbvektoren, die für die Plotausgabe häufig herangezogen werden. Der nächste Codeabschnitt (Zeile 13-41) beschreibt eine Funktion, mit deren Hilfe man bestimmte Zeilen aus einem *data.frame*, der durch das Argument `tabelle` übergeben wird, selektieren kann. Das Argument `spalte`, vom Typ *numeric*, bestimmt die Spaltennummer in dem *data.frame*, auf die der Filter angewendet wird. Der Parameter `gruppenvektor` besteht aus Strings, die einen Einzelwert oder eine Sequenz, gebildet durch ein `"-"` zwischen zwei Werten, darstellen. Beispielsweise werden mit Hilfe dieser Funktion aus den Stammdaten und den oben definierten Postleitzahlbereichen (siehe Seite 38), die vierzehn Landkreise/kreisfreien Städte abgebildet. Diese sind in den nachfolgenden Zeilen 44-59 zu finden. Der letzte Parameter der Funktion `func_bereich()` heißt `typ` und gibt an, ob es sich bei dem Spaltenfeld um einen numerischen oder alphanumerischen Wert handelt.

`alle_kreise` ist ein *data.frame*, der durch ein `rbind()` alle vierzehn Regionen zusammenfasst, wodurch, falls alle Regionen benötigt werden, der Programmcode vereinfacht wird (siehe Zeile 61). Nachfolgender *data.frame* `alle_anderen` beinhaltet die Differenzmenge zwischen den Stammdaten und der Menge, die durch `alle_kreise` abgebildet werden. Konkret sind dies alle Pseudonyme, die nicht in Sachsen-Anhalt ansässig sind.

Um das *shapefile* in `R` zu nutzen, wird dieses und alle benötigten Zusatzpakete automatisch durch die Funktion `shape_laden()` geladen (Zeile 69-84). Die Funktion besitzt einen Inputparameter `karte`, mit der entweder die landkreis- oder postleitzahlenbasierende Karte von Sachsen-Anhalt ausgewählt wird. Die letzte Zeile definiert die unkorrigierte Standardabweichung als Funktion `sd2()`, die für spätere Analysen benutzt wird.

5. Auswertende und explorative Studien

In diesem Kapitel werden die Untersuchungsergebnisse vorgestellt, die sich aus den in der Einleitung erwähnten Fragestellungen ergeben haben. Schwerpunkt der folgenden Analysen ist die Auswertung des Datenmaterials nach geografischen Gesichtspunkten. Grundlage dafür bildet der Einblick in die statistischen Methoden, die Programmiersprache `R` und den Datenbestand.

5.1. Regionale Verteilung der Versicherten in Sachsen-Anhalt

Die erste Analyse hat zum Ziel, sich einen Überblick über die zahlenmäßige Verteilung der AOK-Versicherten in Sachsen-Anhalt zu verschaffen. Es soll mit Hilfe der absoluten Häufigkeit festgestellt werden, wie sich die Gesamtanzahl der Versicherten verteilt und ob regionale Konzentrationen auftreten. Die Versicherten werden anhand ihrer Postleitzahlenbereiche bzw. ihres Wohnortes gruppiert, wodurch eine genauere Abbildung möglich wird, als durch die Einteilung nach Landkreisen/kreisfreien Städten.
Dazu werden die bereinigten Stammdaten benötigt, die sich in Verbindung mit der *.shp*-Datei als eine farbcodierte Landkarte präsentieren. Diese Verknüpfung wird ermöglicht, da jedem Polygon der *.shp*-Datei das Attribut *PLZ* zugeordnet ist. Es ist anzumerken, dass in Sachsen-Anhalt geografisch getrennte Gebiete eine identische Postleitzahl aufweisen und es dadurch bei der Darstellung der Farben partiell zu Ungenauigkeiten kommen kann. Dies ist dem Umstand geschuldet, dass der Örtlichkeitsbezug technisch über die Postleitzahlen und nicht über die Ortsangabe umgesetzt wird. (Insgesamt 17 verschiedene Postleitzahlen sind zwei bis maximal vier lokal getrennten Bereichen zugeordnet).

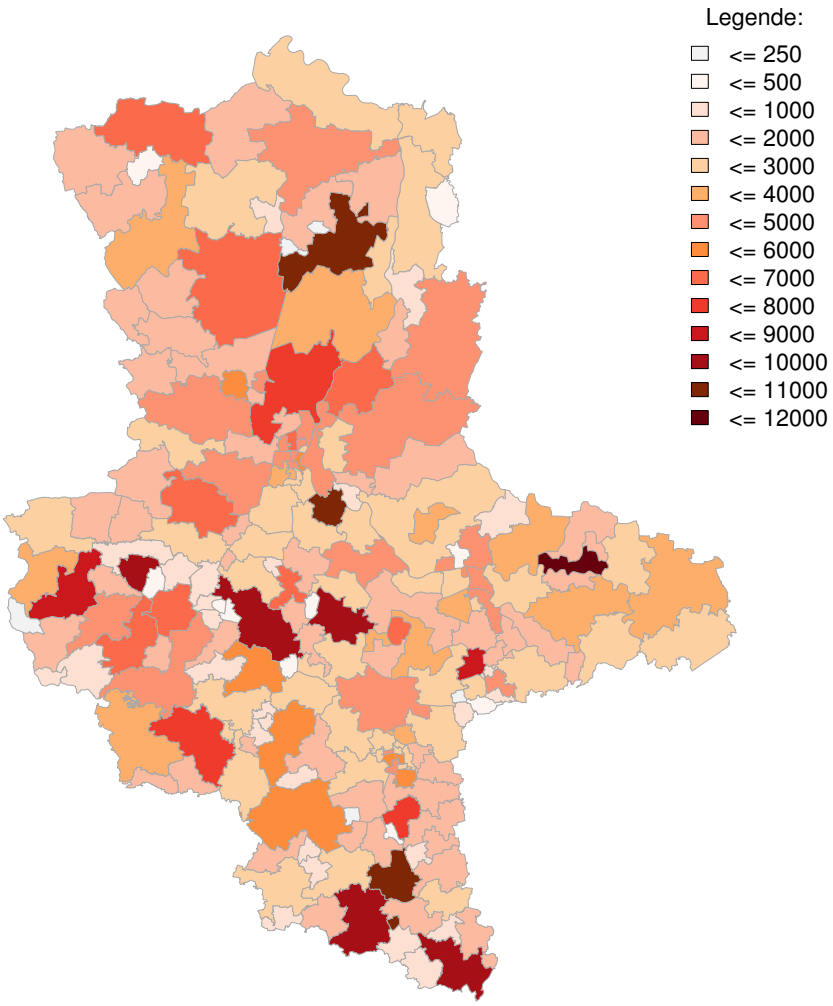

Abb. 5.1.: Absolute Anzahl AOK-Versicherter

Die oben dargestellte Abbildung 5.1 zeigt das Resultat der Analyse. Zu sehen ist eine Heatmap, in der Informationen über eine Farbpalette abgebildet werden. Je kräftiger das betreffende Postleitzahlgebiet eingefärbt ist, desto mehr Versicherte sind an diesem Ort bei der AOK angemeldet. Dabei schwankt die Anzahl der Versicherten pro Postleitzahlgebiet zwischen 93 (Wünsch, Saalekreis) und 11.389 (Wittenberg).
Die Untersuchung filtert nicht in Sachsen-Anhalt ansässige Patienten heraus, sodass von

den insgesamt 605.842 Pseudonymen, 580.529 Versicherte verbleiben, die ihren Wohnsitz in Sachsen-Anhalt haben.
Weiterhin lässt sich aus der Grafik ableiten, dass durch die etwas intensivere Farbgebung im Norden mehr AOK-Versicherte leben als im Osten Sachsen-Anhalts, erkenntlich durch eine deutlich zartere Farbschattierung. Auch sind im südlichen Teil der Karte, im Vergleich zu den restlichen Gebieten, schwächere Farben zu erkennen, mit Ausnahme von partiell sehr kräftigen Regionen. Der Westen und das Zentrum von Sachsen-Anhalt haben eine eher inhomogene Verteilung im Gegensatz zum östlichen Teil. Hier grenzen häufiger stark mit weniger stark belegten Gebieten aneinander.

Die R-Befehle dieser Auswertung:

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")
# Data.frame für die Darstellung (maptools) erzeugen
anzahl_lk <- data.frame(shapes$plz)
anzahl_lk <- cbind(anzahl_lk,0)
# Spaltenüberschriften vergeben
colnames(anzahl_lk) <- c("PLZ","Freq")
# Häufigkeitstabelle über alle vorkommenden PLZ der Landkreise
plz_lk <- data.frame(table(alle_kreise$PLZ))
# Spaltenüberschriften vergeben
colnames(plz_lk) <- c("PLZ", "Freq")
# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
plz_lk$PLZ <- str_pad(string=plz_lk$PLZ,width=5,pad="0")
# Anzahl der vorkommenden PLZ der LK in den data.frame "anzahl_lk" schreiben
for (i in 1:length(anzahl_lk$PLZ)) {tmp <- subset(x=plz_lk$Freq,subset=(plz_
    lk$PLZ == anzahl_lk[i,1]))
  if (length(tmp) == 0) {
    anzahl_lk[i,2] <- 0
  }
  else {
    anzahl_lk[i,2] <- tmp
  }
}
# Einteilung der Schritte für die Heatmap
break_points <- c(0,250,500,1000,2000,3000,4000,5000,6000,7000,8000,9000,
                  10000,11000,12000)
class <- cut(anzahl_lk$Freq,breaks=break_points)
```

```
28 levels(class) <- break_points[2:length(break_points)]
29 ### PLOT BEGINNT ###
30 # Seitenabstand
31 par(mar=c(3,2,2,10))
32 # Plotten
33 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="Anzahl der
       Versicherten in Sachsen-Anhalt")
34 # Gesamtanzahl an Versicherten anzeigen
35 mtext(text=paste("Gesamtanzahl AOK-Versicherte: ",sum(plz_lk$Freq),sep=""),
       side=1,cex=0.7,line=1)
36 # Legende anzeigen
37 legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
       black",cex=1.0,title="Legende:",bty="n")
```

Da der Quellcode für die Erzeugung einer Heatmap in den weiteren Analysen zum Teil redundant vorkommt, wird in diesem Abschnitt der Code ausführlich erläutert und bei Bedarf an diese Stelle verwiesen.
Die ersten beiden Zeilen laden die `initial.R` in `R` ein und die Zeilen 3-4 rufen die selbstdefinierte Funktion `shape_laden()` mit dem Argument `karte=` und dem Wert `"plz"` auf. Dadurch wird das *shapefile*, das nach den einzelnen PLZ-Gebieten gegliedert ist, geladen. Die folgenden Zeilen 6-9 erzeugen ein *data.frame*, in den im weiteren Verlauf des Quellcodes Informationen hinterlegt werden, die für die Darstellung der Karte notwendig sind. Mit dem Befehl `colnames()` werden die Spaltenbeschriftungen des *data.frames* angepasst. Die erste Spalte wird *PLZ*, da hier die insgesamt 202 unterschiedlichen Postleitzahlen von Sachsen-Anhalt erfasst sind, und die zweite *Freq*, für die absolut vorkommende Anzahl an Versicherten jeder Postleitzahl, genannt.
Für die Analyse wird ein zweiter *data.frame* `plz_lk` benötigt, in dem durch die `table()`-Funktion, eine Häufigkeitstabelle aller auftretenden Postleitzahlen aus den Stammdaten, erstellt wird (Zeile 11). `alle_kreise$PLZ` entspricht dabei der Spalte der Postleitzahlen aus dem in der `initial.R` definierten *data.frame* `alle_kreise`, in dem alle vierzehn Landkreise/kreisfreien Städte zusammengefasst sind. Die Funktion `str_pad()` in Zeile 15 nutzt das Zusatzpaket *stringr* und ermöglicht es, dass die Postleitzahlen immer 5-stellig sind. Die `for()`-Schleife füllt den *data.frame* `anzahl_lk`, deren Werte der Spalte *PLZ* der Reihenfolge der Polygone der *.shp*-Datei entsprechen, mit den ermittelten Häufigkeitswerten aus dem *data.frame* `plz_lk`. Die `subset()`-Funktion stellt sicher, dass

die richtigen Häufigkeitswerte den einzelnen Postleitzahlen dem *data.frame* `anzahl_lk` zugeordnet werden.
Dieser Schritt ist notwendig, da die Reihenfolge der Polygone der *.shp*-Datei auch die Folge der Postleitzahlen bestimmt und diese eingehalten werden muss. Aus diesem Grund kann der *data.frame* `plz_lk` nicht direkt eingesetzt werden.
Die Festlegungen `break_points` und `class` in den Zeilen 26 bis 28 bestimmen die Bereiche, in der die Heatmap farblich eingeteilt wird. Der Zahlenvektor `c(0,250,...,12000)` klassifiziert dabei einzelne Abschnitte zwischen der minimalen und maximalen Häufigkeit (absolut vorkommende Versicherte pro PLZ-Gebiet). Mit der `cut()`-Funktion werden die einzelnen absoluten Häufigkeitswerte entsprechend der `break_points` kategorisiert.
Mittels `par()` werden Grafikoptionen für die Darstellung festgelegt und das hier benutzte Argument `mar=` legt die Randabstände für die Plotausgabe fest. Der Funktionsaufruf `plot()` gibt die Landkarte in einem Fenster aus, wobei `border=` eine Farbe übergeben wird, mit der die Polygonumringe gezeichnet werden und `col=` erhält den selbst definierten Farbvektor `farben_shape`, der jeder einzelnen Postleitzahl eine bestimmte Farbe seiner Palette, basierend auf der Kategorisierung der `class` und `break_points`, zuordnet. Der letzte `R`-Befehl `legend()` erzeugt die Kartenlegende.

Die erste Analyse hat die absolute Anzahl aller AOK-Versicherten regional aufgeschlüsselt. Die anknüpfende Betrachtung liefert eine weitere Auswertung in Bezug auf die regionale Verteilung der Versicherten in Sachsen-Anhalt.
Es ist interessant zu erfahren, wie stark die AOK in jedem Landkreis/kreisfreien Stadt prozentual vertreten ist bzw. inwiefern die AOK als Versicherungsträger präsent ist, bezüglich der Gesamtbevölkerung des Landes im Jahr 2012. Um einen Vergleich vorzunehmen, werden vom statistischen Landesamt Sachsen-Anhalt die Bevölkerungszahlen der Gemeinden aus dem Jahr 2012 herangezogen [23, S. 7].
Die folgende Abbildung zeigt das Ergebnis, in der die prozentualen Anteile an der Gesamtbevölkerung sichtbar sind.

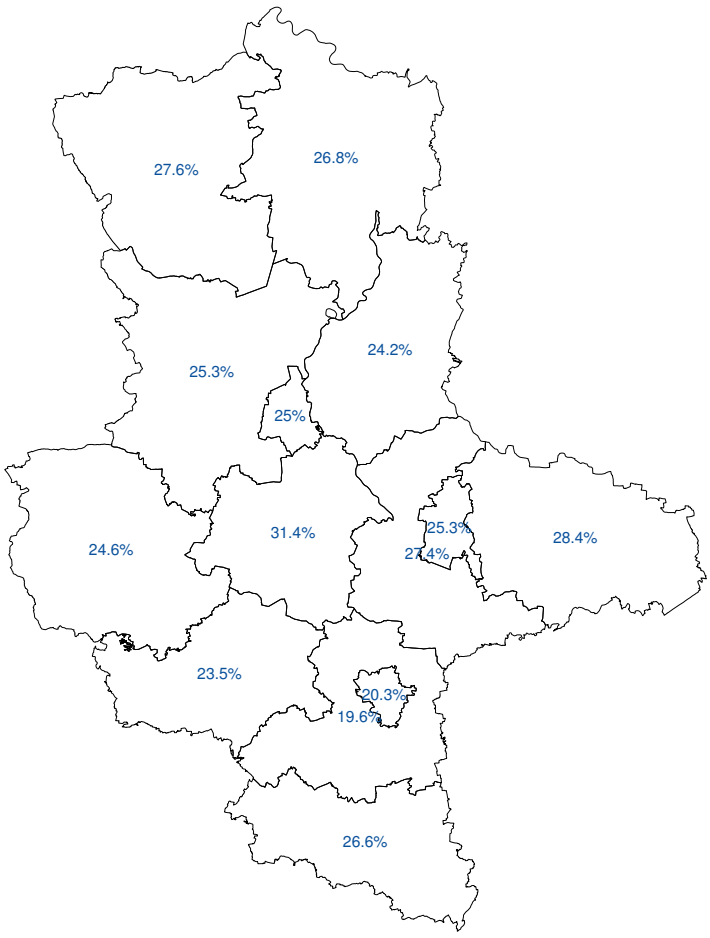

Abb. 5.2.: Prozentualer Anteil AOK-Versicherter an der Gesamtbevölkerung

Wie abzulesen ist, bewegt sich der prozentuale Anteil zwischen 19.6% (Saalekreis) und 31.4% (Salzlandkreis). Im Durchschnitt beträgt der Anteil AOK-Versicherter an der Gesamtbevölkerung ziemlich exakt ein Viertel (25.42%). Aus dieser Gleichverteilung kann geschlussfolgert werden, dass der Datenbestand für die Landkreise/kreisfreien Städte, eine homogene und repräsentative Menge für ganz Sachsen-Anhalt darstellt. Für weitere Analysen ist dies von Vorteil, da bei der statistischen Betrachtung eine homogene Ausgangslage für ein besseres Fundament sorgt und zu aussagekräftigeren Ergebnissen führen kann. Landkreise/kreisfreie Städte sind dadurch ebenfalls besser zu vergleichen.

Der zuständige Quellcode der Heatmap:

```
1  # initial laden und ausführen
2  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
3  # Laden der Shape-Datei: Sachsen-Anhalt (nach Landkreisen)
4  shape_lk <- shape_laden(karte="lk")
5  # Data.frame für die Darstellung (maptools) erzeugen
6  bevoelkerung <- data.frame(shape_lk$name)
7  # Spalte hinzufügen: Gesamtanzahl der Bevölkerung pro Landkreis
8  bevoelkerung$Anzahl_LK <- c(173138,190545,176699,85329,233107,228030,94776,
                               232203,147036,194180,205672,88055,119470,134622)
9  # Spalte hinzufügen: Anzahl der AOK-Versicherten pro Landkreis
10 bevoelkerung$Anzahl_AOK <- c(length(anhalt_bitterfeld$PSEUDONYM),
                                length(burgenlandkreis$PSEUDONYM),
                                length(boerde$PSEUDONYM),
                                length(dessau_rosslau$PSEUDONYM),
                                length(halle$PSEUDONYM),
                                length(harz$PSEUDONYM),
                                length(jerichower_land$PSEUDONYM),
                                length(magdeburg$PSEUDONYM),
                                length(mansfeld_suedharz$PSEUDONYM),
                                length(saalekreis$PSEUDONYM),
                                length(salzlandkreis$PSEUDONYM),
                                length(altmarkkreis_salzwedel$PSEUDONYM),
                                length(stendal$PSEUDONYM),
                                length(wittenberg$PSEUDONYM)
                                )
11 # Prozentualen Anteil berechnen
12 bevoelkerung$Proz <- (bevoelkerung[,3] / bevoelkerung[,2])*100
13 ### PLOT BEGINNT ###
14 # Anzeigen der Landkarte bzw. Plotten
15 plot(x=shape_lk,border="black")
16 # Berechneten proz. Anteil auf der Landkarte ausgeben
17 text(x=getSpPPolygonsLabptSlots(shape_lk),labels=paste(round(x=bevoelkerung$
       Proz,digits=1),"%",sep=""),cex=0.55,col=farben[1])
```

Zeile 2 und 4 laden, wie bereits erläutert, die `initial.R` und das *shapefile* in `R` ein. Mit dem Befehl in der sechsten Zeile wird ein *data.frame* aus den Namen der Attribute des *shapefiles* erstellt (in diesem Fall den Namen der Landkreise/kreisfreien Städte). Dem *data.frame* `bevoelkerung` wird eine Spalte hinzugefügt, die die Anzahl der Einwohner der einzelnen Landkreise enthält. Die Zahlenwerte im Vektor sind dem Bericht des statistischen Landesamtes Sachsen-Anhalt entnommen. Weiterhin wird die Gesamtanzahl der

AOK-Versicherten pro Landkreis als ein Vektor angehängt (Zeile 8-10). Die Elemente des Vektors sind durch die Anwendung der `length()`-Funktion auf die in der `initial.R` definierten Landkreise erstellt worden. In der nächsten Zeile wird der prozentuale Anteil der AOK-Versicherten an der Gesamtbevölkerung berechnet, indem die Anzahl der AOK-Versicherten durch die Gesamtanzahl der Bevölkerung pro Landkreis dividiert und mit 100 multipliziert wird. In Zeile 15 geschieht das "zeichnen" (`plot()`) der Karte und mit der letzten Zeile wird erreicht, dass die berechneten Prozentwerte auf der Landkarte dargestellt werden. Dies wird mit dem `text()`-Befehl realisiert, der mit dem Argument `x=` und dessen Wert `getSpPPolygonsLabptSlots(shape_lk)` die Koordinatenpositionen der einzelnen Landkreise erhält und mittels `labels=` die prozentualen Werte aus dem *data.frame* `bevoelkerung` darstellen kann. Mit der `round()`-Funktion werden die prozentual berechneten Werte auf eine Nachkommastelle gerundet und anschließend, mit Hilfe der `paste()`-Funktion, mit einem %-Symbol versehen. `cex=` legt die Schriftgröße des darzustellenden Textes fest.

5.2. Regionale Verteilung der Versicherungsarten in Sachsen-Anhalt

Aus den Stammdaten und der regionalen Einteilung können weitere Informationen herausgezogen werden. Diese Untersuchung gliedert Sachsen-Anhalt in seine elf Landkreise und drei kreisfreien Städte, zuzüglich einer Gruppe der restlichen Versicherten, die ihren Wohnsitz außerhalb des Erhebungsgebietes haben. Dabei werden die Versicherten entsprechend ihrer Versicherungsarten gruppiert bzw. getrennt betrachtet. Das Ziel dieser Analyse ist die Beantwortung der Frage, ob bestimmte Bevölkerungsgruppen (repräsentiert durch den Versicherungsstatus) gebietsweise stärker vertreten sind als andere. Eine Stadt-Land-Betrachtung soll ebenfalls zeigen, ob bestimmte Versicherungsgruppen wie zum Beispiel Arbeitslose häufiger in ländlichen als in städtischen Gebieten anzutreffen sind. Zum Ende der Auswertung wird der Versicherungsstatus noch differenziert, indem eine geschlechtsspezifische Unterteilung erfolgt.

Zunächst wird die Betrachtung anhand von absoluten Zahlen vorgenommen. Für die Aufgabenstellung werden von den insgesamt neun verschiedenen Versicherungsarten fünf in der Gruppe *Sonstige* zusammengefasst. Bei diesen handelt es sich um *freiwillig versichert, nicht_GKV_versichert, Rehabilitanten, Rentenantragsteller* und *unbekannt*. Wie die unteren Abbildungen erkennen lassen, macht die Summe aller fünf Arten weniger als 3% aus, sodass eine Einzeldarstellung aller neun Arten in der Grafik keinen höheren Informationsgehalt hätte.

Die Abbildung 5.3 zeigt ein mehrstufig gestapeltes Balkendiagramm, in dem die ersten vierzehn Balken Regionen innerhalb Sachsen-Anhalts und der fünfzehnte Regionen der außerhalb lebenden Versicherten verkörpern. Die Beschriftung der X-Achse ist beispielhaft für alle folgenden Diagramme dieser Art: Auf die drei kreisfreien Städte, sortiert nach deren Größe, folgen die Landkreise, sortiert nach dem Nord-Süd-Gefälle.
Die Höhen der Balken spiegeln die absolute Anzahl an Versicherten jedes Landkreises/kreisfreien Stadt wieder, die ebenfalls am oberen Balkenrand abzulesen ist. Für eine ungefähre Größenangabe der Versicherungsarten wird die Y-Achse in einer Schrittweite von 5000 skaliert und beschriftet. Die verschiedenen Gruppen sind, wie in der Legende zu erkennen, durch differenzierte Blautöne zu unterscheiden.
Weiterhin ist abzulesen, dass bezogen auf die einzelnen Landkreise/kreisfreien Städte, die größte Anzahl an AOK-Versicherten aus dem Salzlandkreis stammen, gefolgt von der Stadt Magdeburg und dem Harz. Aufgrund der vorhergehenden Feststellung im Kapitel 5.1 (AOK fungiert zu ca. 25% als Versicherungsträger im ganzen Bundesland), zeigt die Grafik, dass der Norden von Sachsen-Anhalt im Vergleich zum südlichen Teil, dünner besiedelt ist. Bezüglich des Versicherungsstatus ist zu erkennen, dass in allen vierzehn Regionen eine große Mehrheit der Versicherten *Rentner* sind. Eine Ausnahme bildet die Gruppe *außerhalb_Sachsen_Anhalt*, in der im Vergleich zu den anderen, alle Versicherungsarten relativ gleichmäßig verteilt sind.

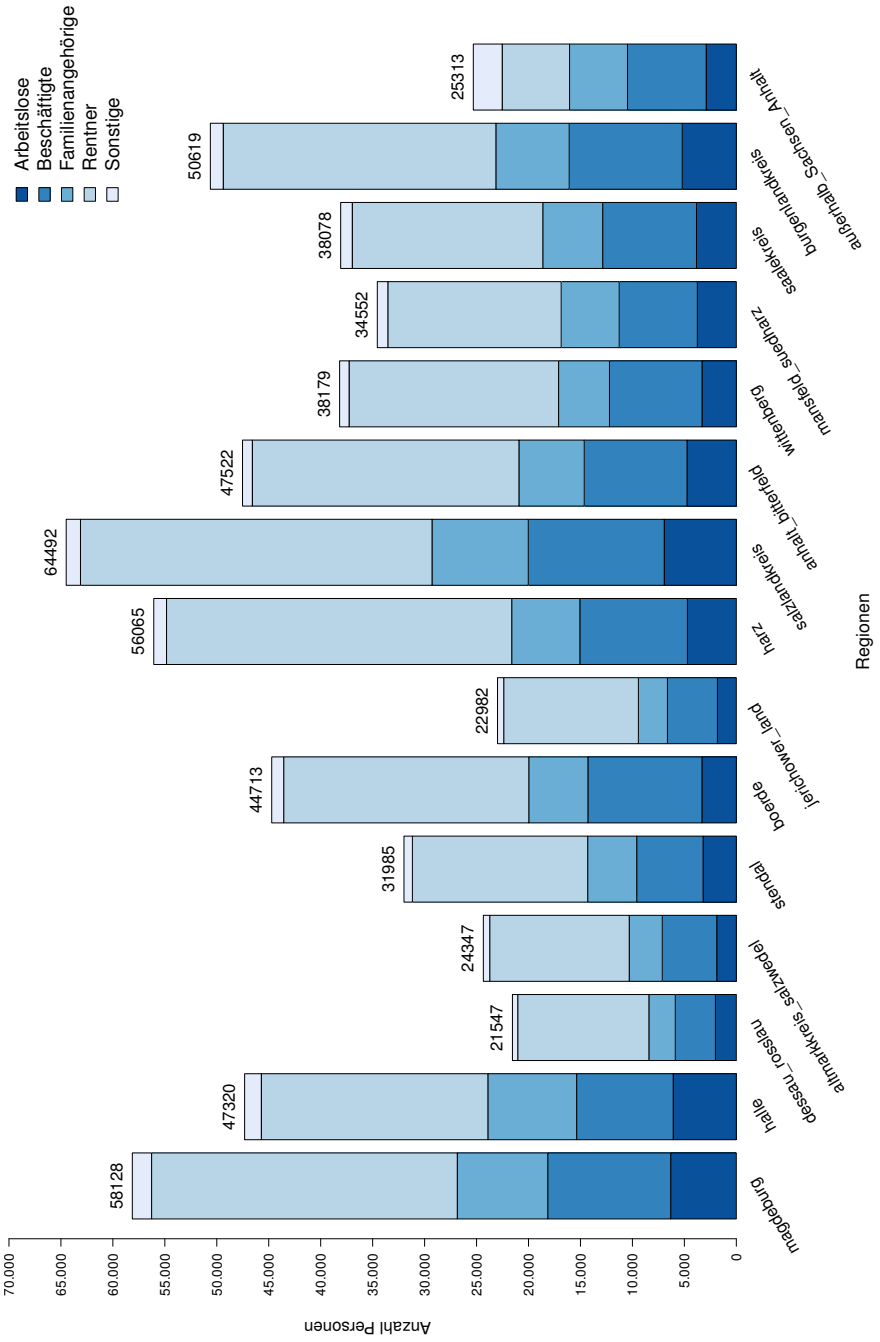

Abb. 5.3.: Absolute Häufigkeit der Versicherungsarten

Um eine bessere Vergleichbarkeit zwischen den Regionen herstellen zu können, wird nachfolgend auf Basis der absoluten, eine relative Betrachtung durchgeführt. Dazu wird jeweils die Gesamtanzahl auf 100% normiert, sodass die Y-Achsenbeschriftung eine prozentuale Skala von 0-100% besitzt. Wie bereits oben erwähnt, wird zusätzlich eine Unterteilung nach Frauen und Männer vorgenommen, wie in Abbildung 5.5 zu sehen ist.

Die Prozentangabe innerhalb der Legende der Abbildung 5.4, bezieht sich je Gruppe auf alle AOK-Versicherten des Datenbestandes und bildet den Gesamtdurchschnitt. Es ist zu erkennen, dass im Durchschnitt mehr als 51% der AOK-Versicherten Rentner sind, wobei die kreisfreie Stadt Dessau-Roßlau und der Landkreis Harz mit knapp 60% hervorstechen. Die restlichen Versicherungsarten bewegen sich in allen Landkreisen/kreisfreien Städten in der gleichen Größenordnung.

Bei der geschlechterabhängigen Abbildung (5.5) handelt es sich um ein gestapeltes und gruppiertes Balkendiagramm, indem sich die Prozentangaben jeweils auf das Verhältnis des entsprechenden Geschlechts zu ihrer geschlechtsspezifischen Gesamtanzahl innerhalb des Landkreises/kreisfreien Stadt beziehen.
Das Resultat zeigt ähnliche Ergebnisse wie in der vorherigen Abbildung, mit dem Unterschied, dass bei den Männern der Beschäftigungsanteil höher und der Rentneranteil stets niedriger ist als bei den Frauen. Eine mögliche Ursache dafür ist auch Gegenstand der Untersuchung in Kapitel 5.4, in welchem die Altersstruktur in Sachsen-Anhalt betrachtet wird.
Bezüglich des Arbeitslosenanteils ist ein leichter Überhang bei den Männern zu verzeichnen.

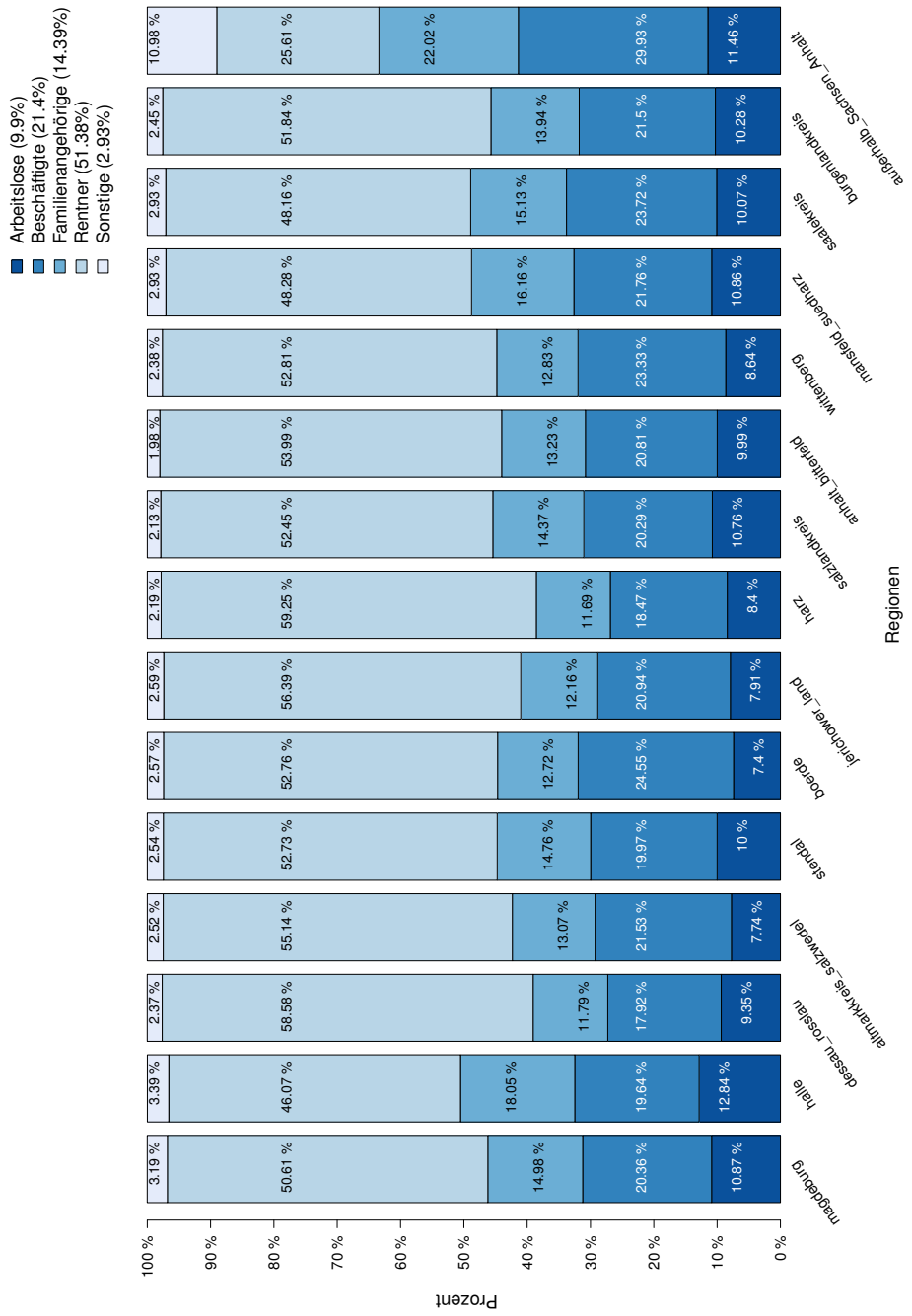

Abb. 5.4.: Prozentuale Häufigkeit der Versicherungsarten

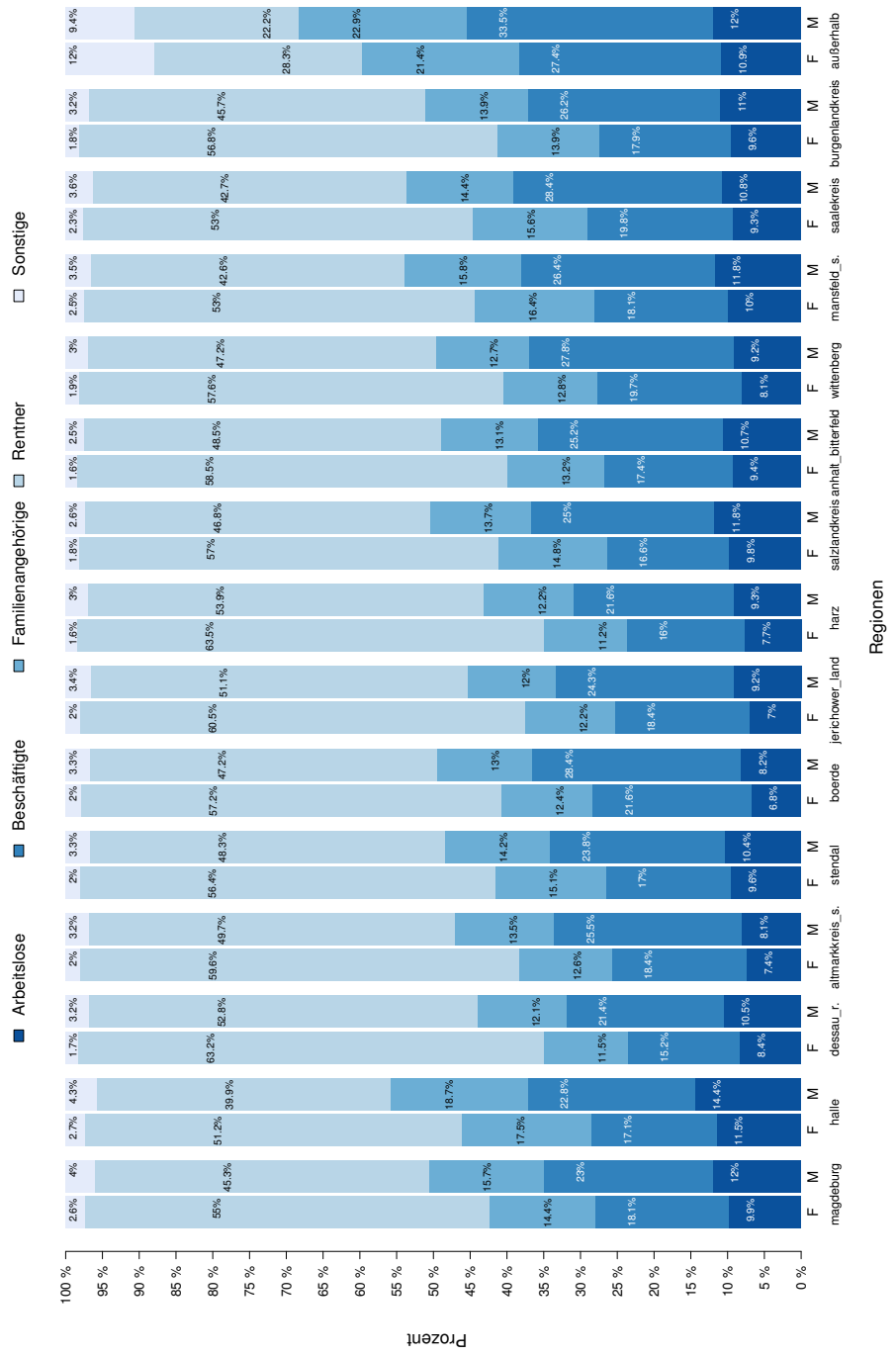

Abb. 5.5.: Prozentuale Häufigkeit der Versicherungsarten (Mann und Frau)

Der Quellcode gliedert sich für die drei Grafiken dieses Kapitels in mehrere Abschnitte, da bestimmte Programmteile von Abbildung 5.3 Voraussetzung für die Abbildungen 5.4 und 5.5 sind.
Der erste Abschnitt enthält eine Funktion, die für alle drei Grafiken erforderlich ist:

```
1  # Funktion um aus 9 nur 5 Versicherungsarten zu erzeugen
2  func_v_arten <- function(frequenz,vers_arten) {
3    # Variablen definieren
4    grpvektor <- NULL; arten <- NULL
5    # 5 Versicherungsarten
6    art1 <- "arbeitslos"
7    art2 <- "beschäftigt"
8    art3 <- "Familienangehörige"
9    art4 <- "Rentner"
10   art10 <- "Sonstige"
11   arten <- c(art1,art2,art3,art4,art10)
12   # Arten 1-4 werden übernommen
13   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art1)))
14   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art2)))
15   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art3)))
16   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art4)))
17   # Arten 5-9 werden zu art10=Sonstiges zusammengefasst
18   grpvektor = rbind(grpvektor,sum(subset(x=frequenz,subset=(vers_arten !=
         art1 & vers_arten != art2 & vers_arten != art3 & vers_arten != art4))))
19   # Rückgabe
20   return(data.frame(VERSICHERUNGSART=arten,Freq=grpvektor))
21 }
```

Mit der `func_v_arten()`-Funktion werden von den insgesamt neun verfügbaren Versicherungsarten fünf zusammengefasst. Der Funktion werden zwei Parameter mit der Bezeichnung `frequenz`, ein Vektor vom Typ *numeric*, und `vers_arten`, ein Vektor vom Typ *character*, übergeben. Dazu werden ab Zeile 4 bis 11 verschiedene Variablen definiert. Falls die Versicherungsart eine der Hauptarten ist, werden diese direkt in den `grpvektor` durch ein `rbind()` geschrieben (Zeile 13 bis 16). In Zeile 18 wird der `grpvektor` erweitert, durch die Zusammenfassung (`sum()`-Funktion) der restlichen Versicherungsarten zu der Gruppe *Sonstige*. In der letzten Zeile wird der neugebildete *data.frame* an den Funktionsaufruf zurückgegeben.

Der zweite Abschnitt enthält einen Programmcode, der für die Abbildungen 5.3 und 5.4 benötigt wird. Es ist anzumerken, dass der Code an bestimmten Stellen, wo ein Programmablauf auf alle Landkreise/kreisfreien Städte angewandt wird, verkürzt dargestellt ist. Im Quelltext wird dies durch die Zeichen `{ ... }` angedeutet. Der vollständige Quellcode ist im Quelltext-Anhang vorzufinden (Seite 177 bzw. 180).

```
1  # Häufigkeitstabelle für jede Region erstellen
2  sum_magdeburg <- data.frame(table("VERSICHERUNGSART"=magdeburg$
                                     VERSICHERUNGSART))
3  { ... }
4  # Funktionsaufruf um Versicherungsarten zusammenzufassen
5  gruppe_sum_magdeburg <- func_v_arten(sum_magdeburg$Freq,sum_magdeburg$
                                     VERSICHERUNGSART)
6  { ... }
7  # Ergebnismatrix der abs. Häufigkeiten aller Regionen
8  v_art_plot <- cbind(
9    gruppe_sum_magdeburg$Freq,
10   gruppe_sum_halle$Freq,
11   gruppe_sum_dessau_rosslau$Freq,
12   gruppe_sum_altmarkkreis_salzwedel$Freq,
13   gruppe_sum_stendal$Freq,
14   gruppe_sum_boerde$Freq,
15   gruppe_sum_jerichower_land$Freq,
16   gruppe_sum_harz$Freq,
17   gruppe_sum_salzlandkreis$Freq,
18   gruppe_sum_anhalt_bitterfeld$Freq,
19   gruppe_sum_wittenberg$Freq,
20   gruppe_sum_mansfeld_suedharz$Freq,
21   gruppe_sum_saalekreis$Freq,
22   gruppe_sum_burgenlandkreis$Freq,
23   gruppe_sum_alle_anderen$Freq
24 )
25 # Überschriften setzen
26 rownames(v_art_plot) <- c("Arbeitslose","Beschäftigte","Familienangehörige","
       Rentner","Sonstige")
27 colnames(v_art_plot) <- c(landkreise,"außerhalb_Sachsen_Anhalt")
```

Der erste R-Befehl in Zeile 2, der hier exemplarisch für Magdeburg dargestellt ist, erstellt eine absolute Häufigkeitstabelle (`table()`), aggregiert nach dem Versicherungsstatus. Für die Kombination von *character* (Versicherungsarten) und *numeric* (Häufigkeiten) Werten eignet sich der *data.frame* als Datentyp. In der folgenden Zeile werden für jeden Landkreis/kreisfreie Stadt mit Hilfe der Funktion `func_v_arten()` die Versicherungsar-

ten zusammengefasst. In Zeile 8 bis 27 wird vorbereitend für die `barplot()`-Funktion eine Ergebnismatrix mit den darzustellenden Werten erzeugt und unter Anwendung von `rownames()` bzw. `colnames()` die Zeilen bzw. Spalten betitelt.

Der dritte Abschnitt enthält R-Befehle, die nur für die jeweilige Abbildung erforderlich sind.
Für die Abbildung 5.3 ist folgender Programmcode zuständig:

```
1  ### PLOT BEGINNT ###
2  # SeitenabstandErgebnismatrix
3  par(mar=c(9,6,4,2))
4  # Y-Achse Markierung
5  ybereich <- pretty(x=c(0,signif(1.1*max(colSums(v_art_plot)),digits=1)),n=15)
6  # Plotten
7  balken <- barplot(height=v_art_plot,main="Verteilung der Versicherungsarten
       nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,axisnames=FALSE,col=
       farben,ylim=c(0,1.1*max(colSums(v_art_plot))))
8  # Legende anzeigen
9  legend('topright',legend=rownames(v_art_plot),fill=farben,cex=0.8)
10 # X-Achse anzeigen
11 text(x=balken+0.1,y=-2500,srt=35,adj=1,xpd=TRUE,labels=colnames(v_art_plot),
       cex=0.9)
12 # X-Achsen-Beschriftung
13 mtext(text="Regionen",side=1,line=6.5)
14 # Y-Achse anzeigen
15 axis(side=2,at=ybereich,labels=format(ybereich,big.mark=".",scientific=FALSE)
       ,las=2,cex.axis=0.9,pos=-0.1)
16 # Y-Achsen-Beschriftung
17 mtext(text="Anzahl Personen",side=2,line=3.5)
18 # Absolute Anzahl pro Region über Balken anzeigen
19 text(x=balken,y=colSums(v_art_plot)+1500,labels=colSums(v_art_plot),cex=0.8)
20 ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit der
       Versicherungsarten pro Landkreis)
21 write.table(x=cbind(Landkreise=rownames(v_art_plot),v_art_plot),file="CSV/
       versicherungsarten_pro_lk_absolut.csv",sep=";",row.names=FALSE,col.names=
       TRUE,fileEncoding="latin1")
```

Mit dem ersten Befehl, siehe Zeile 3, werden die Seitenabstände mittels `par(mar=)` des barplots durch einen Vektor mit vier Elementen, fixiert. Die vier Zahlen müssen in der Reihenfolge `c(`Unten, Links, Oben, Rechts`)` angeordnet sein. Mit der Variablen `ybereich` wird die Skalierung der Y-Achse durch die `pretty()`-Funktion, die eine gleichmäßige

Aufteilung einer Sequenz vornimmt, erzeugt.
Eine Zeile weiter erfolgt der `barplot()`-Aufruf des Balkendiagramms, dessen Rückgabe die X-Positionen der einzelnen Balken enthält und der Variablen `balken` zugeordnet werden, da sie an späterer Stelle für die Positionierung der Texte durch die `text()`-Funktion benötigt werden. Das Argument `height=` ist der einzige Pflichtparameter und legt fest, welche Wertewerte grafisch visualisiert werden sollen. Mit `main=` wird ein Titel der Grafik festgelegt, `axes=` und `axisnames=` sind Boolean und steuern die automatische Beschriftung der Achsen. Das Argument `col=` legt den Farbvektor fest und `ylim=` limitiert die Y-Achse.
Mit dem `text()`-Befehl in Zeile 11 können verschiedenste Texte innerhalb der Grafik positioniert werden. An dieser Stelle wird er für die Beschriftung der X-Achse mit den einzelnen Landkreisnamen (`legend=colnames(v_art_plot)`) benutzt. Dabei legen `x=` und `y=` die Positionen im Zeichnungsfenster fest, `srt=` den Beschriftungswinkel und `adj=` sowie `xpd=` sorgen dafür, dass der Text unterhalb der Balken steht. `mtext()` in Zeile 13 fügt der Abbildung ebenfalls Texte hinzu, jedoch werden diese mit dem Parameter `side=` nur an einen der vier Seitenränder geschrieben. Dabei kann `side=` die Werte eins bis vier annehmen und entspricht damit der gleichen Reihenfolge wie bei dem Parameter `par(mar=)` (siehe Seite 57). Um Achsen einem Plot hinzuzufügen, wird die `axis()`-Funktion benötigt. Der Parameter `side=` ist identisch mit dem von `mtext()` und `at=` bzw. `labels=` bestimmen die Position bzw. den Text, der an die Y-Achse geschrieben wird. `las=` mit dem Wert 2 richtet den Text des Arguments `labels=` horizontal aus. Die Schriftgröße wird durch `cex.axis=` festgelegt und durch den letzten Parameter `pos=` kann die Achse horizontal nach links bzw. rechts verschoben werden.
Die in Zeile 21 stehende Funktion `write.table()` speichert die Ergebnismatrix `v_art_plot` als CSV-Datei auf die Festplatte. Die CSV-Datei enthält eine zahlengenaue tabellarische Auflistung aller absoluten Häufigkeiten der Versicherungsarten pro Region.

Der Quellcode für die Abbildung 5.4:

```
# Prozentuale Ergebnismatrix erstellen
v_art_plot_proz <- round(x=prop.table(x=v_art_plot,margin=2)*100,digits=2)
### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(9,5,5,10.3))
# Legende zusammensetzen
leg_name <- rownames(v_art_plot)
leg_proz <- c(
      round(sum(v_art_plot[1,]) * 100 / sum(v_art_plot),digits=2),
      round(sum(v_art_plot[2,]) * 100 / sum(v_art_plot),digits=2),
      round(sum(v_art_plot[3,]) * 100 / sum(v_art_plot),digits=2),
      round(sum(v_art_plot[4,]) * 100 / sum(v_art_plot),digits=2),
      round(sum(v_art_plot[5,]) * 100 / sum(v_art_plot),digits=2) )
legende <- paste(leg_name," (",leg_proz,"%)",sep="")
# Y-Achse Markierung
ybereich <- pretty(x=c(0,100),n=10)
# Plotten
balken2 <- barplot(height=v_art_plot_proz,main="Verteilung der
    Versicherungsarten nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,
    axisnames=FALSE,col=farben,space=0.2)
# Legende anzeigen
legend(x=18.5,y=100,legend=legende,fill=farben,cex=0.8,xpd=TRUE)
# X-Achse anzeigen
text(x=balken2+0.1,y=-3,srt=35,adj=1,xpd=TRUE,labels=colnames(v_art_plot_proz
    ),cex=0.9)
# X-Achsen-Beschriftung
mtext(text="Regionen",side=1,line=6.5)
# Y-Achse anzeigen
axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos
    =-0.1)
# Y-Achsen-Beschriftung
mtext(text="Prozent",side=2,line=2)
# Prozente anzeigen (Arbeitslose,Beschäftigte,Familienang.,Rentner,Sonstige)
text(x=balken2+0.33,y=v_art_plot_proz[1,]-4,adj= 1,xpd=TRUE,labels=paste(v_
    art_plot_proz[1,],"%"),cex=0.75,col="white")
text(x=balken2+0.33,y=22,adj=1,xpd=TRUE,labels=paste(v_art_plot_proz[2,],"%")
    ,cex=0.75,col="white")
text(x=balken2+0.33,y=colSums(v_art_plot_proz[1:3,])-8,adj=1,xpd=TRUE,labels=
    paste(v_art_plot_proz[3,],"%"),cex=0.75,col="black")
text(x=balken2+0.33,y=78,adj=1,xpd=TRUE,labels=paste(v_art_plot_proz[4,],"%")
    ,cex=0.75,col="black")
text(x=balken2+0.33,y=colSums(v_art_plot_proz[1:5,])-1,adj=1,xpd=TRUE,labels=
    paste(v_art_plot_proz[5,],"%"),cex=0.75,col="black")
```

Wie in Zeile 2 zu erkennen, wird auf Basis der absoluten Häufigkeitstabelle `v_art_plot` eine prozentuale Tabelle berechnet. `R` bietet dafür die integrierte Funktion `prop.table()`. Der Parameter `margin=` übernimmt die Steuerfunktion, ob Zeilen oder Spalten für die prozentuale Berechnung herangezogen werden; zwei steht für eine spaltenweise Auswertung. Das Ergebnis wird mit `round()` und `digits=2` auf zwei Nachkommastellen gerundet. Die Zeilen 7 bis 14 erweitern die Legende um die Prozentangaben der Versicherungsarten, bezogen auf das gesamte Bundesland. Der weitere Quellcode ist überwiegend identisch mit dem aus Abbildung 5.4.

Für die Abbildung 5.5 ist nachfolgender `R`-Code zuständig. Eine vollständige Auflistung der Befehle ist im Quelltext-Anhang auf Seite 184 nachzuschlagen.

```
1  # Ambudaten einlesen und bereinigen
2  ambudaten <- ambudaten_laden()
3  # Die Spalten "PSEUDONYM", "GESCHLECHT" werden extrahiert
4  ambu_gender <- ambudaten[c(1,7)]
5  # Verbinden der Stammdaten mit den Ambudaten für jede Region
6  magdeburg_gender <- unique(merge(magdeburg,ambu_gender,by="PSEUDONYM"))
7  { ... }

9  # Aggregieren der Versicherungsarten nach Mann und Frau für jede Region
10 magdeburg_gender_freq <- aggregate(x=magdeburg_gender$GENDER,by=list(
       VERSICHERUNGSART=magdeburg_gender$VERSICHERUNGSART),FUN=table)
11 { ... }

13 # Funktionsaufruf um Versicherungsarten zusammenzufassen (Frauen)
14 gruppe_sum_magdeburg_f <- func_v_arten(magdeburg_gender_freq$x[,1],
                               magdeburg_gender_freq$VERSICHERUNGSART)
15 { ... }

17 # Funktionsaufruf um Versicherungsarten zusammenzufassen (Männer)
18 gruppe_sum_magdeburg_m <- func_v_arten(magdeburg_gender_freq$x[,2],
                               magdeburg_gender_freq$VERSICHERUNGSART)
19 { ... }

21 # Ergebnismatrix der abs. Häufigkeiten aller Regionen für Mann und Frau
22 v_art_plot <- cbind(
23   gruppe_sum_magdeburg_f$Freq,
24   gruppe_sum_magdeburg_m$Freq,
25   gruppe_sum_halle_f$Freq,
26   gruppe_sum_halle_m$Freq,
```

```
27    gruppe_sum_dessau_rosslau_f$Freq,
28    gruppe_sum_dessau_rosslau_m$Freq,
29    gruppe_sum_altmarkkreis_salzwedel_f$Freq,
30    gruppe_sum_altmarkkreis_salzwedel_m$Freq,
31    gruppe_sum_stendal_f$Freq,
32    gruppe_sum_stendal_m$Freq,
33    gruppe_sum_boerde_f$Freq,
34    gruppe_sum_boerde_m$Freq,
35    gruppe_sum_jerichower_land_f$Freq,
36    gruppe_sum_jerichower_land_m$Freq,
37    gruppe_sum_harz_f$Freq,
38    gruppe_sum_harz_m$Freq,
39    gruppe_sum_salzlandkreis_f$Freq,
40    gruppe_sum_salzlandkreis_m$Freq,
41    gruppe_sum_anhalt_bitterfeld_f$Freq,
42    gruppe_sum_anhalt_bitterfeld_m$Freq,
43    gruppe_sum_wittenberg_f$Freq,
44    gruppe_sum_wittenberg_m$Freq,
45    gruppe_sum_mansfeld_suedharz_f$Freq,
46    gruppe_sum_mansfeld_suedharz_m$Freq,
47    gruppe_sum_saalekreis_f$Freq,
48    gruppe_sum_saalekreis_m$Freq,
49    gruppe_sum_burgenlandkreis_f$Freq,
50    gruppe_sum_burgenlandkreis_m$Freq,
51    gruppe_sum_alle_anderen_f$Freq,
52    gruppe_sum_alle_anderen_m$Freq)
53  # Überschriften setzen
54  rownames(v_art_plot) <- c("Arbeitslose","Beschäftigte","Familienangehörige","
        Rentner","Sonstige")
55  colnames(v_art_plot) <- c(paste(rep(x=landkreise,times=1,each=2),rep(x=c("F",
        "M"),times=14),sep="_"),"außerhalb_sachsen_anhalt_F","außerhalb_sachsen_
        anhalt_M")

57  # Prozentuale Ergebnismatrix erstellen
58  v_art_plot_proz <- round(x=prop.table(x=v_art_plot,margin=2)*100,digits=2)

60  ### PLOT BEGINNT ###
61  # Seitenabstand
62  par(mar=c(5,4,5,6.5))
63  # Y-Achse Markierung
64  ybereich <- pretty(x=c(0,100),n=20)
65  # Abstände der Barplots definieren
66  a1 <- c(0.1)
67  a2 <- c(0.8)
```

```
68 abstand <- c(a1,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,
       a1,a2,a1,a2,a1,a2,a1,a2,a1)
69 # Plotten
70 balken3 <- barplot(height=v_art_plot_proz,main="Verteilung der Versicherungs-
       arten nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,axisnames=FALSE,
       col=farben,border=NA,space=abstand)
71 # Legende anzeigen
72 legend(x=44.5,y=100,legend=rownames(v_art_plot),fill=farben,cex=0.6,xpd=TRUE)
73 # Geschlecht anzeigen
74 text(x=balken3,y=-2.2,labels=rep(x=c("F","M"),n=15),xpd=TRUE,cex=0.7)
75 # Regionen anzeigen
76 mtext(text=c(landkreise,"außerhalb"),side=1,at=balken3[seq(from=1,to=length(
       balken3),by=2)]+0.7,line=0.8,cex=0.7)
77 # X-Achsen-Beschriftung
78 mtext(text="Regionen",side=1,line=3.3,cex=0.8)
79 # Y-Achse anzeigen
80 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.7,pos
       =-1.2)
81 # Y-Achsen-Beschriftung
82 mtext(text="Prozent",side=2,line=2.5,cex=0.8)
83 # Prozente anzeigen (Arbeitslose,Beschäftigte,Familienang.,Rentner,Sonstige)
84 text(x=balken3+0.35,y=5,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz
       [1,],digits=1),"%",sep=""),cex=0.6,col="white")
85 text(x=balken3+0.35,y=20,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz
       [2,],digits=1),"%",sep=""),cex=0.6,col="white")
86 text(x=balken3+0.35,y=colSums(v_art_plot_proz[1:3,])-8,adj=1,xpd=TRUE,
       labels=paste(round(v_art_plot_proz[3,],digits=1),"%",sep=""),
       cex=0.6,col="black")
87 text(x=balken3+0.35,y=75,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz
       [4,],digits=1),"%",sep=""),cex=0.6,col="black")
88 text(x=balken3+0.35,y=99,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz
       [5,],digits=1),"%",sep=""),cex=0.6,col="black")
```

Für die geschlechterabhängige Betrachtung müssen die Stammdaten mit den ambulanten Daten kombiniert werden, um den einzelnen Pseudonymen, gruppiert nach Landkreisen, ihr Geschlecht zuzuweisen (Zeile 2-7). Anschließend wird für jeden Landkreis/kreisfreie Stadt mit Hilfe der `aggregate()`-Funktion eine Häufigkeitstabelle, aufgeschlüsselt nach dem Geschlecht, erstellt (siehe Zeile 10-11). Das Argument `x=` der Aggregatsfunktion gibt an, auf welches Objekt eine Funktion, wählbar durch den Parameter `FUN=`, angewendet werden soll.

Das Objekt dieser Analyse ist die Spalte *GENDER* der `ambudaten`, auf die die Funktion `table()` angewandt wird. Gruppiert wird das Ergebnis mit Hilfe des `by=` Parameters, der hier auf die Versicherungsarten zeigt.
Nachfolgende Zeilen rufen die bereits erläuterte Funktion `func_v_arten()` auf, die den Versicherungsstatus, getrennt nach dem Geschlecht, gruppiert und eine Ergebnismatrix erzeugt (bis Zeile 53). Eine Abweichung zu den vorherigen Quellcodes zeigt sich in Zeile 67-69. Mit `a1=` und `a2=` werden manuell die Abstände zwischen den Balken gesetzt, um das gewünschte Ergebnis einer zweisäuligen Gruppierung für jede Region zu erhalten.

5.3. Regionale Geschlechterverteilung in Sachsen-Anhalt

Die vorherige Analyse betrachtete die Versicherten bereits geschlechterspezifisch, jedoch lieferte sie keine Informationen über das Mengenverhältnis Männer zu Frauen eines Landkreises, aufgrund der separaten Behandlung. Diese Untersuchung soll die Geschlechter in Beziehung setzen und zeigen wie sie regional, d.h. nach Landkreisen/kreisfreien Städten, verteilt sind. Dazu wird außer der Datei `stammdaten.txt` noch die `ambuDaten.txt` benötigt, aus der das Geschlecht über den gemeinsamen Pseudonymschlüssel abgefragt werden kann.

Die unten stehende Abbildung (5.6) zeigt ein zweistufig gestapeltes Balkendiagramm, aus dem für jeden Landkreis bzw. kreisfreie Stadt die Prozentzahlen der versicherten Frauen und Männer abgelesen werden können. Im arithmetischen Mittel ergibt sich ein Männeranteil von 42.74%, respektive 57.25% für die Frauen. Zu diesen Durchschnittswerten variieren die einzelnen Prozentwerte der Regionen nur sehr wenig, sodass, wie auch in der Grafik anschaulich dargestellt, im ganzen Bundesland eine nahezu homogene Verteilung vorliegt.

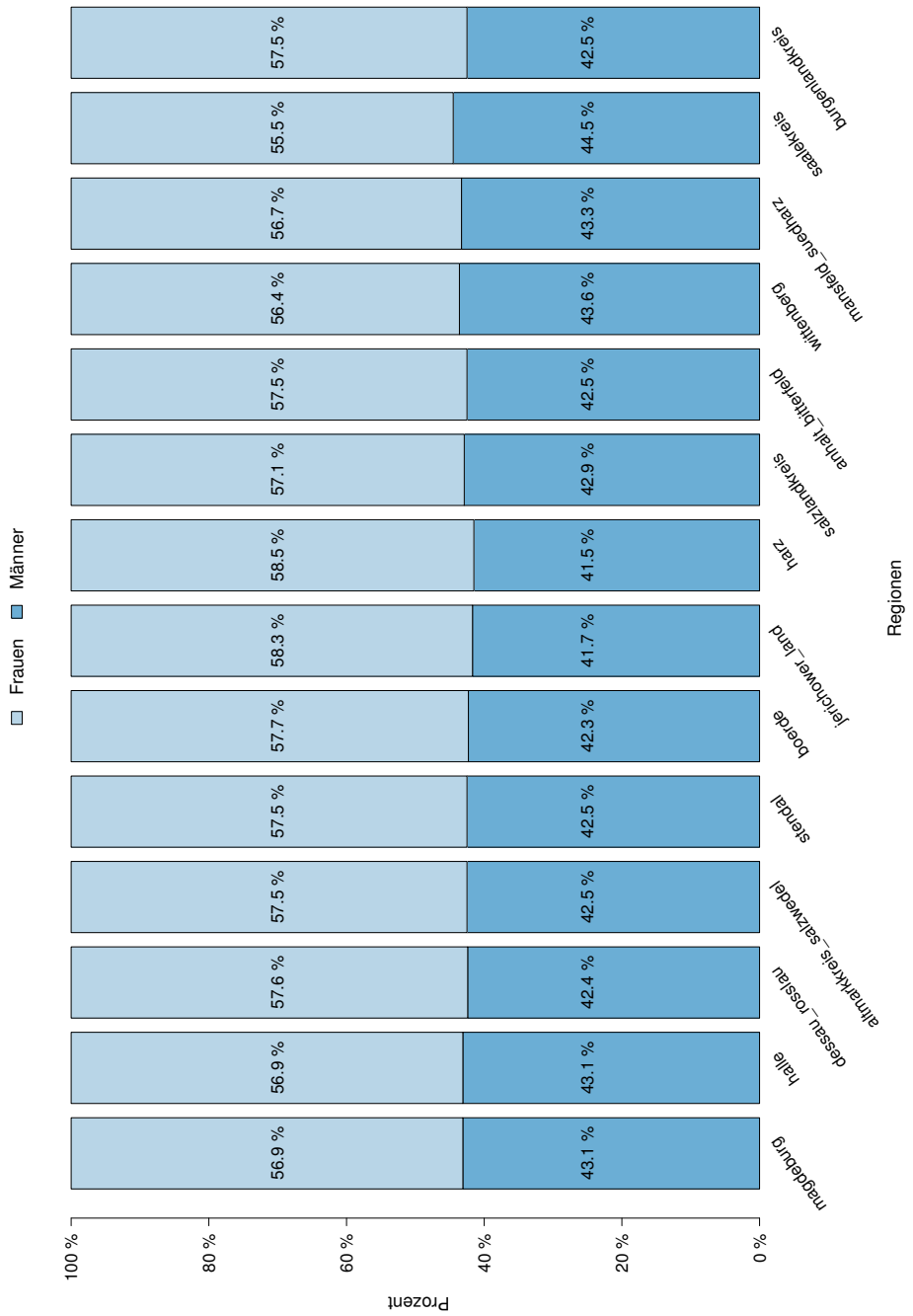

Abb. 5.6.: Prozentuale Geschlechterverteilung

Die Abbildung 5.6 ist aus folgendem Quellcode entstanden:

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "GESCHLECHT" werden extrahiert
ambu_gender <- ambudaten[c(1,7)]
# Funktion um den prozentualen Anteil der Frauen und Männer für jeden
    Landkreis zu erhalten
gender_verteilung <- function(landkreis,ambu_gender) {

  # Verbinden der Stammdaten des Landkreises mit Alter und Geschlecht der
      Ambudaten
  tmp_landkreis <- unique(merge(landkreis,ambu_gender,by="PSEUDONYM"))
  # Prozentanteil für Frauen und Männer berechnen
  landkreis_data <- rbind(
     round(x=length(subset(tmp_landkreis,tmp_landkreis$GENDER=="M")[,1]) /
         length(tmp_landkreis$PSEUDONYM)*100,digits=1),
     round(x=length(subset(tmp_landkreis,tmp_landkreis$GENDER=="F")[,1]) /
         length(tmp_landkreis$PSEUDONYM)*100,digits=1)
  )
  # Rückgabe
  return(landkreis_data)
}
# Ergebnismatrix für den Plot erstellen
gender_plot <- cbind(
  gender_verteilung(magdeburg,ambu_gender),
  gender_verteilung(halle,ambu_gender),
  gender_verteilung(dessau_rosslau,ambu_gender),
  gender_verteilung(altmarkkreis_salzwedel,ambu_gender),
  gender_verteilung(stendal,ambu_gender),
  gender_verteilung(boerde,ambu_gender),
  gender_verteilung(jerichower_land,ambu_gender),
  gender_verteilung(harz,ambu_gender),
  gender_verteilung(salzlandkreis,ambu_gender),
  gender_verteilung(anhalt_bitterfeld,ambu_gender),
  gender_verteilung(wittenberg,ambu_gender),
  gender_verteilung(mansfeld_suedharz,ambu_gender),
  gender_verteilung(saalekreis,ambu_gender),
  gender_verteilung(burgenlandkreis,ambu_gender) )
# Überschriften setzen
rownames(gender_plot) <- c("proz_male","proz_female")
colnames(gender_plot) <- landkreise
### PLOT BEGINNT ###
```

```
40 # Seitenabstand
41 par(mar=c(9,5,7,7))
42 # Plotten
43 balken <- barplot(height=gender_plot,beside=FALSE,main="
       Geschlechterverteilung in den Landkreisen (Sachsen-Anhalt)",col=c(farben
       [3],farben[4]),axes=FALSE,axisnames=FALSE)
44 # Legende anzeigen
45 legend(x=17.5,y=100,legend=c("Frauen","Männer"),fill=c(farben[4],farben[3]),
       cex=0.8,xpd=TRUE)
46 # Prozente der Männer anzeigen
47 text(x=balken+0.35,y=25,labels=paste(gender_plot[1,],"%"),cex=0.8,xpd=TRUE,
       adj=1)
48 # Prozente der Frauen anzeigen
49 text(x=balken+0.35,y=70,labels=paste(gender_plot[2,],"%"),cex=0.8,xpd=TRUE,
       adj=1)
50 # Y-Achsenbereich festlegen und Schritte definieren
51 ybereich <- pretty(x=c(0,100),n=5)
52 # Y-Achse anzeigen
53 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos
       =-0.35)
54 # Y-Achsen-Beschriftung
55 mtext(text="Prozent",side=2,line=3)
56 # X-Achsen-Beschriftung
57 text(x=balken+0.2,y=-4,labels=landkreise,cex=0.9,srt=35,xpd=TRUE,adj=1)
58 mtext(text="Regionen",side=1,line=6)
```

Die Zeilen 4-6 laden die `ambuDaten.txt` und wählen die benötigten Spalten für diese Analyse aus. Ab Zeile 8 bis einschließlich 19 wird eine Funktion `gender_verteilung()` definiert, die den prozentualen Anteil der Frauen bzw. Männer berechnet und als Matrix zurückgibt. Dies geschieht durch die Kombination der drei Befehle `subset()`, die jeweils nur Frauen bzw. Männer herausfiltert, `length()`, die die Gesamtanzahl zurückgibt und `round()`, die das Ergebnis auf eine Nachkommastelle rundet. Nachfolgende Zeilen (bis einschließlich 35) definieren eine komplette Ergebnismatrix, die mit einem `cbind()`-Befehl alle Funktionsaufrufe, mit den einzelnen Landkreisen als Argument, verbindet. Aus Gründen der Übersichtlichkeit werden Zeilen- und Spaltenüberschriften gesetzt (Zeile 37 und 38). Mit dem nachfolgenden Programmcode wird das Diagramm und Zusatzelemente wie Achsen, Texte und Legende erstellt. Auf eine detaillierte Beschreibung der weiteren Befehle wird hier und im weiteren Verlauf verzichtet, da sie für die zuvor gezeigten Abbildungen bereits ausführlich beschrieben wurden.

5.4. Regionale Altersverteilung in Sachsen-Anhalt

In diesem Abschnitt soll der Datenbestand hinsichtlich der Altersstruktur untersucht werden. Die erste Aufgabe besteht in der Ermittlung des regions- und geschlechtsspezifischen Durchschnitts- und Maximalalters, um herauszufinden, inwieweit sich bestimmte Landstriche in Sachsen-Anhalt bezüglich der Altersstruktur abheben.
Für den Fall, dass Auffälligkeiten zu erkennen sind, könnte dies ein erster Aspekt und Anhaltspunkt sein, um die Bereitstellung für etwaige Maßnahmen hinsichtlich der regionalen Anpassung der Vorsorge und des medizinisches Angebots zu verändern.
Die zweite Aufgabe besteht darin, eine Altersverteilung, gruppiert nach selbstdefinierten Altersklassen, vorzunehmen. Für die Betrachtung notwendige Daten werden aus den Stamm- und ambulanten Daten herausgezogen.
Die Ergebnisse werden in Form eines Diagramms (Abbildung 5.7) und zweier Tabellen (Tabelle 5.1 und Tabelle 5.2) illustriert.

Das Balkendiagramm (Abbildung 5.7) zeigt für jeden Landkreis/kreisfreie Stadt zwei gruppierte Balken, die das Geschlecht repräsentieren. Die blauen Zahlenwerte stellen das jeweilige Durchschnittsalter von Männern und Frauen in der Region dar. Die roten Zahlen geben das maximal vorkommende Alter eines Versicherten an.
Aus der Grafik ist abzulesen, dass die älteste von der AOK erfasste Person, eine Frau aus dem Salzlandkreis im Alter von 108 Jahren ist. Weiterhin ist zu erkennen, dass das Durchschnittsalter der Männer im Mittel um 5 Jahre geringer ist, als das der Frauen. Diese Feststellung steht ebenfalls im Einklang mit dem Ergebnis der Abbildung 5.5, sowie den vom statistischen Bundesamt ermittelten Daten zur Lebenserwartung. Diese besagen, dass Frauen im Durchschnitt eine höhere Lebenserwartung als Männer haben, wodurch das Durchschnitts- und Maximalalter höher liegen [22]. Das niedrigste Durchschnittsalter ist in den städtischen Gebieten, u.a. in Halle mit 49 Jahren und in Magdeburg mit 52 Jahren für die Männer, zu finden.

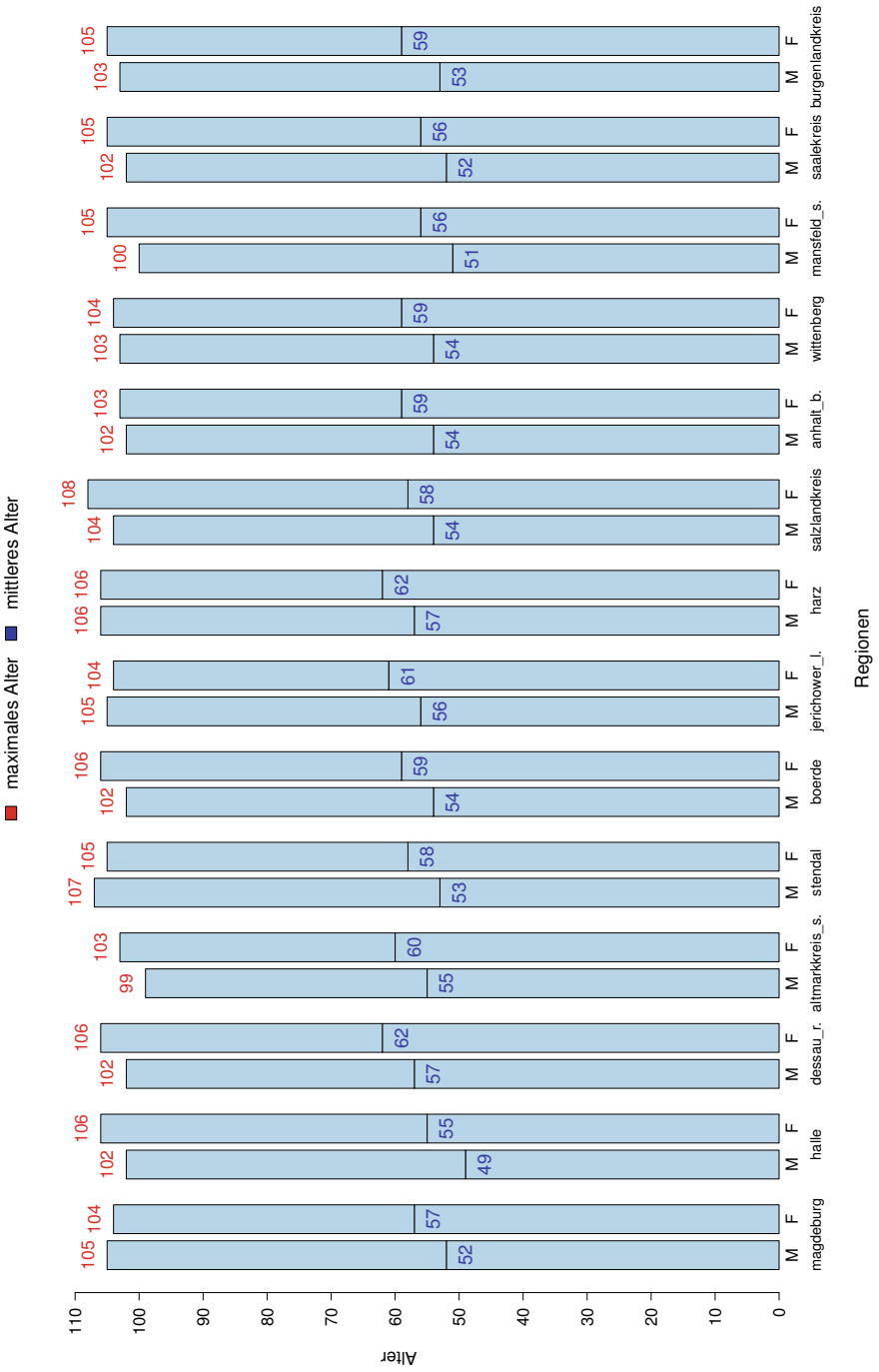

Abb. 5.7.: Durchschnitts- und Maximalalter in Sachsen-Anhalt

Landkreis	Geschlecht	0-20 J	21-40 J	41-60 J	61-80 J	81-100 J	101-120 J
Magdeburg	F	3696	4505	6519	11473	6457	20
	M	3514	3712	6051	8868	2547	4
Halle	F	3717	4043	5492	8073	5182	20
	M	3379	3280	5391	6263	1736	2
Dessau-Roßlau	F	1087	1200	2307	4795	2893	14
	M	963	977	2277	3710	1098	2
Altmarkkreis-	F	1365	1326	3219	5222	2732	8
Salzwedel	M	1294	1226	2792	3784	1140	0
Stendal	F	2040	1989	4285	6555	3342	10
	M	1810	1615	3756	4961	1296	2
Börde	F	2376	2869	6121	9260	4919	17
	M	2317	2270	5417	6886	1819	2
Jerichower Land	F	1194	1269	2988	5033	2773	15
	M	1060	964	2701	3698	1068	3
Harz	F	2696	2949	6760	12713	7360	36
	M	2509	2332	6028	9465	2701	5
Salzlandkreis	F	3722	4040	8481	13355	6819	22
	M	3449	3498	7690	10194	2536	4
Anhalt-Bitterfeld	F	2659	2864	6172	9958	5356	23
	M	2427	2513	5595	7521	1913	2
Wittenberg	F	1979	2280	5095	7978	3962	13
	M	1999	2001	4714	6224	1509	2
Mansfeld-Südharz	F	2207	2395	4938	6512	3288	7
	M	2174	2074	4528	4776	1203	0
Saalekreis	F	2440	2646	5184	6890	3710	8
	M	2242	2320	5102	5637	1407	3
Burgenlandkreis	F	2957	3207	6823	10067	5740	20
	M	2782	2739	6164	7505	2035	4

Tab. 5.1.: Absolute Häufigkeitstabelle der Altersgruppen

Landkreis	Geschlecht	0-20 J	21-40 J	41-60 J	61-80 J	81-100 J	101-120 J
Magdeburg	F	6.44 %	7.85 %	11.36 %	20 %	11.26 %	0.04 %
	M	6.13 %	6.47 %	10.55 %	15.46 %	4.44 %	0.01 %
Halle	F	7.98 %	8.68 %	11.79 %	17.33 %	11.12 %	0.04 %
	M	7.25 %	7.04 %	11.57 %	13.45 %	3.73 %	0.0 %
Dessau-Roßlau	F	5.1 %	5.63 %	10.82 %	22.49 %	13.57 %	0.07 %
	M	4.52 %	4.58 %	10.68 %	17.4 %	5.15 %	0.01 %
Altmarkkreis-	F	5.66 %	5.5 %	13.35 %	21.66 %	11.33 %	0.03 %
Salzwedel	M	5.37 %	5.08 %	11.58 %	15.7 %	4.73 %	0.0 %
Stendal	F	6.44 %	6.28 %	13.53 %	20.7 %	10.56 %	0.03 %
	M	5.72 %	5.1 %	11.86 %	15.67 %	4.09 %	0.01 %
Börde	F	5.37 %	6.48 %	13.83 %	20.92 %	11.11 %	0.04 %
	M	5.23 %	5.13 %	12.23 %	15.55 %	4.11 %	0.0 %
Jerichower Land	F	5.25 %	5.57 %	13.12 %	22.11 %	12.18 %	0.07 %
	M	4.66 %	4.23 %	11.86 %	16.24 %	4.69 %	0.01 %
Harz	F	4.85 %	5.31 %	12.17 %	22.88 %	13.25 %	0.06 %
	M	4.52 %	4.2 %	10.85 %	17.04 %	4.86 %	0.01 %
Salzlandkreis	F	5.83 %	6.33 %	13.29 %	20.93 %	10.69 %	0.03 %
	M	5.41 %	5.48 %	12.05 %	15.98 %	3.97 %	0.01 %
Anhalt-Bitterfeld	F	5.66 %	6.09 %	13.13 %	21.19 %	11.39 %	0.05 %
	M	5.16 %	5.35 %	11.9 %	16 %	4.07 %	0.0 %
Wittenberg	F	5.24 %	6.04 %	13.49 %	21.13 %	10.49 %	0.03 %
	M	5.29 %	5.3 %	12.48 %	16.48 %	4.0 %	0.0 %
Mansfeld-Südharz	F	6.47 %	7.02 %	14.48 %	19.1 %	9.64 %	0.02 %
	M	6.38 %	6.08 %	13.28 %	14.01 %	3.53 %	0.0 %
Saalekreis	F	6.49 %	7.04 %	13.79 %	18.33 %	9.87 %	0.02 %
	M	5.96 %	6.17 %	13.57 %	15 %	3.74 %	0.01 %
Burgenlandkreis	F	5.91 %	6.41 %	13.63 %	20.12 %	11.47 %	0.04 %
	M	5.56 %	5.47 %	12.32 %	15 %	4.07 %	0.01 %

Tab. 5.2.: Prozentuale Häufigkeitstabelle der Altersgruppen

Die beiden Tabellen zeigen die Altersverteilung der AOK-Versicherten in selbstdefinierten Bereichen. Dabei beinhaltet die erste Spalte die Regionen, die zweite die Geschlechter (`F` für Frauen und `M` für Männer) und die restlichen die Altersgruppen. Die Tabelle 5.1 listet die absolute und die Tabelle 5.2 die prozentuale Häufigkeit der Versicherten pro Gebiet und Altersgruppe auf.

Die prozentualen Werte beziehen sich bei Männern und Frauen auf den entsprechenden Landkreis/kreisfreie Stadt, d.h. zur Berechnung der Prozentangaben wird die Gesamtanzahl der Versicherungsnehmer (Männer und Frauen) innerhalb des Landkreises benutzt.

Anhand der Tabellen wird sichtbar, wie sich mit zunehmendem Alter der Abstand zwischen Männern und Frauen vergrößert. In der ersten Altersgruppe von 0-20 Jahren ist die Verteilung annähernd ausgeglichen. Ab der nächsten Altersklasse steigt der Frauenanteil, im Vergleich zu den Männern, sichtbar an und vergrößert sich mit jeder weiteren Altersstufe. Zum Beispiel beträgt die prozentuale Differenz zwischen Männern und Frauen in der Altersgruppe von 81-100 Jahren im Mittel ca. 8%.

Eine geschlechtsunabhängige Betrachtung zeigt, dass die Altersgruppe 61-80 Jahre am stärksten vertreten ist, welches deutlich über dem in Abbildung 5.7 berechneten Durchschnittsalter liegt. Diese Feststellung steht im Einklang mit dem sog. Median (Zentralwert). Der Median beschreibt das Zentrum einer geordneten Zahlenreihe [26, S. 52 f.]. Das Medianalter ist ein Maß für die Alterung in den Regionen und beträgt 62 Jahre (zum Vergleich beträgt das Durchschnittsalter 56 Jahre). Diese Differenz entsteht durch eine ungleichmäßige Verteilung der Altersgruppen. Das Medianalter bzw. der Medianwert zeigt an, dass 50% der AOK-Versicherten in Sachsen-Anhalt jünger und 50% älter sind als 62 Jahre.

Der Quellcode der Abbildung 5.7:

```
1  # initial laden und ausführen
2  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
3  # Ambudaten einlesen und bereinigen
4  ambudaten <- ambudaten_laden()
5  # Die Spalten "PSEUDONYM", "ALTER", "GESCHLECHT" werden extrahiert
6  ambu_age <- ambudaten[c(1,6,7)]
7  # Funktion um das Durchschnitts- und Maximalalter der Männer und Frauen für
       jeden Landkreis zu erhalten
8  age_verteilung <- function(landkreis,ambu_age) {
9    # Verbinden der Stammdaten pro LK mit Alter und Geschlecht der Ambudaten
10   tmp_landkreis <- unique(merge(landkreis,ambu_age,by="PSEUDONYM"))
```

```
11    # Prozentanteil für die Männer berechnen
12    subset_m <- subset(x=tmp_landkreis$AGE,subset=(tmp_landkreis$GENDER=="M"))
13    landkreis_data_m <- rbind(
14      round(mean(subset_m)),
15      max(subset_m) - round(mean(subset_m))
16    )
17    # Prozentanteil für die Frauen berechnen
18    subset_f <- subset(x=tmp_landkreis$AGE,subset=(tmp_landkreis$GENDER=="F"))
19    landkreis_data_f <- rbind(
20      round(mean(subset_f)),
21      max(subset_f) - round(mean(subset_f))
22    )
23    # Ergebnis für Männer und Frauen verbinden
24    landkreis_data <- cbind(landkreis_data_m,landkreis_data_f)
25    # Rückgabe
26    return(landkreis_data)
27  }
28  # Ergebnismatrix für den Plot erstellen
29  age_plot <- cbind(
30    age_verteilung(magdeburg,ambu_age),
31    age_verteilung(halle,ambu_age),
32    age_verteilung(dessau_rosslau,ambu_age),
33    age_verteilung(altmarkkreis_salzwedel,ambu_age),
34    age_verteilung(stendal,ambu_age),
35    age_verteilung(boerde,ambu_age),
36    age_verteilung(jerichower_land,ambu_age),
37    age_verteilung(harz,ambu_age),
38    age_verteilung(salzlandkreis,ambu_age),
39    age_verteilung(anhalt_bitterfeld,ambu_age),
40    age_verteilung(wittenberg,ambu_age),
41    age_verteilung(mansfeld_suedharz,ambu_age),
42    age_verteilung(saalekreis,ambu_age),
43    age_verteilung(burgenlandkreis,ambu_age)
44    )
45  # Überschriften setzen
46  rownames(age_plot) <- c("mean", "max-mean")
47  ### PLOT BEGINNT ###
48  # Seitenabstand
49  par(mar=c(8,5,8,3))
50  # Abstände der Barplots
51  a1 <- c(0.3)
52  a2 <- c(1.3)
53  abstand <- c(a1,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,
        a1,a2,a1,a2,a1,a2,a1)
```

```
54 # Plotten
55 balken <- barplot(height=age_plot,beside=FALSE,main="Altersverteilung in den
       Landkreisen (Sachsen-Anhalt)",col=farben[4],axes=FALSE,axisnames=FALSE,
       space=abstand,ylim=c(0,1.1*max(colSums(age_plot))))
56 # Legende anzeigen
57 legend(x=46.5,y=140,legend=c("maximales Alter","mittleres Alter"),fill=c(
       farben_2[2],farben_2[13]),cex=0.8,xpd=TRUE)
58 # Mittleres Alter anzeigen
59 text(x=balken,y=age_plot[1,]-3,labels=age_plot[1,],col=farben_2[13])
60 # Maximales Alter anzeigen
61 max_alter <- rbind(colSums(age_plot))
62 text(x=balken,y=max_alter+3,labels=max_alter,col=farben_2[2])
63 # Y-Achsenbereich festlegen und Schritte definieren
64 ybereich <- pretty(x=c(0,max(colSums(age_plot))),n=12)
65 # Y-Achse anzeigen
66 axis(side=2,at=ybereich,labels=ybereich,las=2,cex.axis=0.9,pos=-0.85)
67 # Y-Achsen-Beschriftung
68 mtext(text="Alter",side=2,line=2)
69 # X-Achsen-Beschriftung
70 mtext(text="Regionen",side=1,line=5.5)
71 # Geschlecht anzeigen
72 text(x=balken,y=-4,labels=rep(x=c("M","F"),times=14),xpd=TRUE,cex=0.8)
73 # Regionen anzeigen
74 mtext(text=landkreise,side=1,at=balken[seq(from=1,to=length(balken),by=2)
       ]+0.7,line=2,cex=0.65)
```

Die Zeilen 1 bis 7 laden, wie bereits bekannt, die `initial.R` und `ambuDaten.txt`. In Zeile 8 beginnt die Funktion `age_verteilung()`, die jeweils das mittlere sowie das maximale Alter für Männer und Frauen eines Landkreises berechnet und als Matrix an den entsprechenden Funktionsaufruf zurückgibt. Innerhalb der Funktion werden zu Beginn, mit Hilfe der `merge()`-Funktion, die Spalten *ALTER* und *GESCHLECHT* der `ambudaten` mit den Stammdaten bzw. den in der `initial.R` definierten Landkreisen, verbunden. Der `unique()`-Befehl entfernt zum Schluss doppelte oder mehrfach auftretende Pseudonyme (Zeile 10). Anschließend werden über ein `rbind()`, das durch die `mean()`- und `max()`-Funktion, jeweils für Mann und Frau, bestimmte Durchschnitts- und Maximalalter, den Variablen `landkreis_data_m` respektive `landkreis_data_f` zugeordnet. Der `subset()`-Befehl dient zur Filterung der beiden Geschlechter (Zeile 12 und 18). Am Ende der Funktion, siehe Zeilen 24-26, werden alle Informationen zusammengeführt und übergeben.

Mit der in den Zeilen 29-44 definierten Variablen `age_plot` wird, ähnlich wie im Quelltext des Unterkapitels 5.3 beschrieben, eine Ergebnismatrix für den späteren Plot (`barplot()`) erzeugt. Nachfolgende Befehle sind für die Darstellung des Balkendiagramms zuständig.
Im Folgenden ist der Quellcode für die Tabelle 5.1 und Tabelle 5.2 gelistet:

```
1  # Variablen setzen (Altersgruppen)
2  alter_unterteilung_breaks <- c(0,20,40,60,80,100,120)
3  alter_unterteilung_labels <- c("0-20 J","21-40 J","41-60 J","61-80 J",
                                  "81-100 J","101-120 J")
4  # initial laden und ausführen
5  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
6  # Ambudaten einlesen und bereinigen
7  ambudaten <- ambudaten_laden()
8  # Die Spalten "PSEUDONYM", "ALTER", "GESCHLECHT" werden extrahiert
9  ambu_age <- ambudaten[c(1,6,7)]
10 # Verbinden der einzelnen Landkreise mit den Ambudaten
11 magdeburg_age <- unique(merge(magdeburg,ambu_age,by="PSEUDONYM"))
12 { ... }

14 # Gruppierung jedes Pseudonyms eines Landkreises in die Altersgruppen
15 magdeburg_tmp2 <- cut(x=magdeburg_age$AGE,breaks=alter_unterteilung_breaks,
                        labels=alter_unterteilung_labels)
16 { ... }
17 # Ergebnismatrix erzeugen
18 final_landkreise <- cbind(
19     t(table(magdeburg_age$GENDER,magdeburg_tmp2)),
20     t(table(halle_age$GENDER,halle_tmp2)),
21     t(table(dessau_rosslau_age$GENDER,dessau_rosslau_tmp2)),
22     t(table(altmarkkreis_salzwedel_age$GENDER,altmarkkreis_salzwedel_tmp2)),
23     t(table(stendal_age$GENDER,stendal_tmp2)),
24     t(table(boerde_age$GENDER,boerde_tmp2)),
25     t(table(jerichower_land_age$GENDER,jerichower_land_tmp2)),
26     t(table(harz_age$GENDER,harz_tmp2)),
27     t(table(salzlandkreis_age$GENDER,salzlandkreis_tmp2)),
28     t(table(anhalt_bitterfeld_age$GENDER,anhalt_bitterfeld_tmp2)),
29     t(table(wittenberg_age$GENDER,wittenberg_tmp2)),
30     t(table(mansfeld_suedharz_age$GENDER,mansfeld_suedharz_tmp2)),
31     t(table(saalekreis_age$GENDER,saalekreis_tmp2)),
32     t(table(burgenlandkreis_age$GENDER,burgenlandkreis_tmp2))
33 )
34 # Summieren der Männer und Frauen für jeweils einen Landkreis für die
```

```
    prozentuale Betrachtung
summe <- c(
  sum(final_landkreise[,1],final_landkreise[,2]),
  sum(final_landkreise[,3],final_landkreise[,4]),
  sum(final_landkreise[,5],final_landkreise[,6]),
  sum(final_landkreise[,7],final_landkreise[,8]),
  sum(final_landkreise[,9],final_landkreise[,10]),
  sum(final_landkreise[,11],final_landkreise[,12]),
  sum(final_landkreise[,13],final_landkreise[,14]),
  sum(final_landkreise[,15],final_landkreise[,16]),
  sum(final_landkreise[,17],final_landkreise[,18]),
  sum(final_landkreise[,19],final_landkreise[,20]),
  sum(final_landkreise[,21],final_landkreise[,22]),
  sum(final_landkreise[,23],final_landkreise[,24]),
  sum(final_landkreise[,25],final_landkreise[,26]),
  sum(final_landkreise[,27],final_landkreise[,28])
)
# Prozentualen Anteil jeder Altersgruppe berechnen (bezogen auf ihren
    Landkreis)
final_landkreise_proz <- NULL
for (i in 1:length(final_landkreise[1,])) {final_landkreise_proz <- cbind(
    final_landkreise_proz,final_landkreise[,i] * 100 / rep(x=summe,each=2)[i
    ])}
### CSV Tabellen erzeugen ###
# Prozentuale Tabelle
tabelle <- cbind("Geschlecht"=rep(x=c("F","M"),times=14))
tabelle <- cbind(tabelle,round(x=t(final_landkreise_proz),digits=2))
tabelle[,2:7] <- paste(tabelle[,2:7],"%",sep="")
# Überschriften setzen
rownames(tabelle) <- c("Magdeburg","","Halle","","Dessau-Rosslau","","
    Altmarkkreis-Salzwedel","","Stendal","","Börde","","Jerichower Land","","
    Harz","","Salzlandkreis","","Anhalt-Bitterfeld","","Wittenberg","","
    Mansfeld-Südharz","","Saalekreis","","Burgenlandkreis","")
### Tabelle als CSV auf Festplatte speichern (prozentuale Tabelle)
write.table(x=cbind(Landkreise=rownames(tabelle),tabelle),file="CSV/
    altersverteilung_pro_lk_prozentual.csv",sep=";",row.names=FALSE,col.names
    =TRUE,fileEncoding="latin1")
# Absolute Tabelle
tabelle <- cbind("Geschlecht"=rep(x=c("F","M"),times=14))
tabelle <- cbind(tabelle,t(final_landkreise))
# Überschriften setzen
rownames(tabelle) <- c("Magdeburg","","Halle","","Dessau-Rosslau","","
    Altmarkkreis-Salzwedel","","Stendal","","Börde","","Jerichower Land","","
    Harz","","Salzlandkreis","","Anhalt-Bitterfeld","","Wittenberg","","
```

```
        Mansfeld-Südharz","","Saalekreis","","Burgenlandkreis","")
68  ### Tabelle als CSV auf Festplatte speichern (absolute Tabelle)
69  write.table(x=cbind(Landkreise=rownames(tabelle),tabelle),file="CSV/
        altersverteilung_pro_lk_absolut.csv",sep=";",row.names=FALSE,col.names=
        TRUE,fileEncoding="latin1")
```

Der R-Quellcode der Tabellen ist so aufgebaut, dass die Altersgruppen bei Bedarf beliebig fein- oder grobgranular eingestellt werden können. Dazu muss der in Zeile 2 erstellte numerische Vektor `alter_unterteilung_breaks` angepasst werden. Die nachfolgende Zeile beschriftet die einzelnen Spalten und muss pro Altersgruppe je ein Element besitzen (Zeile 3). Ab Zeile 11 werden für jede Pseudonymnummer aus dem jeweiligen Landkreis/kreisfreien Stadt deren Alter und Geschlecht aus den ambulanten Daten, mittels der `merge()`-Funktion, herausgezogen und zusammengeführt. Dies ist im Quelltext exemplarisch für Magdeburg dargestellt. Der folgende Quellcodeabsatz kategorisiert mit Hilfe des `cut()`-Befehls und dessen Argument `breaks=` das Alter in die oben definierten Altersgruppen und bestimmt durch das Argument `labels=` die Beschriftung der Gruppierung für jede Region. In Zeile 18 werden, wie bei den obigen Analysen, alle Regionen durch ein `cbind()` in eine Ergebnismatrix (`final_landkreise`) überführt. Zusätzlich wird hier die `table()`-Funktion auf jede Region angewandt, die die absolute Häufigkeit, jeweils gruppiert nach Geschlecht, bestimmt. Der `t()`-Befehl bewirkt, dass die Matrix transponiert wird, d.h. die Spalten werden zu Zeilen und umgekehrt. Folglich bilden die Regionen die Spalten und die Altersgruppen die Zeilen.
Der Vektor `summe` in Zeile 35-50 summiert die Anzahl der Männer und Frauen jeweils für die Region, die für die prozentuale Berechnung in der `for()`-Schleife (Zeile 52-53) benötigt wird. Die prozentuale Ergebnismatrix heißt `final_landkreise_proz`.
Nachfolgende Zeilen beschriften die beiden Tabellen und speichern sie als CSV-Datei auf die Festplatte (Zeile 62 und 69).

5.5. Krankheitsbilder in Sachsen-Anhalt

Wie auf Seite 29 bereits erläutert, sind die ICD-Codes Teil eines Klassifikationssystems zur Verschlüsselung von Krankheitsdiagnosen in der ambulanten und stationären Versorgung. Das Klassifikationssystem wird von der Weltgesundheitsorganisation (WHO) verwaltet und liegt seit 2012 in der 10. Revision (ICD-10) vor. In Deutschland sind Ärzte und medizinische Einrichtungen verpflichtet, Diagnosen nach ICD-10 in einem für Deutschland modifizierten System, dem sog. ICD-10 German Modification (GM), zu verschlüsseln. Die Verpflichtung, diesen Katalog anzuwenden, ist unter anderem die Tatsache, dass auf dessen Grundlage die medizinischen Vergütungs- und Finanzierungssysteme basieren, zum Beispiel die Abrechnung der Ärzte mit den Krankenkassen [6]. Um zu gewährleisten, dass die von den Ärzten festgestellten Erkrankungen so präzise wie möglich diagnostiziert werden, enthält der Katalog weit über 10.000 sog. endständige Codes [8]. Unter ihnen befinden sich in jedem Kapitel ebenfalls nicht spezifizierte Diagnoseschlüssel, die mit dem Zusatz *nicht näher bezeichnet* gekennzeichnet sind.
Die Strukturierung der fünfstelligen Diagnoseschlüssel folgt einem klaren Aufbau. Die oberste Hierarchieebene gliedert alle Krankheiten thematisch in 22 Kapitel (Kapitel I bis XXII) und teilt sich weiter in Gruppen/Bereiche und Kategorien/Subkategorien auf [6]. Um dies zu verdeutlichen, wird beispielhaft der ICD-10-Code `I13.10` aufgeschlüsselt:

Die Suche in der Tabelle 5.3 (Seite 84), in der alle 22 Kapitel zu finden sind, ergibt, dass sich der ICD-10-Code in dem Kapitel IX (`I00-I99`) bzw. *Krankheiten des Kreislaufsystems* befindet. Eine Ebene tiefer ist der Bereich `I10-I15` bzw. *Hypertonie (Hochdruckkrankheit)*, in dem die Kategorie `I13.-` bzw. *Hypertensive Herz- und Nierenkrankheit* vorzufinden ist. Die letzte Subkategorie enthält den konkreten ICD-Schlüssel `I13.10` bzw. *Hypertensive Herz- und Nierenkrankheit mit Niereninsuffizienz: Ohne Angabe einer hypertensiven Krise*, womit der ICD-10-Code eindeutig aufgeschlüsselt ist.

Dieses Unterkapitel ist sehr umfassend und unterteilt sich in mehrere Untersuchungsabschnitte, die sich ausführlich mit den Krankheitsbildern der AOK-Versicherten in Sachsen-Anhalt beschäftigen.

5.5.1. Absolute Verteilung der ICD-10-Codes

Um sich einen ersten Überblick über die Vielzahl der Erkrankungen zu verschaffen, wird zunächst ein absolutes Häufigkeitsdiagramm mit allen im ambulanten Datenbestand vorhandenen ICD-10-Codes erstellt.

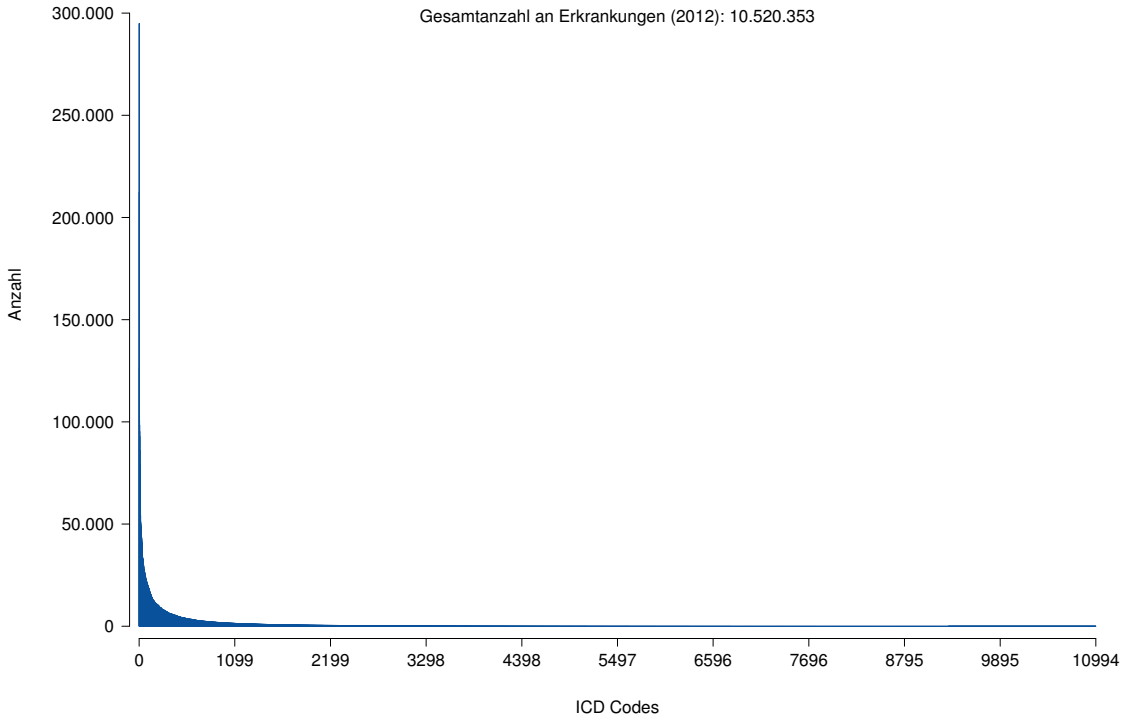

Abb. 5.8.: Gesamtübersicht ICD-10-Code-Verteilung

Die Abbildung 5.8 visualisiert die Anzahl der ambulant festgestellten Erkrankungen, welche in der `Wohlfahrt_ambuDaten.txt` registriert sind. Die X-Achse in der Grafik stellt die verschiedenen ICD-Codes dar, wobei insgesamt 10.994 unterschiedliche Diagnoseschlüssel im Datenbestand erfasst sind. Die Y-Achse spiegelt für jeden ICD-10-Code dessen absolute Häufigkeit wieder. Das Aufsummieren aller Häufigkeiten ergibt eine Zahl von ca. 10,5 Millionen, welche der Gesamtanzahl der diagnostizierten Erkrankungen al-

ler AOK-Versicherten entspricht. Setzt man diese Zahl in Relation zu den auf Seite 44 ermittelten Pseudonymen (605.842), ergeben sich im Durchschnitt für jeden einzelnen Versicherten 17 verschiedene Erkrankungen. Die Anzahl der Erkrankungen (Y-Werte) und die ICD-10-Codes (X-Werte) sind absteigend sortiert, wodurch erkennbar wird, dass sie in einem exponentiellen Verhältnis zueinander stehen. Dies wird aus dem stark abfallenden Verlauf sichtbar. Aufgrund des kleinen Maßstabes der Darstellung, sind keine genaueren Ergebnisse abzulesen, jedoch ist zu erkennen, dass von den meisten Krankheiten nur wenige Versicherte und bei einigen wenigen Erkrankungen sehr viele Versicherte betroffen sind.
Um ein detaillierteres Untersuchungsbild zu erhalten, wird der Maßstab vergrößert, sodass die einzelnen Diagnoseschlüssel sichtbar werden. Die nächste Abbildung 5.9 zeigt die häufigsten ICD-Codes in einem Balkendiagramm an.

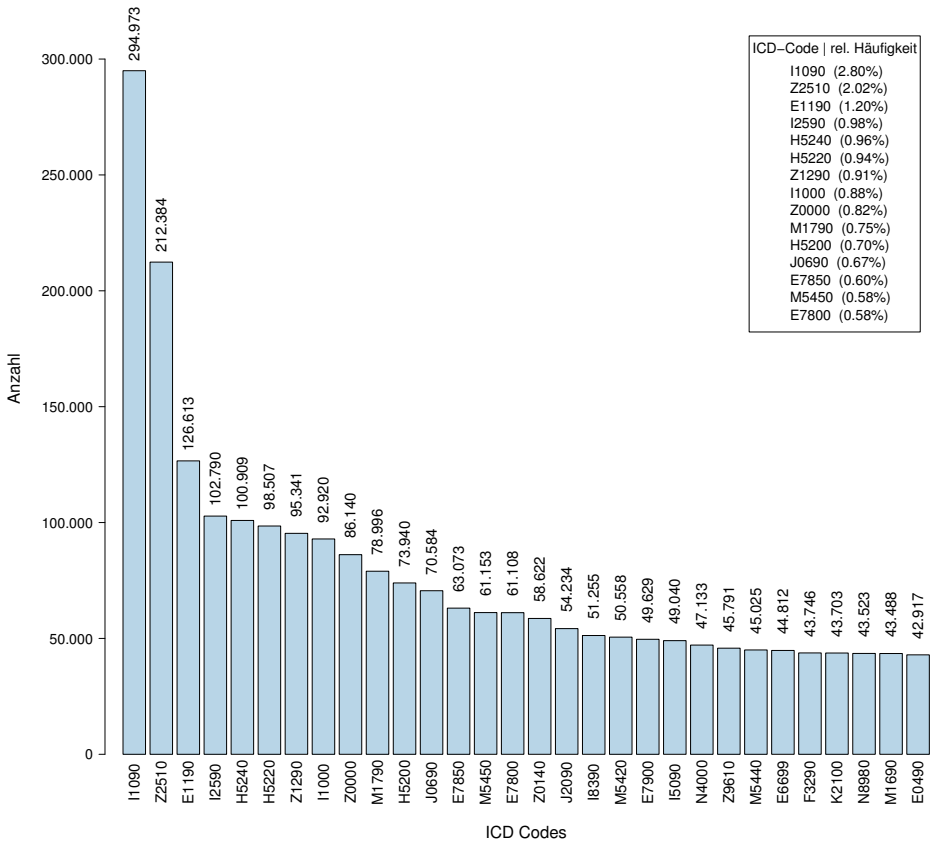

Abb. 5.9.: Die 30 häufigsten ICD-10-Codes

Die Grafik 5.9 bildet die ersten 30 ICD-Codes in einem größeren Maßstab ab. Während die Skala der Y-Achse unverändert geblieben ist, wird bei der X-Achse die Anzahl durch die jeweilige Diagnoseschlüsselbezeichnung ersetzt. Die Zahl oberhalb der Balken zeigt die absolute Häufigkeit des entsprechenden ICD-10-Codes an. Von der Erkrankung mit dem Schlüssel `I1090` sind 294.973 AOK-Versicherte betroffen, welches knapp 50% aller erfassten Personen im Datenbestand entspricht. Die zweithäufigste Diagnose ist `Z2510`, an der 212.384 Patienten leiden. Der Balkenverlauf verzeichnet noch einen weiteren großen Abfall von über 40% vom ICD-10-Code `Z2510` zum Code `E1190`, mit einem anschließenden moderat fallendem Verlauf, der sich später linear fortsetzt.
Die integrierte Tabelle listet die ersten fünfzehn ICD-10-Codes und deren prozentuale Häufigkeit auf. Für die beiden oben genannten ICD-10-Codes, ergeben sich die prozentualen Häufigkeiten von 2,80% bzw. 2,02%, bezogen auf die Gesamtanzahl von ca. 10,5 Millionen registrierten Erkrankungen.

Es folgt die Auflistung des Quellcodes der Abbildung 5.8:

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert und doppelte Zeilen
    entfernt
ambu_icd <- unique(ambudaten[c(1,3)])
# Absolute Häufigkeitstabelle der ATC-Codes erstellen
icd_abs_h <- data.frame(table("ICD"=ambu_icd$ICD))
# Häufigkeitstabelle absteigend sortieren
icd_abs_h <- data.frame(icd_abs_h[order(-icd_abs_h$Freq),])
### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(5,6,3,2))
# Plotten
barplot(height=icd_abs_h[,2],yaxt="n",main="Häufigkeitsverteilung der ICD-
    Codes",xlab="ICD Codes",ylab="Anzahl",ylim=c(0,1.1*max(icd_abs_h[,2])),
    space=0,col=farben[1],border=farben[1])
# Gesamtanzahl ICD-Codes anzeigen
mtext(text=paste("Gesamtanzahl an Erkrankungen (2012): ",format(x=length(ambu
    _icd$ICD),big.mark=".",scientific=FALSE),sep=""),side=3,cex=0.7,line
    =-0.5)
# Y-Achse definieren
```

```
19 ybereich <- pretty(x=c(0,max(icd_abs_h[,2])),n=10)
20 # Y-Achse anzeigen
21 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=
       FALSE),las=2,cex.axis=0.8,pos=-110)
22 # X-Achse definieren
23 xbereich <- round(x=seq(from=0,to=length(icd_abs_h$ICD),by=length(icd_abs_h$
       ICD)/10),digits=0)
24 # X-Achse anzeigen
25 axis(side=1,at=xbereich,cex.axis=0.8,pos=-6000)
```

In Zeile 1-5 werden die ambulanten Daten eingelesen, sodass in Zeile 6 durch den `unique()`-Befehl die mehrfach vorkommenden (identischen) Diagnosen eines Patienten entfernt werden. Dieser Schritt ist notwendig, da ein Versicherter in verschiedenen Quartalen des Jahres 2012 von einem Arzt die gleiche Diagnose erhalten kann, diese aber nur als eine Erkrankung in das Diagramm eingehen darf. Nachfolgend wird eine absolute Häufigkeitstabelle der ICD-10-Codes erstellt und absteigend, durch die `order()`-Funktion, sortiert, sodass der Diagnoseschlüssel mit den meisten Betroffenen an erster Stelle des *data.frames* steht (Zeile 8-10). Die Berechnungen dieser Analyse sind damit abgeschlossen und es beginnt die Erstellung der Grafikausgabe ab Zeile 11. Dazu wird dem Argument `height=` der `barlot()`-Funktion die zweite Spalte des *data.frames* `icd_abs_h`, die die absoluten Frequenzen enthält, übergeben. Mit dem Befehl `mtext()` in Zeile 17 wird oberhalb des Diagramms durch die `length()`-Anweisung auf die ICD-Spalte der Tabelle `icd_abs_h` die Gesamtanzahl aller Erkrankungen ermittelt und angezeigt. Der vorangestellte `format()`-Aufruf formatiert die Ausgabe für eine bessere Lesbarkeit durch Einfügen eines Punktes als 1000er-Trennzeichen. Die restlichen Befehle skalieren und erzeugen die beiden Achsen.

Der Quellcode der Abbildung 5.9:

```
1 # Variable setzen (Anzahl der häufigsten ICD-Codes)
2 # default: 30
3 anzahl_icd_codes <- 30
4 # initial laden und ausführen
5 source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
6 # Ambudaten einlesen und bereinigen
7 ambudaten <- ambudaten_laden()
8 # Die Spalten "PSEUDONYM", "ICD" werden extrahiert und doppelte Zeilen
      entfernt
```

```
9  ambu_icd <- unique(ambudaten[c(1,3)])
10 # Absolute Häufigkeitstabelle der ICD-Codes erstellen
11 icd_abs_h <- data.frame(table("ICD"=ambu_icd$ICD))
12 # Häufigkeitstabelle absteigend sortieren
13 icd_abs_h <- data.frame(icd_abs_h[order(-icd_abs_h$Freq),])
14 # Die häufigsten ICD-Codes extrahieren
15 icd_bereich <- icd_abs_h[1:anzahl_icd_codes,]
16 ### PLOT BEGINNT ###
17 # Seitenabstand
18 par(mar=c(4,6,5.5,2))
19 # Plotten
20 plot <- barplot(height=icd_bereich$Freq,main=paste("Die häufigsten ",anzahl_
       icd_codes," ICD-Codes",sep=""),axes=FALSE,axisnames=FALSE,col=farben[4],
       ylim=c(0,1.1*max(icd_bereich$Freq)))
21 # Tabelle (bzw. Legende) erzeugen
22   # prozentuale Häufigkeit
23   rel <- icd_bereich$Freq[1:15] * 100 / length(ambu_icd$ICD)
24   rel <- round(x=rel,digits=2)
25   rel_legende <- paste(icd_bereich[1:15,1],"  (",format(x=rel,nsmall=2),"%","
       )  ",sep="")
26 legend('topright',legend=rel_legende,title="ICD-Code | rel. Häufigkeit",cex
       =0.7)
27 # Anzahl jeder Säule anzeigen
28 text(x=plot,y=(icd_bereich$Freq+16000),srt=90,labels=format(x=icd_bereich$
       Freq,big.mark=".",scientific=FALSE),cex=0.7)
29 # ICD Codes jeder Säule anzeigen
30 text(x=plot,y=-15000,labels=icd_bereich$ICD,cex=0.6,srt=90,xpd=TRUE)
31 # Y-Achse Schritte definieren
32 ybereich <- pretty(x=c(0,max(icd_bereich$Freq)),n=9)
33 # Y-Achse anzeigen
34 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=
       FALSE),las=2,cex.axis=0.9,pos=-0.6)
35 # Y-Achsen-Beschriftung
36 mtext(text="Anzahl",side=2,line=4,cex=0.9)
37 # X-Achsen-Beschriftung
38 mtext(text="ICD Codes",side=1,line=2.6,cex=0.9)
```

Die Beschreibung des Quellcodes begrenzt sich auf die Unterschiede zu dem Code aus Abbildung 5.8, da nur wenige Abweichungen vorhanden sind. In Zeile 3 ist eine frei definierbare Variable `anzahl_icd_codes`, mit der die Anzahl der in der Grafik darzustellenden Diagnoseschlüssel bestimmt werden kann. Als Standardwert für die Abbildung 5.9 ist die Zahl 30 eingetragen. In der Zeile 15 ist ein zusätzlicher Schritt im Vergleich zum vorheri-

gen Quellcode erforderlich, der je nach Wertzuweisung der Variablen `anzahl_icd_codes` eine Anzahl an ICD-Codes auswählt und in den *data.frame* `icd_bereich` speichert, der für den `barplot()` genutzt wird. Ein weiterer Unterschied ist die Tabelle mit den ersten fünfzehn ICD-10-Codes in prozentualer Darstellung (ab Zeile 21). Dazu werden in Zeile 23 und 24 die Prozentwerte berechnet und mit dem `legend()`-Befehl dargestellt. In Zeile 28 bzw. 30 wird durch die `text()`-Funktion die Häufigkeitsanzahl durch die Frequenz-Spalte des *data.frames* `icd_bereich`, sowie der jeweilige Diagnoseschlüssel durch die ICD-Spalte `icd_bereich$ICD` den entsprechenden Balken hinzugefügt.

5.5.2. Prozentuale Verteilung aller ICD-10-Codes in Sachsen-Anhalt

In Sachsen-Anhalt sind ländliche, industrielle, forstwirtschaftliche und städtische Gebiete vertreten. Für eine erste Abschätzung, ob bestimmte Krankheitsbilder regionale Konzentrationen aufweisen, wird das Bundesland Sachsen-Anhalt in geografisch getrennte Gebiete aufgeteilt, indem einzelne Landkreise/kreisfreie Städte zu fünf Gruppen zusammengefasst werden. Die rechts stehende Grafik zeigt zur Veranschaulichung die Einteilung in die fünf Regionen, in der die Gruppierungen der Landkreise zu erkennen sind. Im weiteren Verlauf werden die Gebiete mit *Norden, Osten, Süden, Westen* und *Mitte* bezeichnet. Um die Übersichtlichkeit des Ergebnisses zu gewährleisten, werden ebenso wie die Landkreise, auch die Diagnoseschlüssel in ihre 22 Kapitel eingeordnet. Die Aufschlüsselung der ICD-10-Kapitel ist in nachstehender Tabelle 5.3 zu finden [7].

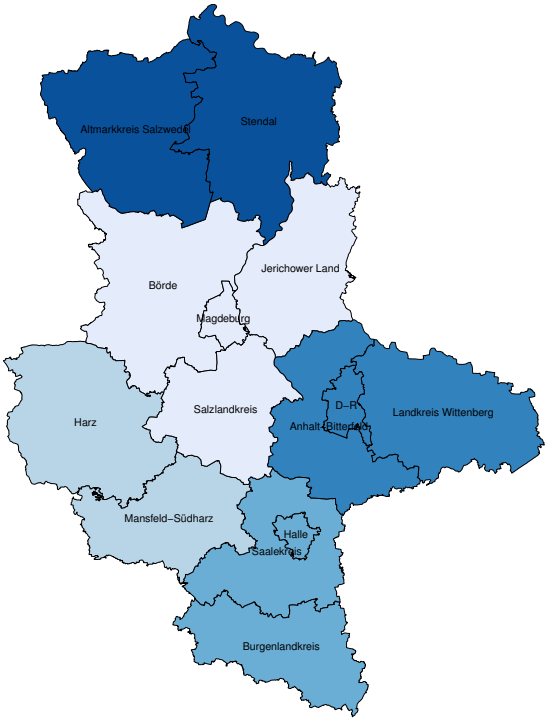

Abb. 5.10.: Sachsen-Anhalt in 5 Regionen

Für eine Gegenüberstellung der Gebiete

wird die Berechnung der Verteilung prozentual durchgeführt. Dazu werden die einzelnen ICD-10-Codes in den jeweiligen Kapiteln aufsummiert und durch die Gesamtanzahl aller Erkrankungen pro Gebiet dividiert. Durch die unterschiedlich geprägten Regionen innerhalb von Sachsen-Anhalt verspricht die Untersuchung interessante Erkenntnisse und gibt möglicherweise erste Antworten auf die Fragestellungen dieser Arbeit.

Kapitel	Code-Bereich	Titel
I	A00-B99	Bestimmte infektiöse und parasitäre Krankheiten
II	C00-D48	Neubildungen
III	D50-D90	Krankheiten des Blutes und der blutbildenden Organe sowie bestimmte Störungen mit Beteiligung des Immunsystems
IV	E00-E90	Endokrine, Ernährungs- und Stoffwechselkrankheiten
V	F00-F99	Psychische und Verhaltensstörungen
VI	G00-G99	Krankheiten des Nervensystems
VII	H00-H59	Krankheiten des Auges und der Augenanhangsgebilde
VIII	H60-H95	Krankheiten des Ohres und des Warzenfortsatzes
IX	I00-I99	Krankheiten des Kreislaufsystems
X	J00-J99	Krankheiten des Atmungssystems
XI	K00-K93	Krankheiten des Verdauungssystems
XII	L00-L99	Krankheiten der Haut und der Unterhaut
XIII	M00-M99	Krankheiten des Muskel-Skelett-Systems und des Bindegewebes
XIV	N00-N99	Krankheiten des Urogenitalsystems
XV	O00-O99	Schwangerschaft, Geburt und Wochenbett
XVI	P00-P96	Bestimmte Zustände, die ihren Ursprung in der Perinatalperiode haben
XVII	Q00-Q99	Angeborene Fehlbildungen, Deformitäten und Chromosomenanomalien
XVIII	R00-R99	Symptome und abnorme klinische und Laborbefunde, die anderenorts nicht klassifiziert sind
XIX	S00-T98	Verletzungen, Vergiftungen und bestimmte andere Folgen äußerer Ursachen
XX	V01-Y84	Äußere Ursachen von Morbidität und Mortalität
XXI	Z00-Z99	Faktoren, die den Gesundheitszustand beeinflussen und zur Inanspruchnahme des Gesundheitswesens führen
XXII	U00-U99	Schlüsselnummern für besondere Zwecke

Tab. 5.3.: Übersicht der ICD-10-Kapitel

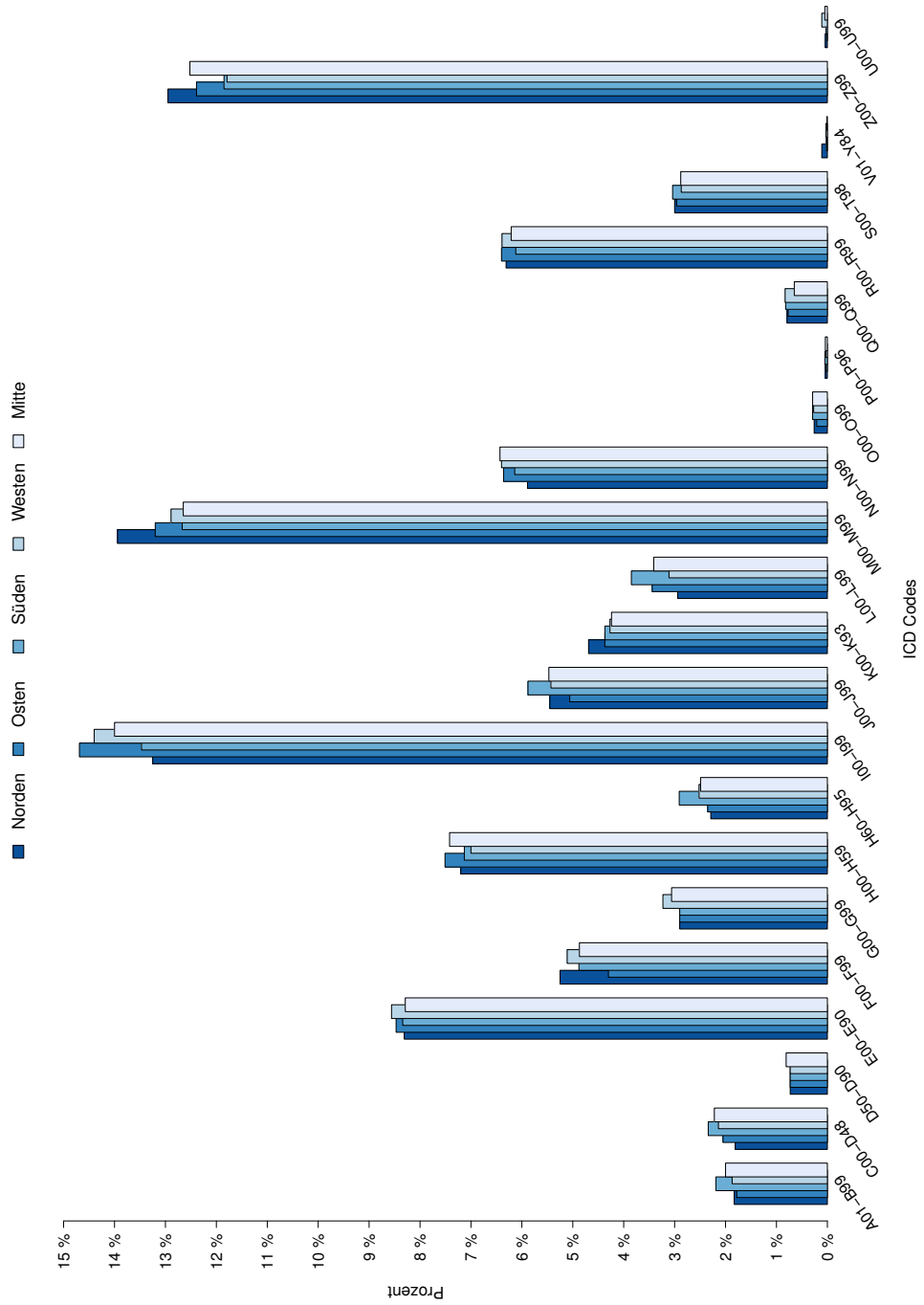

Abb. 5.11.: ICD-10-Verteilung in Sachsen-Anhalt

Die Abbildung 5.11 zeigt ein Balkendiagramm, in dem die X-Achse die ICD-10-Kapitel repräsentiert; beginnend mit dem Kapitel I (A01-B99) und endend mit dem Kapitel XXII (U00-U99). Pro Kapitel sind fünf überlappende, unterschiedlich farbige Säulen zu finden, die die geografische Unterteilung abbilden. Eine Zuordnung der Gebiete wird durch eine entsprechende Legende oberhalb der Balken vorgenommen. Die Y-Achse ist in Schritten á ein Prozent skaliert und zeigt den prozentualen Anteil eines Kapitels.

Das Kapitel IX (*Krankheiten des Kreislaufsystems*) ist in allen Gebieten am häufigsten vertreten, gefolgt von Kapitel XIII (*Krankheiten des Muskel-Skelett-Systems und des Bindegewebes*) und Kapitel XXI (*Faktoren, die den Gesundheitszustand beeinflussen und zur Inanspruchnahme des Gesundheitswesens führen*).
Lässt man die Gruppierung der Gebiete ausser Betracht bzw. fasst man die Säulen pro Kapitel zusammen, kann für das gesamte Bundesland ein Vergleich über das Aufkommen der 22 Hauptgruppen angestellt werden. Beispielsweise sind die AOK-Versicherten seltener von Erkrankungen des Kapitels II und III (*Neubildungen* bzw. *Krankheiten des Blutes und der blutbildenden Organe sowie bestimmte Störungen mit Beteiligung des Immunsystems*) mit 1% und 2%, als an Krankheiten des Kapitel IX (*Krankheiten des Kreislaufsystems*) mit knapp 15%, betroffen.
Auf die Regionen bezogen, lassen sich ebenfalls Abweichungen erkennen. Im zuletzt angesprochenem Kapitel IX weist der *Norden*, gegenüber den anderen Gebieten, die niedrigste Prozentzahl auf. Eine vorstellbare Erklärung ist die Vermutung, dass in der nördlichen Region, die vorrangig landwirtschaftlich geformt ist, bessere Umweltbedingungen und ein ausgewogeneres Lebensumfeld, ohne Stress, Lärm und Hektik, vorzufinden ist, als in einer industriell-städtisch geprägten Zone.
Andererseits hat diese Zone den größten Anteil im Kapitel XIII (Skelett- & Muskelsystem) und bestätigt die nahe liegende Vermutung, dass physisch anstrengende Tätigkeiten in der Landwirtschaft vermehrt körperliche Beschwerden hervorrufen.
Des Weiteren wird aus der Abbildung ersichtlich, dass der *Süden* in den Kapiteln I, II, VIII, X und XII hervorsticht. Stark homogen verteilt in allen Gebieten sind u.a. die Kapitel III, IV, XV, XVIII und XIX.

Bei der Auswertung und Interpretation der Grafik ist zu beachten, dass die beiden in Relation gesetzten Strukturen eine grobe Clusterung aufweisen. D.h., dass zum einen die einzelnen Postleitzahlen zu geografisch großen Zonen und zum anderen die Diagnoseschlüssel in ihre 22 Kapitel stark zusammengefasst sind.

Der Quellcode der Abbildung 5.11:

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]
# Sachsen-Anhalt in 5 Gebiete einteilen
norden <- rbind(altmarkkreis_salzwedel,stendal)
mitte <- rbind(boerde,magdeburg,jerichower_land,salzlandkreis)
osten <- rbind(wittenberg,dessau_rosslau,anhalt_bitterfeld)
sueden <- rbind(burgenlandkreis,saalekreis,halle)
westen <- rbind(harz,mansfeld_suedharz)
# Alle ICD-Codes nach Kapiteln einteilen
icd_alle <- c("A01-B99","C00-D48","D50-D90","E00-E90","F00-F99","G00-G99",
              "H00-H59","H60-H95","I00-I99","J00-J99","K00-K93","L00-L99",
              "M00-M99","N00-N99","O00-O99","P00-P96","Q00-Q99","R00-R99",
              "S00-T98","V01-Y84","Z00-Z99","U00-U99")
# Funktion um ICD-Codes pro Gebiet auf die ICD-Kapitel zu reduzieren
icd_kapitel <- function(landkreis,ambu_icd) {
  # Verbinden der Stammdaten des Landkreises mit Ambudaten (ICD-Codes)
  tmp_landkreis <- unique(merge(landkreis,ambu_icd,by="PSEUDONYM"))
  # ICD-Codes auf 3 Zeichen (Kapitelname) verkürzen
  tmp_landkreis$ICD <- substr(x=tmp_landkreis$ICD,start=1,stop=3)
  # Mehrfach vorkommende (verkürzte) ICD-Codes entfernen
  tmp_landkreis <- unique(tmp_landkreis)
  # Die Spalte "ICD" als String definieren
  tmp_landkreis$ICD <- as.character(tmp_landkreis$ICD)
  # Rückgabe
  return(tmp_landkreis)
}
# Funktion um die verkürzten ICD-Codes nach den Kapiteln zu summieren
icd_sum_kapitel <- function(tabelle,spalte,gruppenvektor) {
  # Variable definieren
  rueckgabe <- NULL
  # Filterung der ICD-Codes nach den ICD-Kapiteln
  for (i in gruppenvektor) {
```

```
34      tmp_icd <- func_bereich(tabelle,spalte,i,"c")
35      # Die Anzahl (hier length) von jedem ICD-Kapitel speichern
36      rueckgabe <- rbind(rueckgabe,length(tmp_icd[,1]))
37    }
38    # Rückgabe
39    return(rueckgabe)
40  }
41  # Norden nach den ICD-Kapiteln gruppieren
42  tmp_norden <- icd_kapitel(norden,ambu_icd)
43  norden_icd <- icd_sum_kapitel(tmp_norden,4,icd_alle)
44  # Prozentuale Spalte hinzufügen
45  norden_icd <- cbind(norden_icd,round(x=norden_icd[,1]*100/sum(norden_icd[,1])
        ,digits=2))
46  # Überschriften setzen
47  rownames(norden_icd) <- icd_alle
48  colnames(norden_icd) <- c("anzahl","rel_h")
49  # Osten nach den ICD-Kapiteln gruppieren
50  tmp_osten <- icd_kapitel(osten,ambu_icd)
51  osten_icd <- icd_sum_kapitel(tmp_osten,4,icd_alle)
52  # Prozentuale Spalte hinzufügen
53  osten_icd <- cbind(osten_icd,round(x=osten_icd[,1]*100/sum(osten_icd[,1]),
        digits=2))
54  # Überschriften setzen
55  rownames(osten_icd) <- icd_alle
56  colnames(osten_icd) <- c("anzahl","rel_h")
57  # Süden nach den ICD-Kapiteln gruppieren
58  tmp_sueden <- icd_kapitel(sueden,ambu_icd)
59  sueden_icd <- icd_sum_kapitel(tmp_sueden,4,icd_alle)
60  # Prozentuale Spalte hinzufügen
61  sueden_icd <- cbind(sueden_icd,round(x=sueden_icd[,1]*100/sum(sueden_icd[,1])
        ,digits=2))
62  # Überschriften setzen
63  rownames(sueden_icd) <- icd_alle
64  colnames(sueden_icd) <- c("anzahl","rel_h")
65  # Westen nach den ICD-Kapiteln gruppieren
66  tmp_westen <- icd_kapitel(westen,ambu_icd)
67  westen_icd <- icd_sum_kapitel(tmp_westen,4,icd_alle)
68  # Prozentuale Spalte hinzufügen
69  westen_icd <- cbind(westen_icd,round(x=westen_icd[,1]*100/sum(westen_icd[,1])
        ,digits=2))
70  # Überschriften setzen
71  rownames(westen_icd) <- icd_alle
72  colnames(westen_icd) <- c("anzahl","rel_h")
73  # Mitte nach den ICD-Kapiteln gruppieren
```

```
74  tmp_mitte <- icd_kapitel(mitte,ambu_icd)
75  mitte_icd <- icd_sum_kapitel(tmp_mitte,4,icd_alle)
76  # Prozentuale Spalte hinzufügen
77  mitte_icd <- cbind(mitte_icd,round(x=mitte_icd[,1]*100/sum(mitte_icd[,1]),
        digits=2))
78  # Überschriften setzen
79  rownames(mitte_icd) <- icd_alle
80  colnames(mitte_icd) <- c("anzahl","rel_h")
81  # Ergebnismatrix erstellen (Prozentwerte aller 5 Gebiete zusammenfügen)
82  alle_regionen <- cbind(norden_icd[,2],osten_icd[,2],sueden_icd[,2],westen_icd
        [,2],mitte_icd[,2])
83  # Überschriften setzen
84  rownames(alle_regionen) <- icd_alle
85  colnames(alle_regionen) <- c("proz_norden","proz_osten","proz_sueden","proz_
        westen","proz_mitte")
86  ## PLOT BEGINNT ###
87  # Seitenabstand
88  par(mar=c(7,4,7,3))
89  # Abstände der Balken
90  space1 <- rep(x=3,times=22)
91  space2 <- c(3.5,rep(x=3,times=21))
92  space3 <- c(4.0,rep(x=3,times=21))
93  space4 <- c(4.5,rep(x=3,times=21))
94  space5 <- c(5.0,rep(x=3,times=21))
95  # Plotten
96  balken <- barplot(height=alle_regionen[,1],col=farben[1],space=space1,border=
        "black",axes=FALSE,axisnames=FALSE,main="ICD-10 Verteilung in Sachsen-
        Anhalt",ylim=c(0,1.1*max(alle_regionen)))
97  balken <- barplot(height=alle_regionen[,2],col=farben[2],space=space2,border=
        "black",add=TRUE,axes=FALSE,axisnames=FALSE)
98  balken <- barplot(height=alle_regionen[,3],col=farben[3],space=space3,border=
        "black",add=TRUE,axes=FALSE,axisnames=FALSE)
99  balken <- barplot(height=alle_regionen[,4],col=farben[4],space=space4,border=
        "black",add=TRUE,axes=FALSE,axisnames=FALSE)
100 balken <- barplot(height=alle_regionen[,5],col=farben[5],space=space5,border=
        "black",add=TRUE,axes=FALSE,axisnames=FALSE)
101 # Legende anzeigen
102 legend(x=85,y=18.5,legend=c("Norden","Osten","Süden","Westen","Mitte"),fill=
        farben,cex=0.8,xpd=TRUE)
103 # X-Achse anzeigen
104 text(x=balken,y=-0.5,srt=35,adj=1,xpd=TRUE,labels=rownames(alle_regionen),cex
        =0.9)
105 # X-Achsen-Beschriftung
106 mtext(text="ICD Codes",side=1,line=4.5)
```

```
107  # Y-Achse Markierung
108  ybereich <- pretty(x=c(0,max(alle_regionen)),n=15)
109  # Y-Achse anzeigen
110  axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos
         =1.5)
111  # Y-Achsen-Beschriftung
112  mtext(text="Prozent",side=2,line=1.25)
```

Um die Landkreise/kreisfreien Städte in die fünf Gebiete zu gruppieren, werden in Zeile 8-12 durch den `rbind()`-Aufruf die entsprechenden Regionen zusammengefasst. In Zeile 14 wird ein String-Vektor namens `icd_alle` definiert, der alle Diagnoseschlüsselbereiche der 22 Kapitel enthält. Ab Zeile 16 bis einschließlich Zeile 27, ist eine Funktion zur Reduzierung der fünfstelligen ICD-10-Codes aus den ambulanten Daten in das dreistellige Format der Kapitel definiert. Der Funktion wird durch das Argument `landkreis=` eine Region (z. B. der `norden`) und durch das Argument `ambu_icd=` die (dazugehörigen) Diagnoseschlüssel übergeben. In Zeile 18 wird innerhalb der Funktion der `merge()`-Befehl für die Verbindung der Pseudonyme aus den Stammdaten mit dessen Diagnoseschlüsseln aus den ambulanten Daten aufgerufen. Anschließend werden alle ICD-10-Codes mit Hilfe der `substr()`-Funktion auf drei Zeichen gekürzt, um sie einem Kapitel zuzuordnen. Der nachfolgende `unique()`-Befehl in Zeile 22 stellt sicher, dass die Kombination ICD-10-Code und Pseudonym nicht mehrfach im Rückgabewert `tmp_landkreis` vorkommt. Es wird eine weitere Funktion (`icd_sum_kapitel()`) benötigt, die alle Diagnoseschlüssel eines Kapitels summiert (Zeile 29-40). Dazu wird in Zeile 34 die Funktion `func_bereich()` aufgerufen, die die ICD-10-Codes eines Gebietes entsprechend der Kapitel gruppiert. Am Ende der Funktion wird ein einspaltiger *data.frame* `rueckgabe` mit 22 Werten zurückgegeben, die der Gesamtanzahl an Diagnoseschlüsseln pro Kapitel entsprechen.
Nachfolgend werden für jedes der fünf Gebiete die Funktionen angewendet und mit dem `cbind()`-Befehl die prozentuale Häufigkeit als neue Spalte für jedes Kapitel hinzugefügt (siehe Zeile 42-45 für den *Norden*).
In Zeile 82 werden vorbereitend für die Grafikausgabe alle berechneten Prozentwerte der fünf Gebiete in der Ergebnismatrix `alle_regionen` gespeichert und beschriftet. Ab der Zeile 86 beginnen die Befehle für den Plot. Für die Überlappung der Balken eines Kapitels sind fünf `space`-Vektoren definiert, die die Positionen der Balken auf der X-Achse festlegen (Zeile 90-94).

Nachstehend sind fünf `barplot()`-Aufrufe zu finden, mit unterschiedlichen Parametern für die Balkenhöhe (`height=`), die Farbe (`col=`) und den Abstand (`space=`). Mit dem Befehl `add=` und dem Wert `TRUE` wird erreicht, dass die fünf Ausgaben überlappend dargestellt werden. Ab der Zeile 102 werden die bereits erläuterten Funktionen `legend()`, `text()`, `mtext()` und `axis()` für die Visualisierung verwendet.

5.5.3. Prozentuale Verteilung einzelner ICD-10-Codes in Sachsen-Anhalt

Für die nächste Untersuchung wird zur Identifizierung von Unregelmäßigkeiten in der Verteilung von Krankheitsbildern die Standardabweichung als mathematisches Werkzeug eingesetzt. Im Gegensatz zum Kapitel 5.5.2 erfolgt die Auswertung nicht auf Grundlage einer Clusterung, sondern für jeden einzelnen Diagnoseschlüssel auf Basis der elf Landkreise und drei kreisfreien Städte.
Die Standardabweichung gibt für jeden ICD-10-Code die sog. Streuung an, um die die tatsächlichen Häufigkeiten der Regionen im Mittel variieren. Für die Berechnung werden die relativen Häufigkeiten umgekehrt proportional verwendet, sodass die Ergebnisse in der Form *1 Fall auf `X` Versicherte* vorliegen. Dadurch wird das Vorkommen unabhängig von der variierenden Versichertenanzahl jedes Landkreises beschrieben, sodass die Werte untereinander vergleichbar sind. Voraussetzung dieser Betrachtung ist, dass nur ICD-10-Codes verwendet werden können, die in jeder Region mindestens einmal vorkommen.
Ein Auszug der in `R` erstellten Häufigkeiten ist in der Tabelle 5.4 zu finden. Ein großer Zahlenwert der Standardabweichung drückt aus, dass der betreffende Diagnoseschlüssel in Sachsen-Anhalt zwischen den Landkreisen/kreisfreien Städten inhomogen verteilt ist. Entsprechend deutet ein niedriger Wert auf eine annähernde Gleichverteilung der Krankheit hin.

ICD-Code	Magdeburg	Halle	Dessau-Roßlau	Altmark.-Salzw.	Stendal	Börde	Jerichower Land	Harz	Salzlandkreis	Anhalt-Bitterfeld	Wittenberg	Mansfeld Südharz	Saalekreis	Burgenlandkreis	St.-Abw.
R8290	5812	3640	1436	973	2665	757	1532	4672	1572	4320	2121	442	264	2109	1639
M4724	5812	775	769	901	3198	4064	1641	5096	2480	3168	1316	2303	656	1745	1596
H5450	484	3943	175	640	1453	2484	534	5606	400	1485	954	3141	536	568	1561
J9611	4844	2253	378	1352	533	2794	534	757	786	2795	587	1016	3173	4218	1433
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
I2590	5.88	6.31	4.59	6.61	6.27	5.63	7.78	5.40	5.50	5.03	6.13	6.10	5.35	5.87	0.74
E1190	4.80	4.98	4.29	5.04	5.10	4.49	5.00	4.46	4.71	4.72	4.59	4.76	4.53	4.51	0.24
Z2510	2.76	3.12	2.64	2.50	2.55	2.68	2.52	2.84	3.01	2.78	2.68	3.16	2.89	2.60	0.21
I1090	2.22	2.05	1.78	2.19	2.16	2.04	2.09	1.77	2.12	1.87	2.33	2.15	1.93	1.82	0.17

Tab. 5.4.: 1 Fall auf X Versicherte

Die Tabelle 5.4 listet einen Auszug der insgesamt 1.770 gemeinsamen ICD-10-Codes auf. Sie verzeichnet für jeden Landkreis/kreisfreie Stadt die Beziehung *1 Fall auf `X` Versicherte*, wobei das `X` den Zahlenwerten in der Tabelle entspricht. In der ersten Spalte stehen die entsprechenden ICD-10-Codes und in der letzten Spalte (*St.-Abw.*) ist deren Standardabweichung berechnet, nach der die Tabelle absteigend sortiert ist.
Damit eine statistisch vertretbare Aussage durch die Tabelle getroffen werden kann, enthält diese nur Diagnoseschlüssel, die mindestens zehnmal in jeder Region existieren. Dies ist erforderlich, um den Einfluss von sog. Ausreißern einzudämmen, die das Resultat verzerren würden. Folglich werden für die Auswertungen im Minimum stets 140 Versicherte pro Diagnoseschlüssel betrachtet. Aufgrund dieser Vorgabe verbleiben nur ca. 16% aller im Datenbestand existierenden ICD-10-Codes (1.770 von insgesamt 10.994), die hierfür herangezogen werden.
Der Code mit der größten berechneten Standardabweichung ist `R8290` (*Sonstige und nicht näher bezeichnete abnorme Urinbefunde*), in dem u.a. im Saalekreis ein Fall auf 264 Versicherte und in Magdeburg ein Fall auf 5.812 Versicherte vorkommt, gefolgt vom Code `M4724` (*Sonstige Spondylose mit Radikulopathie: Thorakalbereich*), der ebenfalls im Saalekreis am häufigsten und in Magdeburg am wenigsten vertreten ist.
Die Krankheit mit der geringsten Standardabweichung hat den Code `I1090` (*Essentielle Hypertonie, nicht näher bezeichnet: Ohne Angabe einer hypertensiven Krise*), an der in allen Regionen fast jeder zweite Versicherte leidet.

Die nächste Abbildung zeigt eine grobe Gesamtübersicht der absteigend sortierten ICD-10-Codes der Tabelle 5.4. Durch die grafische Visualisierung kann ein Verhältnis aller Streuungsmaße abgelesen werden.

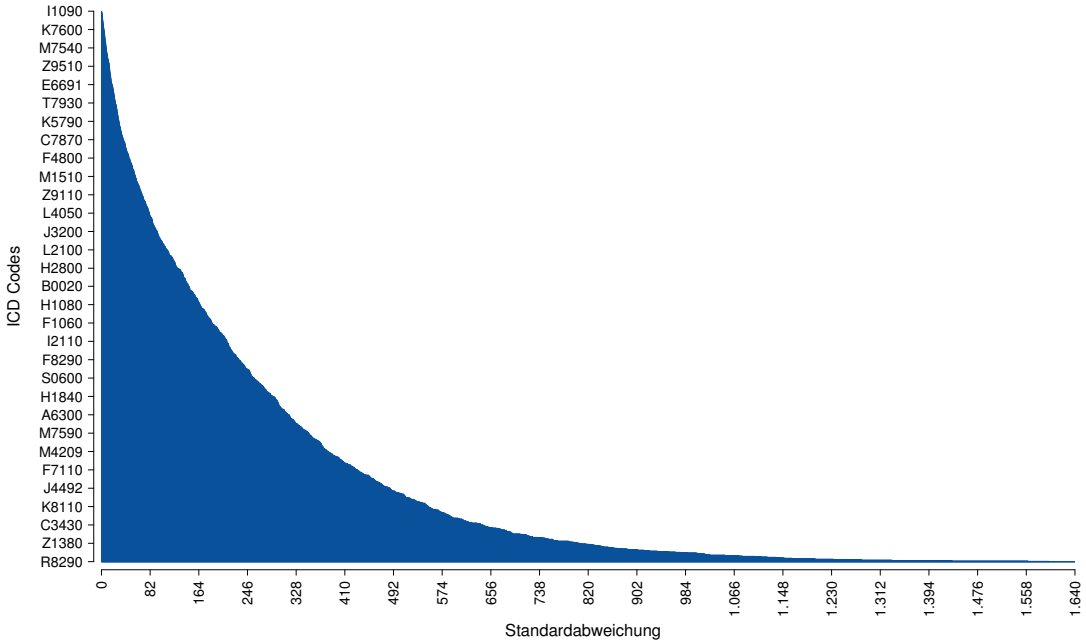

Abb. 5.12.: ICD-10-Standardabweichung

Die Grafik lässt erkennen, dass nur wenige Diagnoseschlüssel eine hohe Standardabweichung und eine verhältnismäßig geringe Anzahl eine sehr kleine Standardabweichung aufweisen. Der Großteil befindet sich erwartungsgemäß im Mittelfeld der Grafik und lässt demnach auf eine moderate Gleichverteilung schließen.

Die Grundlage für die nachfolgende Abbildung ist ebenfalls Tabelle 5.4 und basiert ausschließlich auf Diagnoseschlüsseln mit den höchsten Streuungen. Die Grafik setzt alle Regionen in Relation zueinander, indem die Häufigkeiten eines Diagnoseschlüssels prozentual dargestellt werden. Zu einem späteren Zeitpunkt werden in einer Analyse auch Schlüssel mit minimalen Streuungen untersucht.

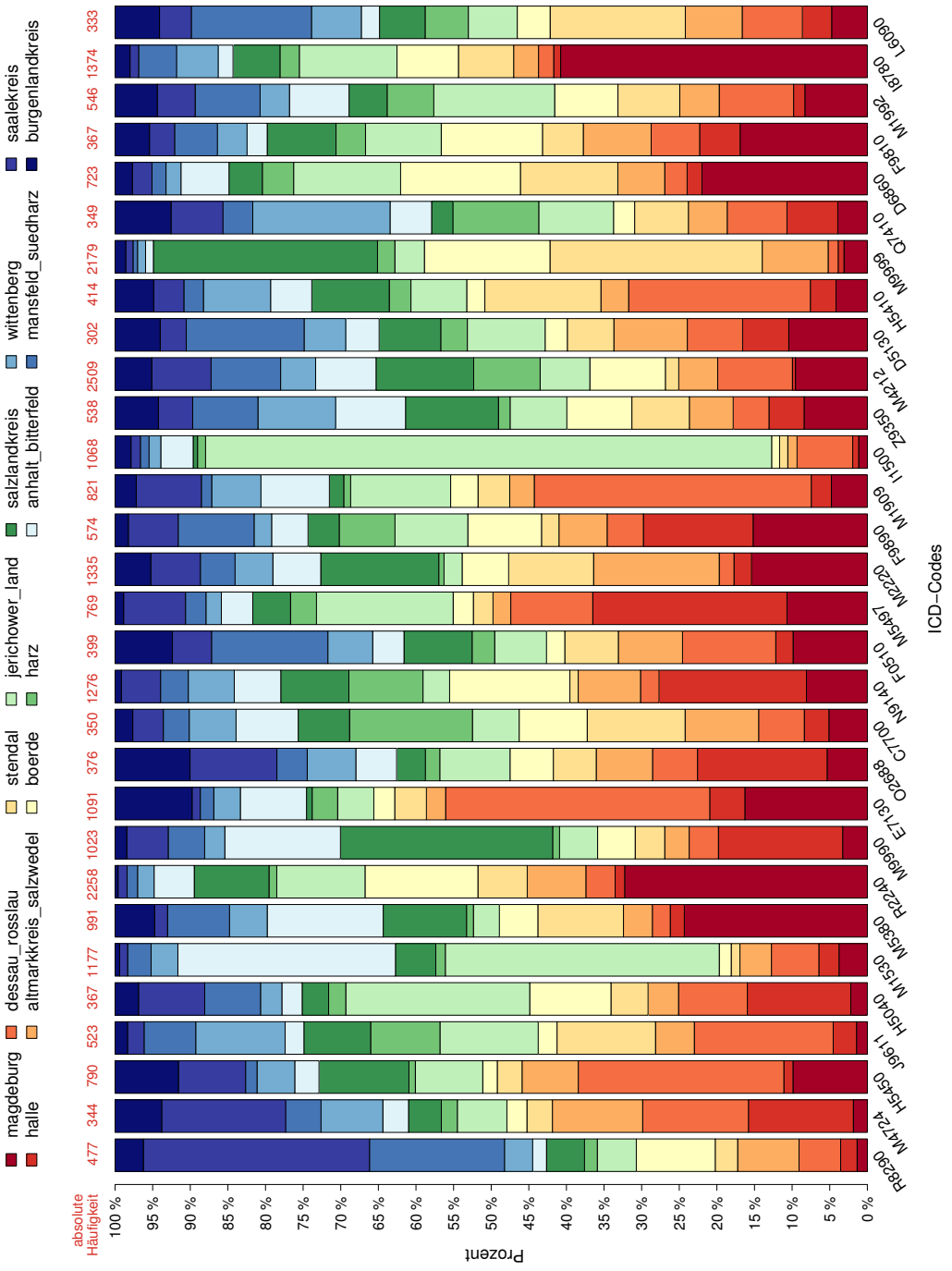

Abb. 5.13.: ICD-10-Codes mit der größten Standardabweichung

Das Diagramm in Abbildung 5.13 fasst die Landkreise/kreisfreien Städte in 14-fach gestapelten Säulen zusammen, wobei die X-Achse die ersten 30 ICD-10-Codes mit der größten Standardabweichung in absteigender Reihenfolge darstellt. Die Y-Achse zeigt die prozentualen Anteile der Regionen und am Kopfende jeder Säule steht die Gesamtanzahl des entsprechenden ICD-10-Codes.
Die Grafik zeigt, dass bei mehreren Diagnoseschlüsseln ein großer Prozentanteil des absoluten Vorkommens auf ein oder zwei Regionen entfallen. Ein Beispiel ist der ICD-10-Code `M1530` (*Sekundäre multiple Arthrose*), der verstärkt im Jerichower Land mit einem Anteil von 36,4% und in Anhalt-Bitterfeld mit 28,9% vorkommt. Betrachtet man die Gesamtanzahl aller Fälle in Sachsen-Anhalt von 1.177, so deuten beide prozentualen Anteile auf regionale Häufung hin. Eine noch stärkere regionale Konzentration eines Krankheitsbildes hat im Jerichower Land der Diagnoseschlüssel `I1500` (*Renovaskuläre Hypertonie: Ohne Angabe einer hypertensiven Krise*). Von den 1068 erkrankten Versicherten, leben dort 694 (entspricht 1 Fall auf 33 Versicherte) und haben somit einen Anteil von 75,2% am Gesamtaufkommen.
In Magdeburg sind vermehrt die Diagnoseschlüssel `R2240` (*Lokalisierte Schwellung, Raumforderung und Knoten der Haut und der Unterhaut an den unteren Extremitäten*) und `I8780` (*Sonstige näher bezeichnete Venenkrankheiten*) mit knapp 33% bzw. 40% vorzufinden.

Basierend auf der prozentualen Betrachtung wird der Quelltext leicht modifiziert, sodass nicht nur Diagnoseschlüssel mit der größten Standardabweichung betrachtet werden, sondern auch ICD-10-Codes in selbstdefinierten Bereichen. Beispielsweise kann es von Interesse sein, den Bereich von `I00.00` bis `I99.00` oder auch einen bestimmten Abschnitt, der für diverse Probleme des Knochengerüsts (`M00.00` bis `M99.00`) zuständig ist, genauer zu untersuchen.
Die nächsten Abbildungen zeigen exemplarisch prozentuale Werte für die Bereiche `D18.00-D39.10`, `I13.00-I25.20`, `J11.10-J35.00` sowie `M79.00-M81.00`.

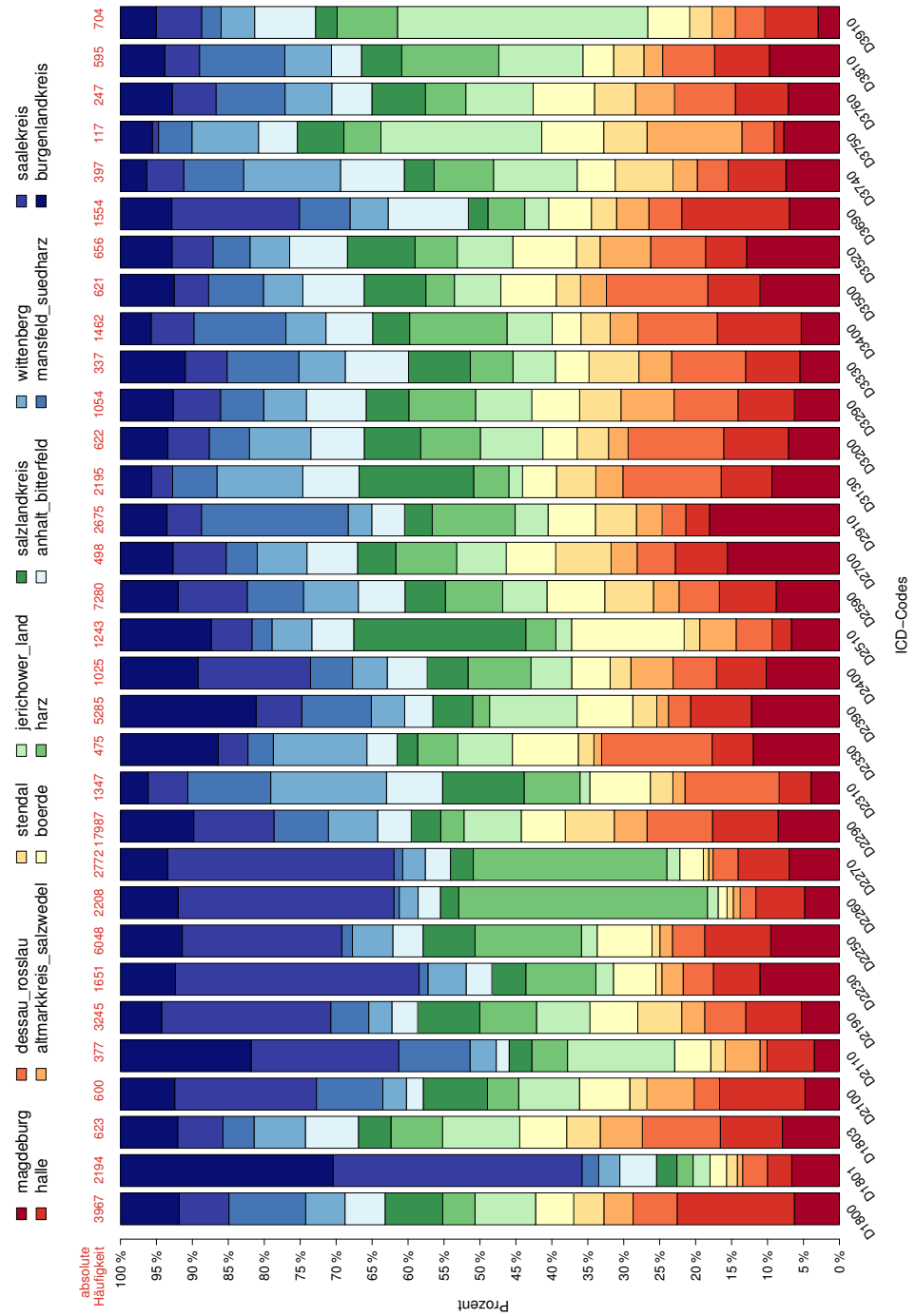

Abb. 5.14.: Prozentuale Verteilung des ICD-10-Bereiches D1800-D3910

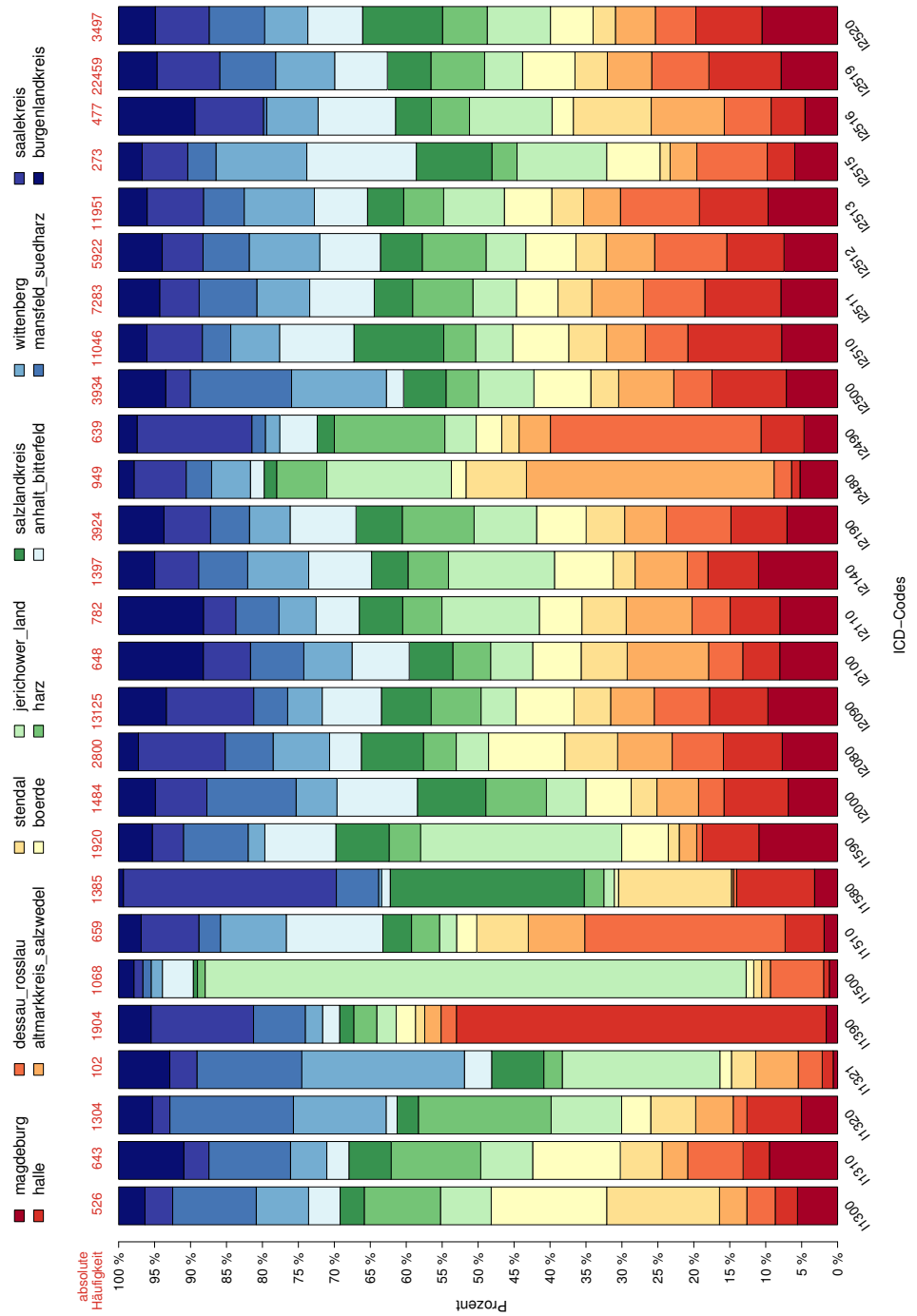

Abb. 5.15.: Prozentuale Verteilung des ICD-10-Bereiches I1300-I2520

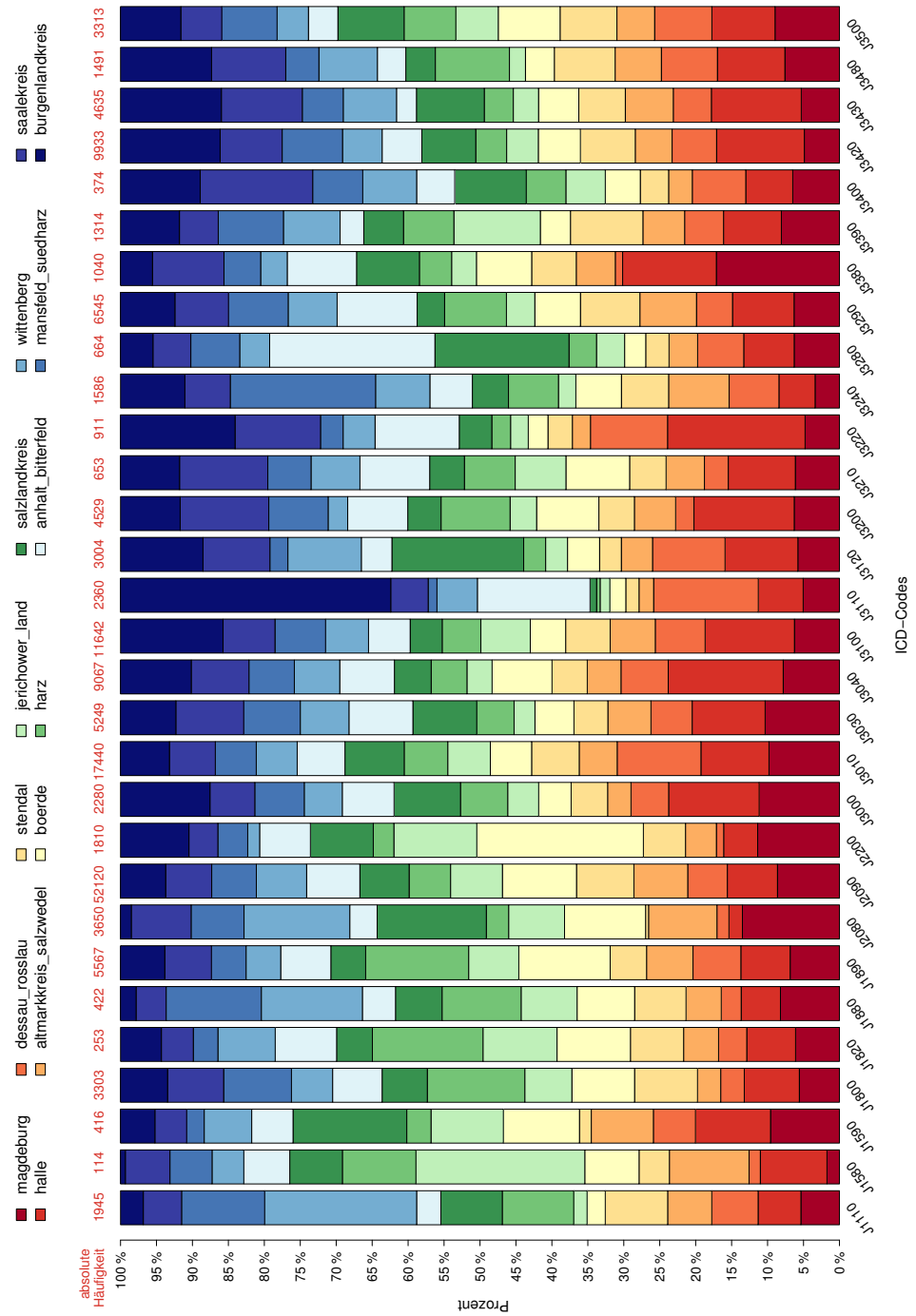

Abb. 5.16.: Prozentuale Verteilung des ICD-10-Bereiches J1110-J3500

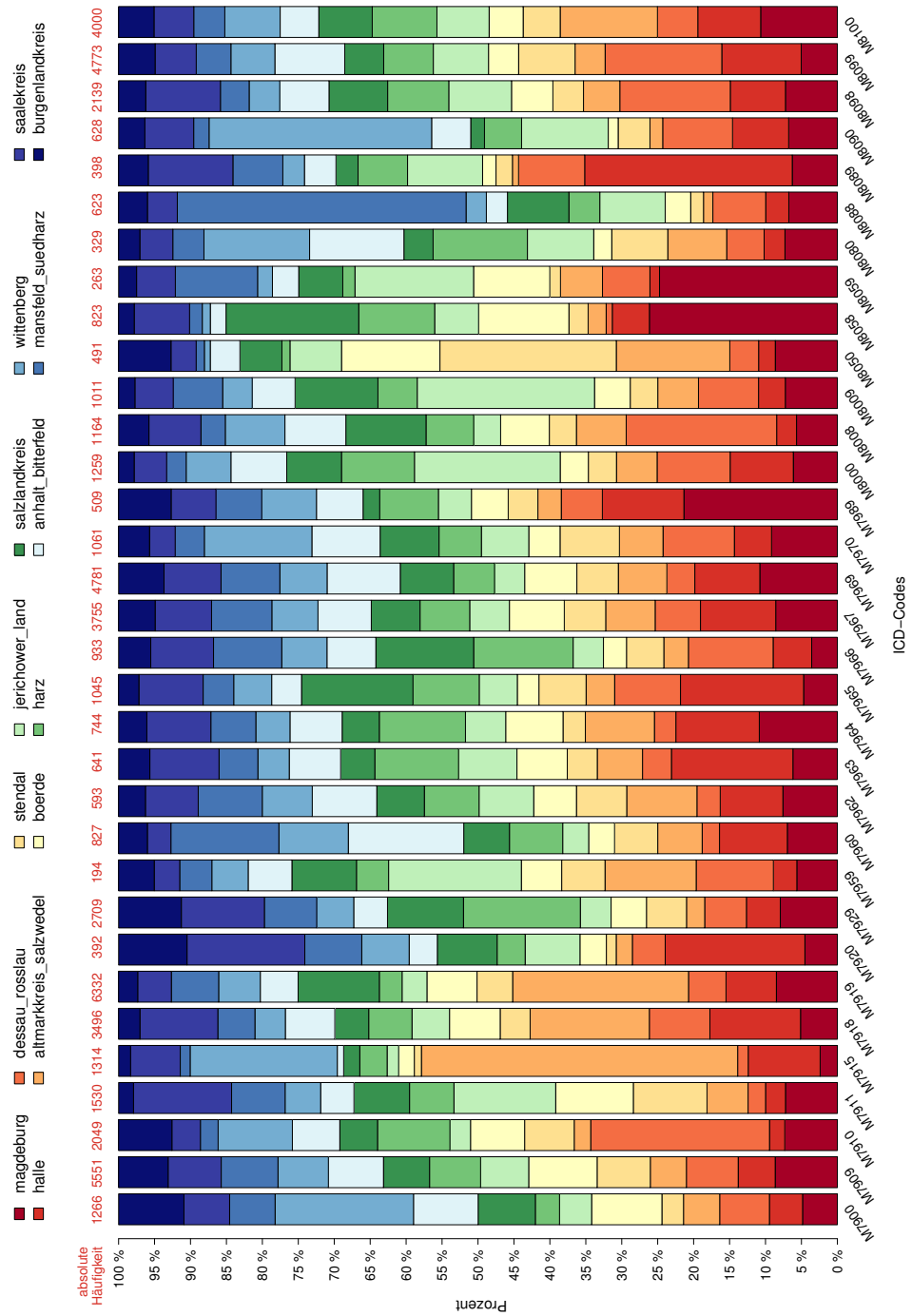

Abb. 5.17.: Prozentuale Verteilung des ICD-10-Bereiches M7900-M8100

Bestimmte Teile des Quellcodes der Tabelle 5.4 sowie der Abbildungen 5.13 und 5.14 bis 5.17 liegen der gleichen Basis zu Grunde, sodass der erste Abschnitt Quelltext enthält, der für alle drei Abbildungen/Tabellen identisch ist.

```
# Variable setzen
  # Mindestanzahl von ICD's pro einzelnem Landkreis
  # Default: 10
  mindestens <- 10
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]
# Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine
    Häufigkeitstabelle auf Basis der ICD-Spalte erstellen
magdeburg_icd <- data.frame(table("ICD"=subset(unique(merge(magdeburg,ambu_
    icd,by="PSEUDONYM")),select=c(ICD))))
{ ... }
# Filtern der ICD-Codes die mindestens die gesetzte Anzahl pro LK haben
if (mindestens != 1) {
  magdeburg_icd <- subset(x=magdeburg_icd,subset=(magdeburg_icd$Freq >=
      mindestens))
  { ... }
}
# Liste erstellen mit allen ICDs der Landkreise
alle_kreise_icd <- list (magdeburg_icd,halle_icd,dessau_rosslau_icd,
    altmarkkreis_salzwedel_icd,stendal_icd,boerde_icd,jerichower_land_icd,
    harz_icd,salzlandkreis_icd,anhalt_bitterfeld_icd,wittenberg_icd,mansfeld_
    suedharz_icd,saalekreis_icd,burgenlandkreis_icd)
# Reduce ist eine Funktion, die merge auf alle Elemente der Liste anwendet
gemeinsame_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_
    kreise_icd)
# ICD's entfernen, die nicht in allen LK mindestens X-mal vorkommen
magdeburg_gemeinsame_icd <- magdeburg_icd[magdeburg_icd$ICD %in% gemeinsame_
    icd$ICD,]
{ ... }
# Die Spalte "pro_patient" jedem LK hinzufügen
# "pro_patient" beschreibt: 1 Fall auf X Versicherte
magdeburg_gemeinsame_icd <- cbind(magdeburg_gemeinsame_icd,"pro_patient"=
    length(magdeburg$PSEUDONYM) / magdeburg_gemeinsame_icd$Freq)
{ ... }
```

```
32 # Ergebnismatrix erzeugen (1 Fall auf X Versicherte)
33 matrix_pro_patient <- cbind(
34   magdeburg_gemeinsame_icd$pro_patient,
35   halle_gemeinsame_icd$pro_patient,
36   dessau_rosslau_gemeinsame_icd$pro_patient,
37   altmarkkreis_salzwedel_gemeinsame_icd$pro_patient,
38   stendal_gemeinsame_icd$pro_patient,
39   boerde_gemeinsame_icd$pro_patient,
40   jerichower_land_gemeinsame_icd$pro_patient,
41   harz_gemeinsame_icd$pro_patient,
42   salzlandkreis_gemeinsame_icd$pro_patient,
43   anhalt_bitterfeld_gemeinsame_icd$pro_patient,
44   wittenberg_gemeinsame_icd$pro_patient,
45   mansfeld_suedharz_gemeinsame_icd$pro_patient,
46   saalekreis_gemeinsame_icd$pro_patient,
47   burgenlandkreis_gemeinsame_icd$pro_patient
48 )
49 # Überschriften setzen
50 rownames(matrix_pro_patient) <- magdeburg_gemeinsame_icd$ICD
51 # Die Standardabweichung pro Zeile für jeden LK berechnen und als Spalte
       hinzufügen
52 matrix_pro_patient <- cbind(matrix_pro_patient,apply(X=matrix_pro_patient,
       MARGIN=1,FUN=sd2))
53 colnames(matrix_pro_patient) <- c(landkreise,"st_abw")
54 # Die Ergebnismatrix absteigend der Standardabweichung sortieren
55 matrix_pro_patient <- matrix_pro_patient[order(-matrix_pro_patient[,15]),]
```

In Zeile 4 wird die globale Variable `mindestens` mit der Mindestanzahl von ICD-10-Codes pro Landkreis/kreisfreien Stadt gesetzt. Nach dem Einlesen der Stamm- und ambulanten Daten werden nachfolgend die Regionen mit den ICD-10-Codes der Pseudonyme verknüpft (`merge()`) und gezählt (`table()` mit `select=`). Dies ist im Quelltext für Magdeburg in Zeile 12-13 exemplarisch aufgelistet. Abhängig von der gesetzten Mindestanzahl in Zeile 4 werden in den Zeilen 15-18 die *data.frames* der Regionen entsprechend gefiltert, d.h. in jedem einzelnen *data.frame* befinden sich nur die regional vorkommenden Codes mit einer Mindestanzahl von zehn. Dadurch besitzen alle eine unterschiedliche Anzahl von Zeilen und können zur Zeit nicht miteinander verbunden werden. Anschließend wird in Zeile 20 eine Liste der erstellten Häufigkeitstabellen jeder Region erzeugt, die für die `Reduce()`-Funktion in Zeile 22 benötigt wird. Die Funktion nimmt immer zwei Elemente aus der Liste `alle_kreise_icd` und verbindet

diese über die Spalte *ICD* mit Hilfe des `merge()`-Aufrufs und dem Argument `by=`. Das Ergebnis ist ein *data.frame*, in welchem nur die Diagnoseschlüssel enthalten sind, die in allen Regionen jeweils mindestens zehn Mal vorkommen. In der folgenden Zeile 24 wird dieser genutzt, um alle vierzehn Regionen abzugleichen, indem nicht-gemeinsame ICD-10-Codes entfernt werden, sodass alle eine identische Länge aufweisen.
Die Zeile 28 fügt eine zusätzliche Spalte hinzu, in der die Gesamtanzahl der Versicherten pro Landkreis/kreisfreien Stadt durch die entsprechende Häufigkeit jedes ICD-10-Codes dividiert wird (1 Fall auf X Versicherte). Diese Ergebnisse werden in der Matrix `matrix_pro_patient` zusammengefügt (Zeile 33-48). Der letzte wesentliche Schritt ist in Zeile 52, der die unkorrigierte Standardabweichung zeilenweise berechnet und als neue Spalte durch `cbind()` hinzufügt. Dazu wird die in der `initial.R` befindliche Funktion `sd2()` herangezogen und dem Argument `FUN=` der Funktion `apply()` übergeben. Die Funktion `apply()` liefert ein Objekt zurück, dass eine Funktion zeilenweise (`MARGIN=1`) oder spaltenweise auf ein übergebenes Objekt (`X=`) anwendet. In Zeile 55 wird die Ergebnismatrix absteigend nach der Standardabweichung durch den `order()`-Befehl sortiert.

Im Folgenden ist der restliche Quelltext beschrieben, der für die Tabelle 5.4 benötigt wird. Im Quelltext-Anhang auf Seite 204 ist der chronologisch vollständige Code nachzulesen.

```
1 ### Tabellen als CSV auf Festplatte speichern (1Fall auf X Versicherte)
2 write.table(x=cbind("ICD-Code"=rownames(matrix_pro_patient),matrix_pro_
      patient),file="CSV/ICD-1Fall_auf_X_Versicherte_pro_lk.csv",sep=";",row.
      names=FALSE,col.names=TRUE,fileEncoding="latin1")
3 # matrix_pro_patient absteigend nach den ICD's sortieren und Zahlen runden
4 matrix_pro_patient_sort <- round(matrix_pro_patient[order(rownames(matrix_pro
      _patient)),],digits=2)
5 write.table(x=cbind("ICD-Code"=rownames(matrix_pro_patient_sort),matrix_pro_
      patient_sort),file="CSV/ICD-1Fall_auf_X_Versicherte_pro_lk_nach_ICD_
      sortiert.csv",sep=";",row.names=FALSE,col.names=TRUE,fileEncoding="latin1
      ")
```

Die Zeilen 2-5 schreiben die Ergebnismatrix `matrix_pro_patient` als CSV-Datei auf die Festplatte. Während in Zeile 2 die Sortierung der Matrix nach der Standardabweichung erfolgt, wird in Zeile 5 eine Sortierung nach den Diagnoseschlüsseln vorgenommen.

Dieser Quellcode-Abschnitt beinhaltet Befehle, die ausschließlich für die Abbildung 5.13 nötig sind. Auf Seite 208 ist der komplette Quelltext zu finden.

```
# Variablen setzen
  # Anzahl von ICD-Codes im Balkendiagramm
  # Default: 30
  anzahl_icd <- 30
# Ergebnismatrix erzeugen (absolute Häufigkeit)
matrix_abs_h <- cbind(
  magdeburg_gemeinsame_icd$Freq,
  halle_gemeinsame_icd$Freq,
  dessau_rosslau_gemeinsame_icd$Freq,
  altmarkkreis_salzwedel_gemeinsame_icd$Freq,
  stendal_gemeinsame_icd$Freq,
  boerde_gemeinsame_icd$Freq,
  jerichower_land_gemeinsame_icd$Freq,
  harz_gemeinsame_icd$Freq,
  salzlandkreis_gemeinsame_icd$Freq,
  anhalt_bitterfeld_gemeinsame_icd$Freq,
  wittenberg_gemeinsame_icd$Freq,
  mansfeld_suedharz_gemeinsame_icd$Freq,
  saalekreis_gemeinsame_icd$Freq,
  burgenlandkreis_gemeinsame_icd$Freq
)
# Überschriften setzen
rownames(matrix_abs_h) <- magdeburg_gemeinsame_icd$ICD
# Aufsummierte Häufigkeit als Spalte hinzufügen
matrix_abs_h <- cbind(matrix_abs_h,rowSums(matrix_abs_h))
colnames(matrix_abs_h) <- c(landkreise,"Freq_ges")
# ICD-Codes mit der größten Standardabweichnung werden ausgewählt und
    prozentual dargestellt
icd_plot <- NULL
for (i in 1:anzahl_icd) {icd_plot <- rbind(icd_plot,sum(matrix_pro_patient[i
    ,1:14]) / matrix_pro_patient[i,1:14])}
rownames(icd_plot) <- rownames(matrix_pro_patient[1:anzahl_icd,])
# "icd_plot" transponieren um ICDs als Spalten zu erhalten
icd_plot <- t(icd_plot)
# Prozentuale Tabelle erstellen
icd_plot_proz <- prop.table(x=icd_plot,margin=2)*100
### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(5,5,7,10))
# Plotten
icd_st_abw_plot <- barplot(height=icd_plot_proz,main=paste(anzahl_icd,"ICD-
```

```
      Codes mit der größten Standardabweichung"),col=farben_2,axes=FALSE,
      axisnames=FALSE)
40  # Legende anzeigen
41  legend(x=37.2,y=100,legend=landkreise,fill=farben_2,cex=0.9,xpd=TRUE)
42  # X-Achse anzeigen
43  text(x=icd_st_abw_plot+0.1,y=-1.8,srt=35,adj=1,xpd=TRUE,labels=colnames(icd_
      plot),cex=0.65)
44  # X-Achsen-Beschriftung
45  mtext(text="ICD-Codes",side=1,line=3)
46  # Y-Achse Markierung
47  ybereich <- pretty(x=c(0,100),n=20)
48  # Y-Achse anzeigen
49  axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
      =-0.5)
50  # Y-Achsen-Beschriftung
51  mtext(text="Prozent",side=2,line=3)
52  # Absolute Häufigkeit des ICD-Codes bestimmen und anzeigen
53  icd_plot_Freq <- NULL
54  for (i in 1:anzahl_icd) {
55    tmp <- subset(x=matrix_abs_h[,15],subset=(colnames(icd_plot_proz)[i] ==
        rownames(matrix_abs_h)))
56    icd_plot_Freq <- cbind(icd_plot_Freq,tmp[[1]])
57  }
58  text(x=icd_st_abw_plot+0.3,y=103,adj=1,xpd=TRUE,labels=icd_plot_Freq,cex
      =0.65,col=farben_2[2])
59  text(x=-0.5,y=103,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.75,col=
      farben_2[2])
60  ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit)
61  write.table(x=cbind("ICD-Code"=rownames(matrix_abs_h),matrix_abs_h),file="CSV
      /ICD10-absolute_Haeufigkeit_pro_lk.csv",sep=";",row.names=FALSE,col.names
      =TRUE,fileEncoding="latin1")
```

Mit der definierten Variablen `anzahl_icd` in Zeile 4 wird die Anzahl der darzustellenden Säulen im späteren `barplot()`-Aufruf bestimmt. Für die Anzeige der absoluten Häufigkeiten über jedem Balken wird ab Zeile 6 bis 21 eine Ergebnismatrix (`matrix_abs_h`) definiert und anschließend beschriftet (Zeile 23 bzw. 26). Nachfolgend werden für die Zahlenwerte der Matrix `matrix_pro_patient` die prozentualen Häufigkeiten für jeden Diagnoseschlüssel pro Region ermittelt (siehe Zeile 28-34). Ab der Zeile 37 beginnen die Befehlsaufrufe für die Plotausgabe. In der letzten Zeile werden die absoluten Häufigkeiten (`matrix_abs_h`) als CSV-Datei auf die Festplatte gesichert.

Der letzte Abschnitt mit Quellcode ist für die Darstellungen der prozentualen Anteile der selbstdefinierbaren ICD-10-Bereiche (Abbildungen 5.14 - 5.17) erforderlich. Der lückenlose Quelltext befindet sich im Anhang auf der Seite 214.

```
# Variablen setzen
  # ICD-Bereich auswählen, der dargestellt wird
  icd_von <- "M5190"
  icd_bis <- "M5500"
# Ergebnismatrix erzeugen (absolute Häufigkeit)
matrix_abs_h <- cbind(
  magdeburg_gemeinsame_icd$Freq,
  halle_gemeinsame_icd$Freq,
  dessau_rosslau_gemeinsame_icd$Freq,
  altmarkkreis_salzwedel_gemeinsame_icd$Freq,
  stendal_gemeinsame_icd$Freq,
  boerde_gemeinsame_icd$Freq,
  jerichower_land_gemeinsame_icd$Freq,
  harz_gemeinsame_icd$Freq,
  salzlandkreis_gemeinsame_icd$Freq,
  anhalt_bitterfeld_gemeinsame_icd$Freq,
  wittenberg_gemeinsame_icd$Freq,
  mansfeld_suedharz_gemeinsame_icd$Freq,
  saalekreis_gemeinsame_icd$Freq,
  burgenlandkreis_gemeinsame_icd$Freq
)
# Überschriften setzen
rownames(matrix_abs_h) <- magdeburg_gemeinsame_icd$ICD
# Aufsummierte Häufigkeit als Spalte hinzufügen
matrix_abs_h <- cbind(matrix_abs_h,rowSums(matrix_abs_h))
colnames(matrix_abs_h) <- c(landkreise,"Freq_ges")
### ICD Bereich darstellen
# Matrix absteigend sortieren für den "von bis" Bereich
matrix_pro_patient_rowname_sort <- matrix_pro_patient[order(rownames(matrix_
    pro_patient)),]
icd_bereich <- subset(x=matrix_pro_patient_rowname_sort,subset=(rownames(
    matrix_pro_patient_rowname_sort) >= icd_von & rownames(matrix_pro_patient
    _rowname_sort) <= icd_bis))[,1:14]
# ICD's im definierten Bereich auswählen und prozentual darstellen
icd_plot <- NULL
for (i in 1:length(icd_bereich[,1])) {icd_plot <- rbind(icd_plot,sum(icd_
    bereich[i,]) / icd_bereich[i,]) }
rownames(icd_plot) <- rownames(icd_bereich)
# "icd_plot" transponieren um ICDs als Spalten zu bekommen
icd_plot <- t(icd_plot)
```

```
37  # Prozentuale Tabelle erstellen
38  icd_plot_proz <- prop.table(x=icd_plot,margin=2)*100
39  ### PLOT BEGINNT ###
40  # Seitenabstand
41  par(mar=c(5,5,7,10))
42  # Y-Achse Markierung
43  ybereich <- pretty(x=c(0,100),n=20)
44  # Plotten
45  icd_bereich_plot <- barplot(height=icd_plot_proz,main=paste("Prozentuale
        Häufigkeit der ICD Codes",icd_von,"-",icd_bis,"in den Landkreisen von
        Sachsen-Anhalt"),col=farben_2,axes=FALSE,axisnames=FALSE)
46  # Legende anzeigen
47  legend(x=max(icd_bereich_plot)+1,y=100,legend=landkreise,fill=farben_2,cex
        =0.9,xpd=TRUE)
48  # X-Achse anzeigen
49  text(x=icd_bereich_plot+0.1,y=-1.8,srt=35,adj=1,xpd=TRUE,labels=colnames(icd_
        plot_proz),cex=0.65)
50  # X-Achsen-Beschriftung
51  mtext(text="ICD-Codes",side=1,line=3)
52  # Y-Achse anzeigen
53  axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)
54  # Y-Achsen-Beschriftung
55  mtext(text="Prozent",side=2,line=2)
56  # Absolute Häufigkeit anzeigen
57    # absolute Häufigkeit der ICD Codes in diesem Barplot bestimmen
58    icd_plot_Freq <- NULL
59    for (i in 1:length(icd_bereich[,1])) {
60      tmp <- subset(x=matrix_abs_h[,15],subset=(colnames(icd_plot_proz)[i] ==
            rownames(matrix_abs_h)))
61      icd_plot_Freq <- cbind(icd_plot_Freq,tmp[[1]])
62    }
63  text(x=icd_bereich_plot+0.35,y=103,adj=1,xpd=TRUE,labels=icd_plot_Freq,cex
        =0.65,col=farben_2[2])
64  text(x=-0.5,y=103,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.75,col=
        farben_2[2])
```

Um einen selbstdefinierten Bereich, anstatt der Diagnoseschlüssel mit der größten Standardabweichung, zu betrachten, enthält dieser Quellcode partiell notwendige Änderungen im Vergleich zum Code der Abbildung 5.13. Für die Begrenzung des ICD-Bereiches werden zwei Variablen (`icd_von` bzw. `icd_bis`) benötigt. Der Bereich wird in Form der fünfstelligen ICD-10-Kapitel als String definiert (Zeile 3-4).

Für die Filterung der Schlüssel des ICD-Bereiches mittels des `subset()`-Befehls in Zeile 30 ist es zuvor erforderlich, die Matrix `matrix_pro_patient` aufsteigend anhand ihrer Zeilenbeschriftungen (ICD-10-Codes) zu sortieren (siehe Zeile 29).
Der Quelltext ab Zeile 6 bis einschließlich 26, die Zeilen 32-38 und die Plotbefehle in den Zeilen 41-64 sind ähnlich dem Quellcode der Abbildung 5.13 und kann in dessen Codebeschreibung auf Seite 104 nachgelesen werden.

5.5.4. Absolute Verteilung einzelner ICD-10-Codes in Sachsen-Anhalt

Wie bereits zu Beginn des Kapitels 5.5.3 erläutert, beinhaltet die Tabelle 5.4 insgesamt nur ca. 16% aller verfügbaren ICD-10-Codes des Datenbestandes. In absoluten Zahlen ausgedrückt entsteht zu der Gesamtanzahl von 10.994 ICD-Codes eine Differenz von 9.224 Codes, die in der tabellarischen Betrachtung nicht enthalten sind. Diese Tatsache deutet ebenfalls auf eine asymmetrische Verteilung hin, da ca. 84% aller Krankheitsbilder nicht in allen Landkreisen/kreisfreien Städten mindestens zehn Mal vorzufinden sind.
Die Untersuchung in diesem Abschnitt befasst sich mit der Identifizierung von Diagnoseschlüsseln, die in möglichst wenigen Regionen besonders häufig auftreten. Dazu wird eine Tabelle mit absoluten Häufigkeiten der 10.994 Codes erzeugt und als CSV-Datei ausgegeben. Ein Ausschnitt der Datei ist in der folgenden Tabelle 5.5 zu finden.

ICD-Code	Magdeburg	Halle	Dessau-Roßlau	Altmark.-Salzw.	Stendal	Börde	Jerichower Land	Harz	Salzlandkreis	Anhalt-Bitterfeld	Wittenberg	Mansfeld Südharz	Saalekreis	Burgenlandkreis	Freq ges	Freq lk
I1090	26216	23125	12099	11139	14826	21949	11001	31706	30356	25444	16363	16087	19699	27764	287774	14
Z2510	21098	15168	8161	9737	12521	16706	9109	19757	21425	17115	14249	10941	13155	19459	208601	14
E1190	12114	9499	5018	4828	6267	9961	4595	12574	13680	10062	8323	7259	8397	11221	123798	14
I2590	9881	7500	4691	3685	5103	7944	2953	10375	11726	9453	6225	5667	7113	8629	100945	14
H5240	10332	10051	3346	5344	5493	6786	4390	9292	10670	6493	7856	4134	6766	8163	99116	14
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
K9200	20	17	7	16	15	21	15	16	25	19	6	11	13	14	215	14
S0081	31	56	15	12	19	7	5	6	21	7	4	9	14	9	215	14
D0480	71	1	0	9	1	2	5	11	0	2	0	0	0	112	214	9
K0280	3	100	8	0	0	1	1	14	11	9	29	11	23	4	214	12
C8290	26	16	14	3	6	19	13	13	22	20	12	10	16	24	214	14
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
M1389	4	4	1	1	0	2	1	7	70	3	1	2	1	3	100	13
N0200	1	0	0	0	0	1	1	0	3	2	13	0	8	70	99	8
M4896	6	0	1	0	1	3	0	2	26	7	49	1	0	3	99	10
L9210	11	11	0	0	4	8	2	8	9	15	10	5	8	8	99	12
E1061	16	14	1	3	0	2	2	5	7	6	4	18	15	6	99	13
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
S9220	7	4	1	2	3	4	3	4	2	8	1	2	2	1	44	14
Z9480	4	7	2	1	1	2	1	3	3	4	1	5	7	3	44	14
Z0988	3	1	0	0	0	0	3	2	32	0	2	0	0	0	43	6
L4130	8	5	3	0	0	3	0	12	2	6	1	0	3	0	43	9
E6430	3	4	1	0	3	3	0	7	6	8	0	1	7	0	43	10
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
K1029	1	0	4	1	1	1	1	2	3	3	1	1	2	1	22	13
Q7610	2	1	1	1	1	1	0	3	2	1	2	1	2	4	22	13
Z4280	0	1	0	0	0	0	0	0	0	0	20	0	0	0	21	2
M1210	0	0	0	0	0	0	0	0	0	0	1	16	4	0	21	3
R8600	0	0	0	1	11	0	9	0	0	0	0	0	0	0	21	3
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
T2430	1	0	0	0	1	1	1	2	1	1	0	2	0	1	11	9
I6020	2	1	1	0	0	0	1	0	1	1	1	1	1	1	11	10
M0732	0	0	0	0	0	0	0	0	0	0	10	0	0	0	10	1
E8410	0	3	0	0	0	0	0	0	0	0	0	7	0	0	10	2
S8550	0	0	0	0	0	0	0	0	8	0	0	2	0	0	10	2
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
Z7520	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1
Z7564	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
Z7610	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1
Z7640	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1

Tab. 5.5.: Absolute Häufigkeit aller ICD-10-Codes pro Region

Zusätzlich zu den Spalten *ICD-Code* und den vierzehn Regionen enthält die Tabelle eine Spalte für die Gesamtsumme der einzelnen Häufigkeiten (`Freq_ges`), sowie eine für die Anzahl der Landkreise/kreisfreien Städte, in der der entsprechende ICD-Code vorkommt (`Freq_lk`). Die Gesamthäufigkeit (`Freq_ges`) ist dabei absteigend und bei glei-

chen Zahlenwerten ist die Häufigkeit der Regionen (`Freq_lk`) aufsteigend sortiert. Der Großteil der ICD-10-Codes weist eine Regionshäufigkeit auf, in der alle vierzehn Landkreise/kreisfreien Städte vertreten sind. Diese Ergebnisse sind für die Untersuchung von untergeordneter Bedeutung.
Betrachtet man beispielweise die ICD-10-Codes `K9200` bis `C8290`, wird die Sortierreihenfolge der Tabelle deutlich. Die absteigende Sortierung der Gesamthäufigkeit wird beim Sprung von 215 zu 214 sichtbar und die aufsteigende Sortierung durch das Mehrfachvorkommen der Häufigkeit von 214 ist in der Spalte `Freq_lk` mit den Werten 9, 12, 14 zu erkennen.
Durch die Anwendung der `subset()`-Funktion in `R` auf die Spalte `Freq_lk`, kann gezielt nach einer bestimmten Häufigkeit oder eines Bereiches gesucht werden:

```
subset(matrix_alle_icd,Freq_lk <= 8)
```

ICD-Code	Magdeburg	Halle	Dessau-Roßlau	Altmark.-Salzw.	Stendal	Börde	Jerichower Land	Harz	Salzlandkreis	Anhalt-Bitterfeld	Wittenberg	Mansfeld Südharz	Saalekreis	Burgenlandkreis	Freq ges	Freq lk
R9370	0	1	605	0	0	0	3	0	2	136	104	1	0	1	853	8
R9360	1	0	545	0	1	0	0	0	5	158	114	0	0	1	825	7
Z1260	3	13	0	1	0	243	0	273	2	0	1	0	8	0	544	8
H1980	0	0	0	0	1	1	0	0	14	1	0	244	1	159	421	7
R9310	2	0	291	0	0	1	1	1	1	30	22	0	0	0	349	8
H6900	0	125	0	1	0	0	1	0	16	8	0	1	85	6	243	8
M9919	3	2	14	0	0	0	0	1	182	6	0	2	0	0	210	7
H2280	0	0	0	0	0	1	0	1	0	0	0	0	2	159	163	4
Z5210	1	1	0	0	1	0	10	0	0	0	1	1	2	130	147	8
K0380	0	3	33	0	0	0	0	0	20	40	28	1	4	0	129	7
M9910	0	0	1	0	2	3	0	0	117	1	0	4	0	1	129	7
H4880	3	28	0	0	0	0	0	1	4	0	4	16	4	59	119	8
B3080	0	5	1	0	1	3	1	1	2	0	98	0	0	0	112	8
M9944	0	0	0	0	0	0	0	0	104	0	0	1	0	0	105	2
R8900	0	0	0	0	2	2	0	30	10	0	1	57	0	0	102	6
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Tab. 5.6.: Absolute Häufigkeit der ICD-10-Codes im eingeschränkten Bereich

Die Tabelle 5.6 ist durch oben angegebenen Befehl entstanden und filtert die Diagnoseschlüssel heraus, die in maximal acht verschiedenen Landkreisen/kreisfreien Städten anzutreffen sind.

Der Auszug lässt einige Auffälligkeiten erkennen. Zum Beispiel weisen die ersten beiden Diagnoseschlüssel `R9370` (*Abnorme Befunde bei der bildgebenden Diagnostik sonstiger Abschnitte des Muskel-Skelett-Systems*) und `R9360` (*Abnorme Befunde bei der bildgebenden Diagnostik der Extremitäten*), sowie der fünfte `R9310` (*Abnorme Befunde bei der bildgebenden Diagnostik des Herzens und des Koronarkreislaufes*), die zusätzlich derselben Subkategorie angehören, eine deutliche regionale Konzentration in der Stadt Dessau-Roßlau auf. Den dritten ICD-10-Code `Z1260` (*Spezielle Verfahren zur Untersuchung auf Neubildung der Harnblase*) teilen sich die Landkreise Börde und der Harz mit einem Anteil von ca. 50% an der Gesamthäufigkeit von 544. Zum Ende der Liste zeigt der Code `M9944` (*Bindegewebige Stenose des Spinalkanals: Sakralbereich*) einen starken lokalen Bezug zum Salzlandkreis mit 104 von insgesamt 105 Fällen. Ebenfalls sind die Diagnoseschlüssel `M9910` (*Subluxation (der Wirbelsäule): Kopfbereich*) und `M9919` (*Subluxation (der Wirbelsäule): Abdomen und sonstige Lokalisationen*) zu einem großen Anteil in dieser Region anzutreffen.

Für die Tabelle 5.5 wird folgender Quelltext benutzt (Seite 221 im Quelltext-Anhang):

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]
# Verbinden der einzelnen Landkreise mit den Ambudaten
magdeburg_icd <- unique(merge(magdeburg,ambu_icd,by="PSEUDONYM"))
{ ... }
# Verbinden aller Landkreise mit den Ambudaten
alle_kreise_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))
# Data.frame mit allen ICD-Codes in Sachsen-Anhalt erstellen
alle_icd <- data.frame("ICD"=unique(alle_kreise_icd$ICD))
# Die Spalte "Freq" hinzufügen und null setzen
alle_icd$Freq <- 0
# Absolute Häufigkeitstabelle der ICD-Codes pro Landkreis
magdeburg_icd_freq <- data.frame(table(magdeburg_icd$ICD))
{ ... }
# Überschriften setzen
colnames(magdeburg_icd_freq) <- c("ICD","Freq")
{ ... }
```

```
22 # Ermittlung der fehlenden ICD-Codes pro Landkreis
23 # Anhängen der fehlenden Codes (mit Freq=0)
24 # Somit haben alle data.frames jedes LK gleiche Anzahl Zeilen
25 icd_temp <- alle_icd[!alle_icd$ICD %in% magdeburg_icd_freq$ICD,]
26 magdeburg_icd_freq <- rbind(magdeburg_icd_freq,icd_temp)
27 { ... }
28 # Liste erstellen mit allen ICDs der Landkreise
29 alle_icd_list <- list (magdeburg_icd_freq,halle_icd_freq,dessau_rosslau_icd_
       freq,altmarkkreis_salzwedel_icd_freq,stendal_icd_freq,boerde_icd_freq,
       jerichower_land_icd_freq,harz_icd_freq,salzlandkreis_icd_freq,anhalt_
       bitterfeld_icd_freq,wittenberg_icd_freq,mansfeld_suedharz_icd_freq,
       saalekreis_icd_freq,burgenlandkreis_icd_freq)
30 # Reduce ist eine Funktion, die merge auf alle Elemente der Liste anwendet
31 matrix_alle_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_
       icd_list)
32 # Die Spalten 2 bis 15 in numeric umwandeln
33 matrix_alle_icd[,2:15] <- sapply(X=matrix_alle_icd[,2:15],FUN=as.numeric)
34 # Gesamt-Häufigkeit (Freq_ges) hinzufügen
35 matrix_alle_icd <- cbind(matrix_alle_icd,"Freq_ges"=rowSums(matrix_alle_icd
       [,2:15]))
36 # Landkreis-Häufigkeit (Freq_lk) hinzufügen
37 for (i in 1:length(matrix_alle_icd[,1])) {matrix_alle_icd[i,17] <- 14-sum(
       matrix_alle_icd[i,2:15]==0)}
38 # Überschriften setzen
39 colnames(matrix_alle_icd) <- c("ICD",landkreise,"Freq_ges","Freq_lk")
40 # Gesamt-Häufigkeit absteigend sortiert und bei gleicher Häufigkeit, die
       Landkreis-Häufigkeit aufsteigend sortiert
41 matrix_alle_icd <- matrix_alle_icd[order(-matrix_alle_icd$Freq_ges,matrix_
       alle_icd$Freq_lk),]
42 ### Tabelle als CSV auf Festplatte speichern
43 write.table(x=matrix_alle_icd,file="CSV/anzahl_icd_pro_lk.csv",sep=";",row.
       names=FALSE,col.names=TRUE)
```

Zeile 2-11 lädt die benötigten Daten und anschließend werden die einzelnen Landkreise/kreisfreien Städte, sowie alle Regionen (`alle_kreise`) mit den Diagnoseschlüsseln verbunden. Der nächste `R`-Befehl erzeugt einen neuen *data.frame* (`alle_icd`) mit allen vorkommenden ICD-10-Codes in Sachsen-Anhalt (10.994 Stück) und fügt in Vorbereitung auf die nachfolgenden Operationen eine Spalte names *Freq*, vorbelegt mit Nullen, hinzu (siehe Zeile 13-15). Anknüpfend daran, werden die ICD-10-Codes in jeder Region gezählt (Zeile 17-18). Um in Zeile 29 eine Liste aller Regionen zu erstellen, ist es notwendig, dass die vierzehn *data.frames* der Regionen eine identische Zeilenanzahl

aufweisen, d.h. jeder Landkreis/kreisfreie Stadt muss zu jedem der 10.994 ICD-10-Codes einen Häufigkeitseintrag besitzen (Zeile 25-27). Ein im jeweiligen *data.frame* nicht vorkommender Diagnoseschlüssel erhält den Häufigkeitswert `0`. In Zeile 31 wird die Ergebnismatrix erstellt, die in den folgenden Zeilen um die Gesamthäufigkeit (`Freq_ges`) und Landkreishäufigkeit (`Freq_lk`) ergänzt und in Zeile 43 als CSV-Datei gespeichert wird.

Ergänzend zu den obigen Tabellendarstellungen, werden in diesem Abschnitt die Ergebnisse als Heatmap präsentiert. Dazu werden einzelne Diagnoseschlüssel entsprechend ihres Vorkommens im Postleitzahlenbereich farblich codiert, wodurch dem Betrachter eine schnelle geografische Lokalisation ermöglicht wird.
Die nachstehenden Abbildungen zeigen exemplarisch ICD-10-Codes mit einer regionsspezifischen Konzentration.

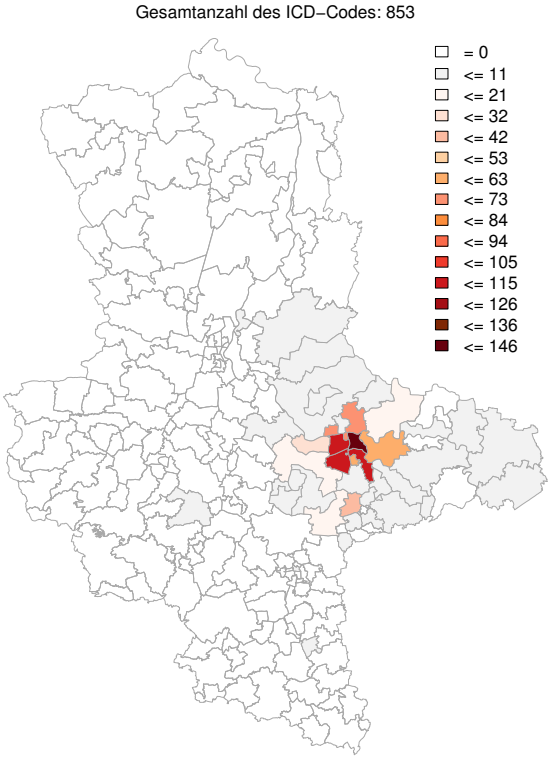

Abb. 5.18.: ICD-10-Code: R9370

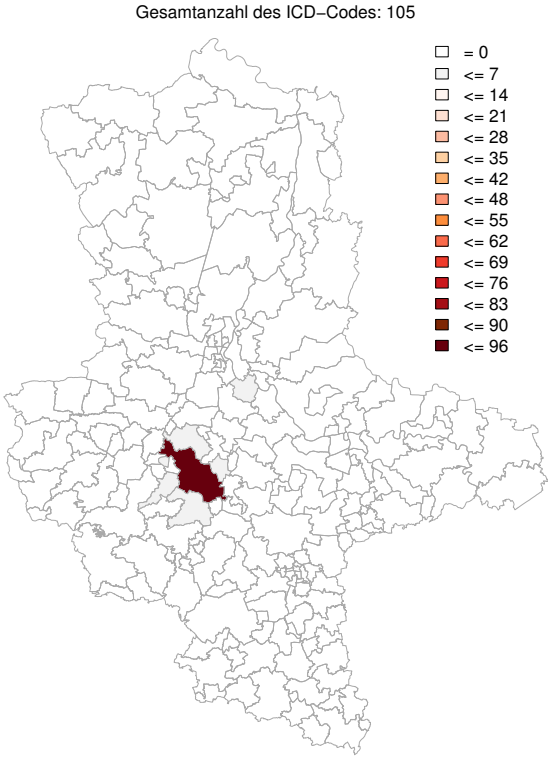

Abb. 5.19.: ICD-10-Code: M9944

Die Abbildung 5.18 zeigt für den Diagnoseschlüssel `R9370` ein deutlich vermehrtes Aufkommen im östlichen Teil Sachsen-Anhalts. Die überwiegenden Teile Wittenbergs und Anhalt-Bitterfelds sind durch graue Bereiche markiert ($\leq$ 11 Fälle pro Postleitzahlgebiet), während das Gebiet Dessau-Roßlau und dessen näherer Umgebung wesentlich stärker mit mehr als 100 Fällen betroffen sind. Der restliche Teil des Landes ist überwiegend durch weiße Flächen (= 0 Fälle) gekennzeichnet.
Für den ICD-10-Code `M9944` liegen 96 von 105 Fällen in einer einzigen Postleitzahl (06449) im Salzlandkreis.

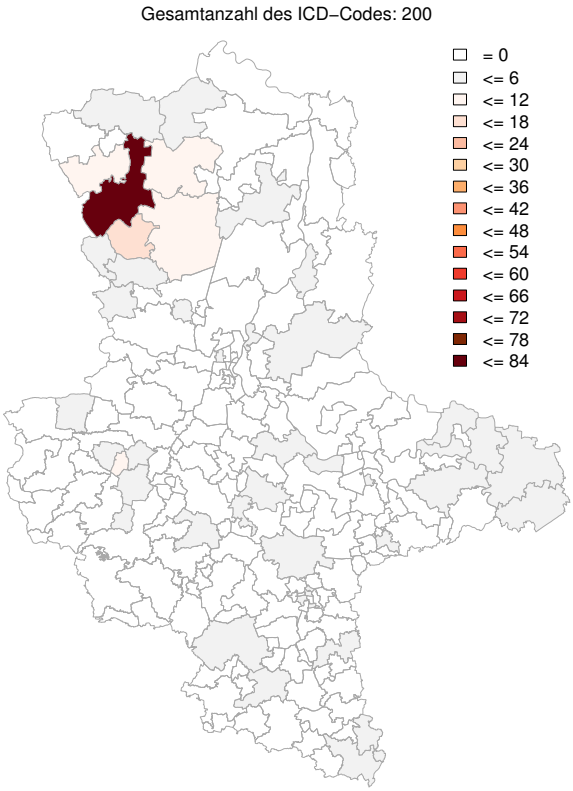

Abb. 5.20.: ICD-10-Code: M5322

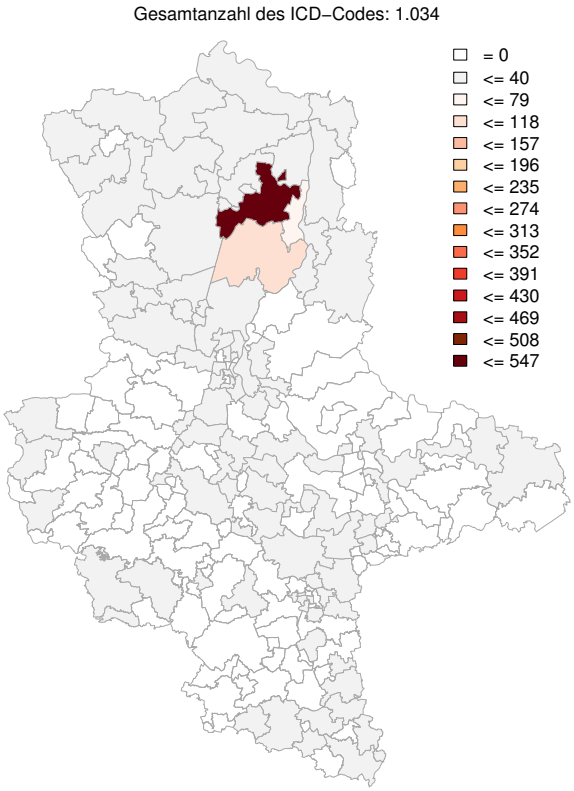

Abb. 5.21.: ICD-10-Code: K5510

Die ICD-10-Codes `M5322` (*Instabilität der Wirbelsäule: Zervikalbereich*) und `K5510` (*Chronische Gefäßkrankheiten des Darmes*) weisen beide eine geringe Anzahl Betrof-

fener über das gesamte Bundesland auf, wobei in der nördlichen, landwirtschaftlich geprägten Region eine hohe Konzentration auftritt.

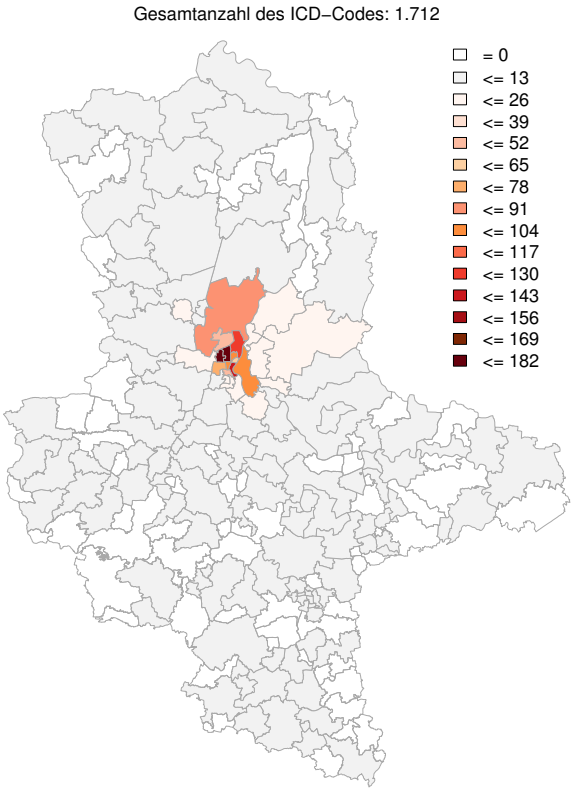

Abb. 5.22.: ICD-10-Code: R0710

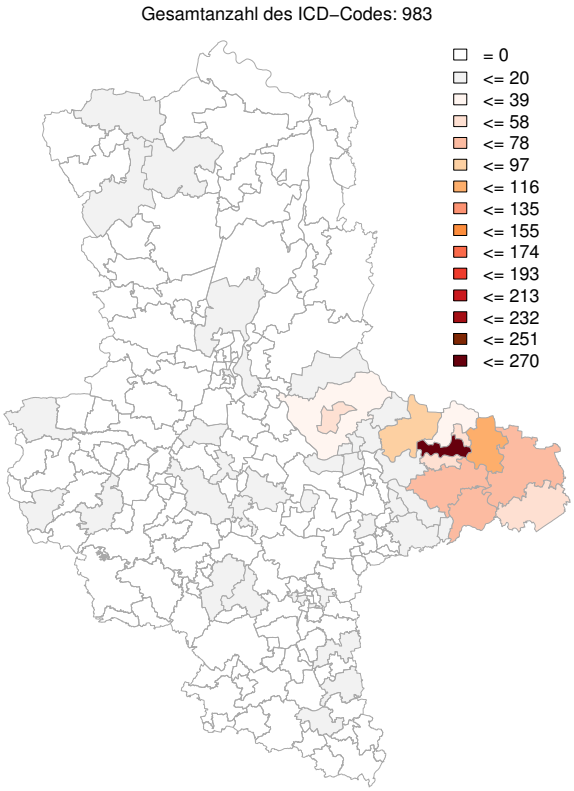

Abb. 5.23.: ICD-10-Code: I1291

Der Diagnoseschlüssel `R0710` (*Brustschmerzen bei der Atmung*) ist im Magdeburger Raum von den insgesamt 1.712 Fällen sichtbar stärker vertreten als im restlichen Teil Sachsen-Anhalts. In den übrigen Landkreisen taucht dieser Schlüssel verhältnismäßig homogen, mit weniger als 13 Fällen pro Postleitzahlgebiet auf.
Der in der Abbildung 5.23 dargestellte Schlüssel `I1291` (*Hypertensive Nierenkrankheit ohne Niereninsuffizienz: Mit Angabe einer hypertensiven Krise*) zeigt auffällig viele Diagnosen im östlichen Teil des Bundeslandes. Hier entfallen ca. 84% aller Fälle auf den Landkreis Wittenberg.

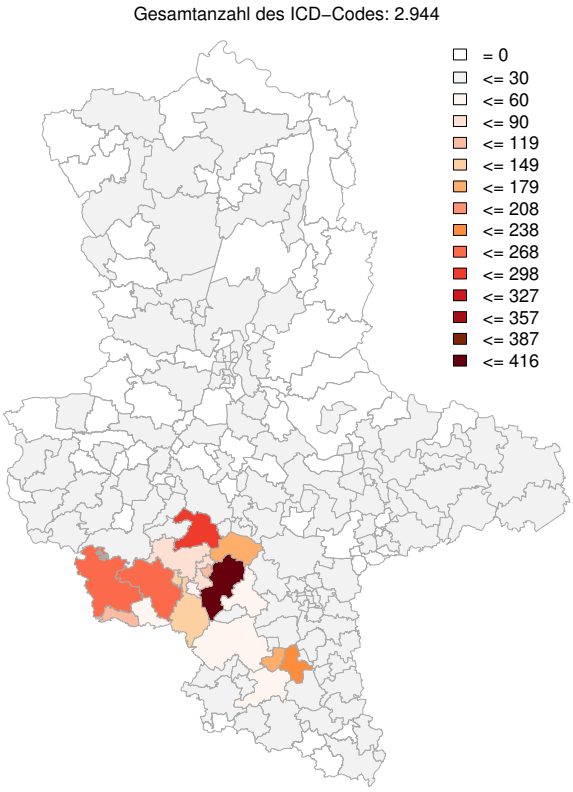

Abb. 5.24.: ICD-10-Code: E2210

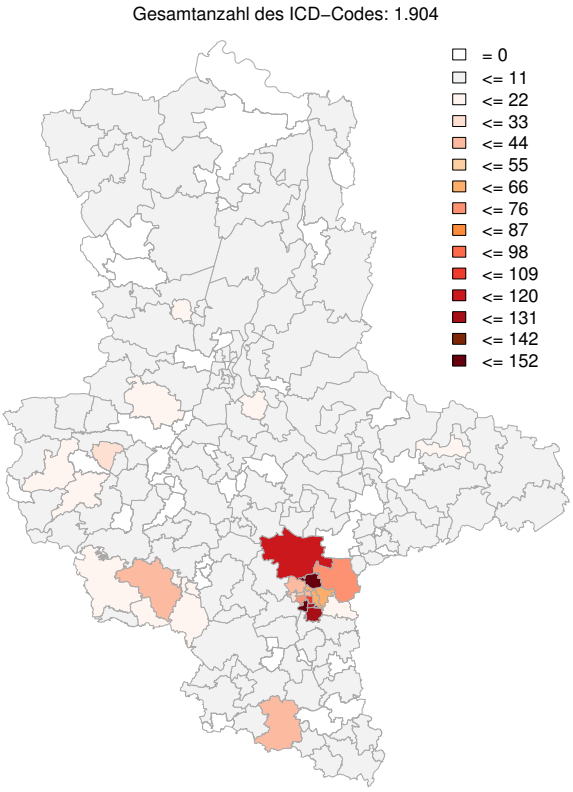

Abb. 5.25.: ICD-10-Code: I1390

Im Bundesland Mansfeld-Südharz befinden sich 2.097 der insgesamt 2.944 Betroffenen, mit dem Diagnoseschlüssel `E2210` (*Hyperprolaktinämie*).
Der ICD-10-Code `I1390` (*Hypertensive Herz- und Nierenkrankheit, nicht näher bezeichnet: Ohne Angabe einer hypertensiven Krise*) ist in allen Regionen schwach vertreten, außer in der kreisfreien Stadt Halle (Saale) und der nördlichen Peripherie.

Die nachfolgenden Abbildungen zeigen Diagnoseschlüssel mit deutlich mehr Fällen im gesamten Bundesland. Für diese Krankheitsbilder wird eine absolute und eine prozentuale Betrachtung vorgenommen, da die Miteinbeziehung der Einwohnerzahl für die relative Berechnung einen großen Einfluss auf das Ergebnis der Untersuchung hat.

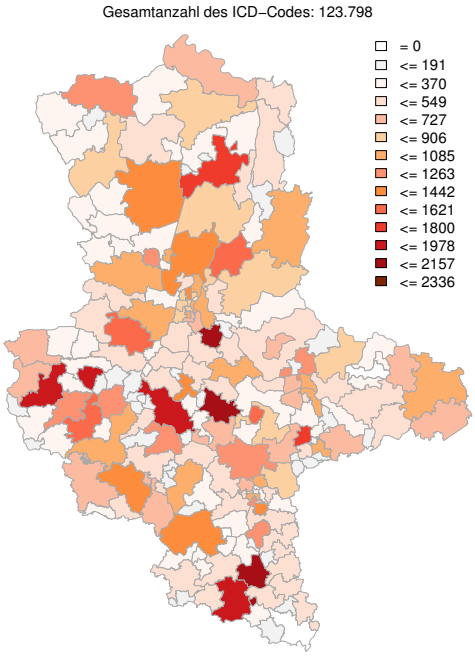

Abb. 5.26.: ICD-10-Code E1190 absolut

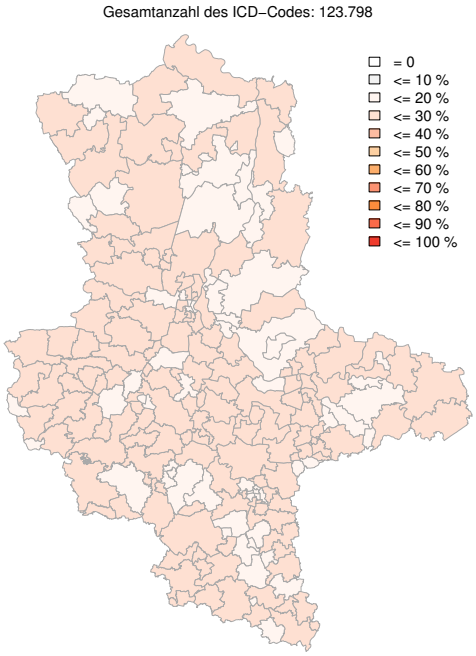

Abb. 5.27.: ICD-10-Code E1190 proz.

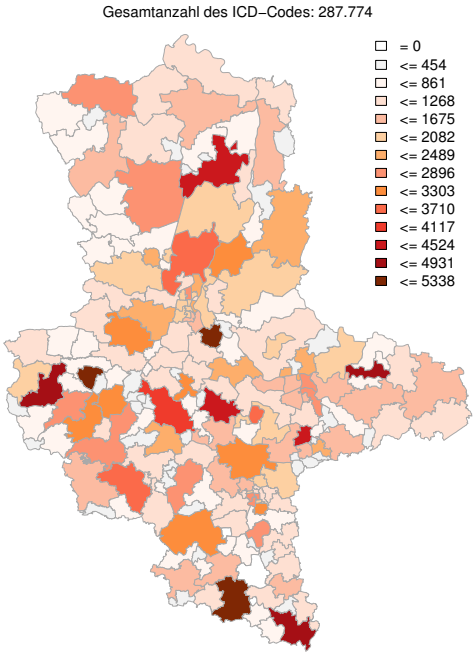

Abb. 5.28.: ICD-10-Code I1090 absolut

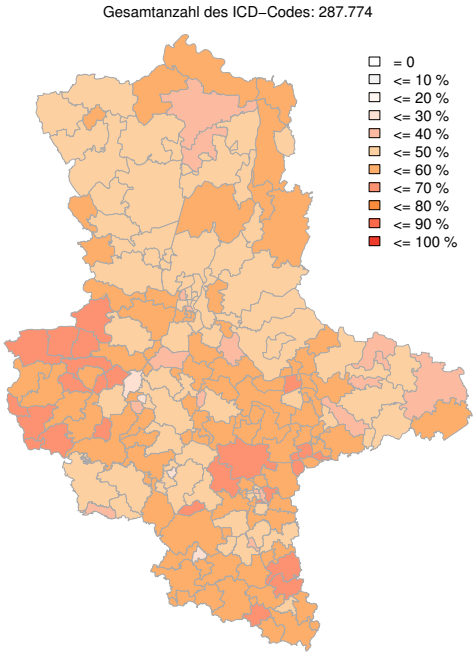

Abb. 5.29.: ICD-10-Code I1090 proz.

Die Abbildungen 5.26 und 5.28 zeigen die absoluten Häufigkeiten der Diagnoseschlüssel `E1190` und `I1090`. Beide lassen ein sehr inhomogenes und fragmentiertes Bild der Verteilung pro Postleitzahlgebiet erkennen, welches auf die divergierende Anzahl der dort ansässigen AOK-Versicherten zurückzuführen ist. Eine prozentuale Ansicht der ICD-10-Codes, unter Berücksichtigung der im jeweiligen Gebiet lebenden Versicherten, erzeugt ein gegenteiliges Bild. Die Abbildungen 5.27 und 5.29 führen zu dem Schluss einer annähernden Gleichverteilung zwischen 20-30% für den Code `E1190` und einem prozentuale Anteil von ca. 50% für `I1090`.

Der zuständige Code der Heatmaps (siehe auch Seite 225 im Quelltext-Anhang):

```
# Variable setzen (ICD-Code)
icd_code = "I1390"
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]
# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebiet)
shapes <- shape_laden(karte="plz")
# Verbinden aller Landkreise mit den Ambudaten
all_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))
# Häufigkeitstabelle der ICD-Codes, gruppiert nach den PLZ
all_icd_freq <- data.frame(table(all_icd$ICD,all_icd$PLZ))
# Überschriften setzen
colnames(all_icd_freq) <- c("ICD","PLZ","Freq")
# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
all_icd_freq$PLZ <- str_pad(string=all_icd_freq$PLZ,width=5,pad="0")
# Data.frame für die Darstellung (maptools) erzeugen
plz_shape <- data.frame(shapes$plz)
plz_shape <- cbind(plz_shape,0)
# Spaltenüberschriften vergeben
colnames(plz_shape) <- c("PLZ","Freq")
# Filterung des ausgewählten ICD-Codes
all_icd_code <- subset(x=all_icd_freq,subset=(ICD==icd_code))
# Spaltenüberschriften vergeben
colnames(all_icd_code) <- c("ICD","PLZ","Freq")
# Anzahl des ICD-Codes pro PLZ-Gebiet in den data.frame "plz_shape" schreiben
for (i in 1:length(plz_shape$PLZ)) {tmp <- subset(x=all_icd_code$Freq,subset
    =(all_icd_code$PLZ == plz_shape[i,1]))
```

```
30    if (length(tmp) == 0) {
31      plz_shape[i,2] <- 0
32    }
33    else {
34      plz_shape[i,2] <- tmp
35    }
36  }
37  # Dynamische Einteilung der Schritte für die Heatmap
38  if (max(plz_shape$Freq) < length(farben_shape)) {
39    break_points <- ceiling(seq(from=min(plz_shape$Freq),to=max(plz_shape$Freq)
          ,by=1))
40  } else {
41    break_points <- ceiling(seq(from=min(plz_shape$Freq),to=max(plz_shape$Freq)
          ,by=(max(plz_shape$Freq)/length(farben_shape))))
42  }
43  class <- cut(plz_shape$Freq,breaks=break_points)
44  levels(class) <- break_points[2:length(break_points)]
45  ### PLOT BEGINNT ###
46  # Plotten
47  plot(x=shapes,border="darkgrey",col=farben_shape[class],main=paste("
        Häufigkeit des ICD-Codes",icd_code,"in Sachsen-Anhalt",sep=" "))
48  # Gesamtanzahl des ICD-Codes anzeigen
49  mtext(text=paste("Gesamtanzahl des ICD-Codes: ",sum(plz_shape$Freq),sep=""),
        side=1,cex=0.7,line=1)
50  # Legende anzeigen
51  legend(x=par("usr")[2]-(1/3.95*par("usr")[2]),y=par("usr")[4],fill="white",
        legend="= 0",border="black",cex=1.2,title="Legende:",bty="n")
52  legend(x=par("usr")[2]-(1/4*par("usr")[2]),y=par("usr")[4]-(1/200*par("usr")
        [4]),fill=farben_shape,legend=paste("<=",levels(class)),border="black",
        cex=1.2,title=NA,bty="n")
```

In der zweiten Zeile wird als Variable (`icd_code`) der darzustellende ICD-10-Code als String angegeben. Anschließend wird das *shapefile* in `R` eingelesen und für alle verfügbaren Diagnoseschlüssel eine Häufigkeitstabelle, gruppiert nach den Postleitzahlengebieten, aufgestellt (Zeile 10-18). In der Zeile 25 wird die Variable `icd_code` als Filterwert der `subset()`-Funktion übergeben, wodurch für die Darstellung der Heatmap nur die Häufigkeitswerte des gewünschten ICD-10-Codes herangezogen werden. Auf die nachfolgenden Befehle wird nur auf Zeile 38-42 eingegangen. Die restlichen Zeilen wurden bereits an früherer Stelle erläutert. In Zeile 38-42 wird für die Legende der Heatmap eine dynamische Einteilung der Schrittweite berechnet. Dies ist erforderlich, da je nach ICD-Code die Gesamtanzahl der betroffenen Personen stark variieren kann.

5.5.5. Zusammenhang zwischen Krankheitsbildern und dem Versicherungsstatus der AOK-Versicherten

Als abschließende Analyse bezüglich der Diagnoseschlüssel wird in diesem Abschnitt der Versuch unternommen, einen Zusammenhang zu den oben untersuchten Versicherungsarten (Kapitel 5.2) und einzelnen Krankheitsbildern herzustellen. Dazu werden die AOK-Versicherten in unterschiedliche Bevölkerungsgruppen, repräsentiert durch ihren Versicherungsstatus, aufgeteilt und geprüft, inwieweit dort besonders häufig oder selten bestimmte Erkrankungen auftreten. Das Ergebnis soll ein differenzierteres Bild über einen eventuellen Zusammenhang liefern und feststellen ob Abhängigkeiten bestehen. Der Versicherungsstatus wird den Stammdaten entnommen, wobei die zusammengefasste Gruppe *Sonstige* unberücksichtigt bleibt, da diese keine spezifizierbare Bevölkerungsgruppe darstellt. Die Analyse stützt sich dementsprechend auf die vier Hauptversicherungsarten *Arbeitslose*, *Beschäftigte*, *Familienangehörige* und *Rentner*.

Dazu werden zu Beginn die Diagnoseschlüssel, gruppiert nach den vier Versicherungsarten, prozentual normiert und unter Verwendung der Standardabweichung absteigend sortiert. Mit Hilfe der Standardabweichung können, wie bereits im Kapitel 5.5.3 erläutert, Auffälligkeiten in der Verteilung der Versicherungsarten pro Diagnoseschlüssel ermittelt werden.
Anschließend werden für diese ICD-10-Codes die regionalen Häufigkeiten gemäß der Berechnung in Abbildung 5.13 ermittelt und in Relation zu den Versicherungsarten gesetzt. Das Resultat besteht aus einer kombinierten Grafik und stellt jeweils für einen Diagnoseschlüssel die Versicherungsarten und das regionale Vorkommen prozentual als Balkendiagramm dar.
In der folgenden Abbildung 5.30 sind die ersten 40 Diagnoseschlüssel mit der höchsten Standardabweichung bezüglich des Versicherungsstatus abgebildet.

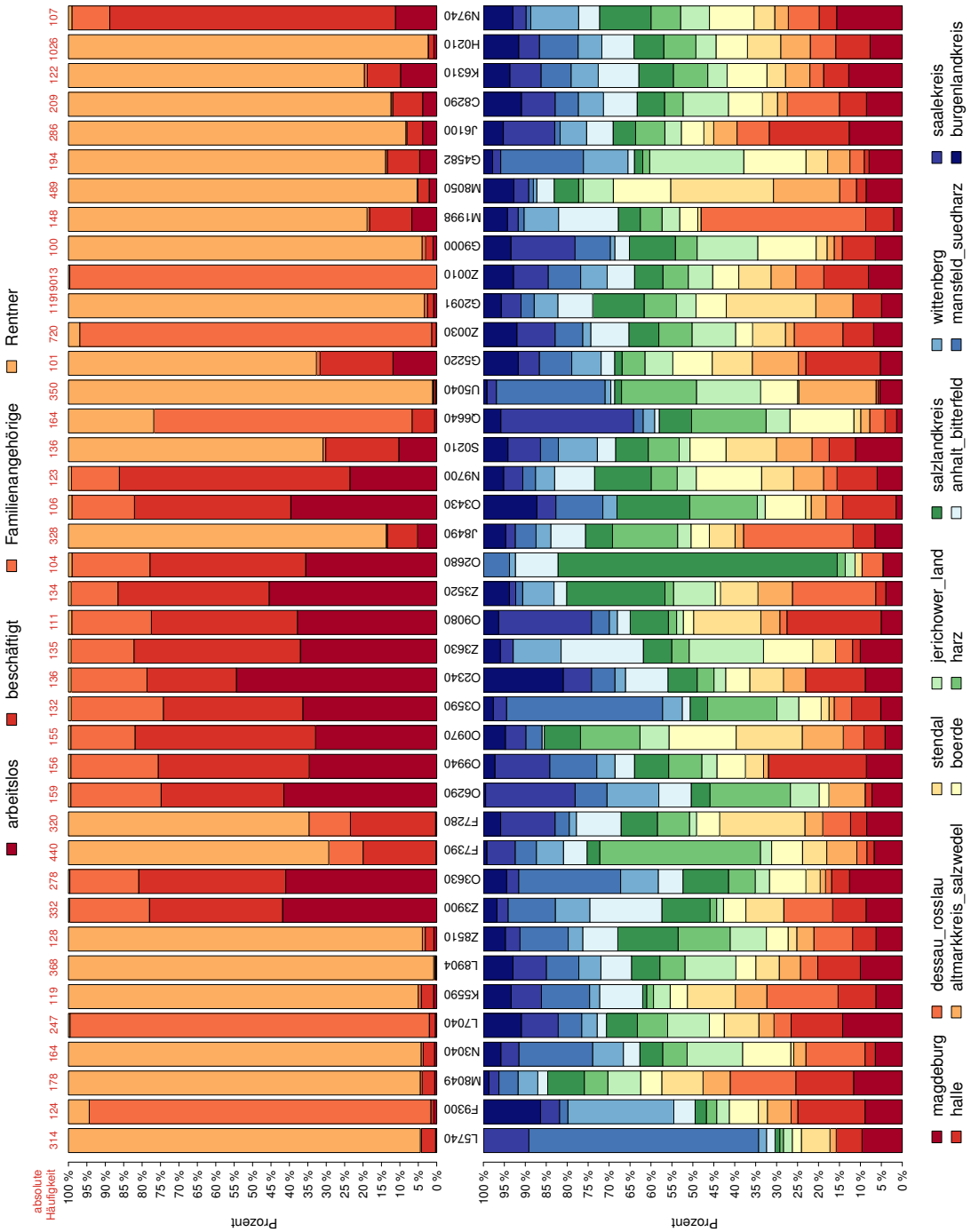

Abb. 5.30.: ICD-10-Codes und die Versicherungsarten

Die Abbildung 5.30 zeigt auf der X-Achse die ICD-10-Codes und auf der Y-Achse die prozentualen Anteile der vier Versicherungsarten bzw. der vierzehn Landkreise/ kreisfreien Städte. Die roten Zahlenwerte oberhalb des Diagramms geben die absolute Häufigkeit der Diagnoseschlüssel an. Um statistische Aussagen treffen zu können, werden nur solche herangezogen, die im gesamten Land mindestens 100 mal auftreten.
Die Grafik lässt erkennen, dass bei der Mehrheit der Diagnoseschlüssel die Gruppe der *Rentner* stark überwiegt. Ebenfalls sind ICD-Codes vorhanden, bei denen diese Gruppe zu annähernd 0% vorhanden ist. Ein Großteil dieser Codes stammt erwartungsgemäß aus dem Kapitel XV (`O00-O99`) bzw. (*Schwangerschaft, Geburt und Wochenbett*). Die übrigen drei Versicherungsarten *arbeitslos, beschäftigt* und *familienangehörig* sind moderat gleich verteilt, mit einem leichten Überhang bei den Arbeitslosen.
Im unteren Balkendiagramm stechen zwei ICD-10-Codes in jeweils einer Region besonders hervor. Im Landkreis Mansfeld-Südharz der Code `L5740` zu ca. 55% und im Salzlandkreis der Code `O2680` mit ca. 68%. Während ein Regionalitätsbezug zum Versicherungsstatus im erst genannten zu erkennen ist, ist dies im Salzlandkreis nicht der Fall, da die drei Versicherungsarten homogen verteilt sind.

Es folgt der zuständige `R`-Quellcode. Eine vollständige Auflistung ist auf Seite 226 zu finden.

```
# Variable setzen
  # Anzahl von ICD Codes die dargestellt werden
  # Default: 55
  anzahl_icd <- 40
  # Mindesthäufigkeit pro ICD-Code in Sachsen-Anhalt
  # Default: 100
  mindest_haeufigkeit <- 100
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]
# Verbinden der Stammdaten (alle Landkreise) mit den Ambudaten
vart_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))
### Beginn: Einteilung der ICD-Codes nach der Versicherungsart
# Aggregation der Versicherungsarten nach den ICD-Codes
```

```
agg_vart_icd <- aggregate(vart_icd$VERSICHERUNGSART,by=list(ICD=vart_icd$ICD)
    ,FUN=table)
# Arbeitslose, Beschäftigte, Familienang. und Rentner extrahieren
matrix_vart_icd_absolut <- cbind(
  agg_vart_icd$x[,1],
  agg_vart_icd$x[,2],
  agg_vart_icd$x[,3],
  agg_vart_icd$x[,8]
)
# Summierung der Zeilen und als Spalte hinzufügen
matrix_vart_icd_absolut <- cbind(matrix_vart_icd_absolut,rowSums(matrix_vart_
    icd_absolut))
# Überschriften setzen
rownames(matrix_vart_icd_absolut) <- agg_vart_icd$ICD
colnames(matrix_vart_icd_absolut) <- c("arbeitslos","beschäftigt","
    Familienangehörige","Rentner","row_Freq")
# Matrix sortieren nach der row-Frequenz
matrix_vart_icd_absolut <- matrix_vart_icd_absolut[order(-matrix_vart_icd_
    absolut[,5]),]
# Erstellung einer Matrix der Form: 1 Fall auf X Versicherte
matrix_vart_icd_pro_patient <- data.frame(length(alle_kreise$PSEUDONYM) /
    matrix_vart_icd_absolut[,1:4])
# Überschriften setzen
colnames(matrix_vart_icd_pro_patient) <- c("arbeitslos","beschäftigt","
    Familienangehörige","Rentner")
# Spalte hinzufügen: absolute Häufigkeit
matrix_vart_icd_pro_patient <- cbind(matrix_vart_icd_pro_patient,abs_h=matrix
    _vart_icd_absolut[,5])
# Filterung der ICD-Codes mit der oben gesetzten Mindesthäufigkeit
matrix_vart_icd_pro_patient <- subset(x=matrix_vart_icd_pro_patient,subset=(
    matrix_vart_icd_pro_patient[,5] >= mindest_haeufigkeit))
# Spalte hinzufügen: Standardabweichung
matrix_vart_icd_pro_patient <- cbind(matrix_vart_icd_pro_patient,st_abw=apply
    (X=matrix_vart_icd_pro_patient[,1:4],MARGIN=1,FUN=sd2))
# Sortierung der Matrix nach der größten Standardabweichung
matrix_vart_icd_pro_patient <- matrix_vart_icd_pro_patient[order(-matrix_vart
    _icd_pro_patient[,6]),]
# ICD-Codes mit der größten Standardabweichnung werden ausgewählt und
    prozentual dargestellt
matrix_vart_icd_pro_patient_proz <- NULL
for (i in 1:anzahl_icd) {matrix_vart_icd_pro_patient_proz <- rbind(matrix_
    vart_icd_pro_patient_proz,sum(matrix_vart_icd_pro_patient[i,1:4]) /
    matrix_vart_icd_pro_patient[i,1:4])}
matrix_vart_icd_pro_patient_proz <- as.matrix(matrix_vart_icd_pro_patient_
```

```
    proz)
49  matrix_vart_icd_pro_patient_proz <- prop.table(x=matrix_vart_icd_pro_patient_
    proz,margin=1)*100
50  top <- t(matrix_vart_icd_pro_patient_proz)
51  ### Beginn: Einteilung der ICD-Codes nach den Landkreisen
52  top_icd <- data.frame("ICD"=colnames(top))
53  # Filterung der gewünschten ICD-Codes aus den Ambudaten
54  ambu_icd_bereich <- unique(ambu_icd[ambu_icd$ICD %in% top_icd$ICD,])
55  # Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine
    Häufigkeitstabelle auf Basis der ICD-Spalte erstellen
56  magdeburg_vart_icd <- data.frame(table("ICD"=subset(merge(magdeburg,ambu_icd_
    bereich,by="PSEUDONYM"),select=c(ICD))))
57  { ... }
58  # Die Spalte "pro_patient" jedem LK hinzufügen
59  # "pro_patient" beschreibt: 1 Fall auf X Versicherte
60  magdeburg_vart_icd <- cbind(magdeburg_vart_icd,"pro_patient"=length(magdeburg
    $PSEUDONYM) / magdeburg_vart_icd$Freq)
61  { ... }
62  # Ergebnismatrix erzeugen (1 Fall auf X Versicherte pro LK)
63  # Mit Nullen vorbelegen damit alle eine identische Länge besitzen
64  matrix_vart_icd_landkreise <- matrix(data=0,nrow=length(top_icd$ICD),ncol=
    length(landkreise))
65  dimnames(matrix_vart_icd_landkreise) <- list(colnames(top),landkreise)
66  # Liste erstellen mit allen ICD der Landkreise
67  lk <- list (magdeburg_vart_icd[c(1,3)],halle_vart_icd[c(1,3)],dessau_rosslau_
    vart_icd[c(1,3)],altmarkkreis_salzwedel_vart_icd[c(1,3)],stendal_vart_icd
    [c(1,3)],boerde_vart_icd[c(1,3)],jerichower_land_vart_icd[c(1,3)],harz_
    vart_icd[c(1,3)],salzlandkreis_vart_icd[c(1,3)],anhalt_bitterfeld_vart_
    icd[c(1,3)],wittenberg_vart_icd[c(1,3)],mansfeld_suedharz_vart_icd[c(1,3)
    ],saalekreis_vart_icd[c(1,3)],burgenlandkreis_vart_icd[c(1,3)])
68  # Ergebnismatrix mit den Werten belegen
69  for (i in 1:length(matrix_vart_icd_landkreise[1,])) {
70    for (y in 1:length(matrix_vart_icd_landkreise[,1])) {
71      tmp <- subset(lk[[i]][2], lk[[i]]$ICD == rownames(matrix_vart_icd_
          landkreise)[y])
72      matrix_vart_icd_landkreise[y,i] <- tmp[1,]
73    }
74  }
75  matrix_vart_icd_landkreise[is.na(matrix_vart_icd_landkreise)] <- 0
76  # Ergebnisse prozentual darstellen
77  top_landkreise <- NULL
78  for (i in 1:length(matrix_vart_icd_landkreise[,1])) {top_landkreise <- rbind(
    top_landkreise,sum(matrix_vart_icd_landkreise[i,1:14]) / matrix_vart_icd_
    landkreise[i,1:14])}
```

```
79  rownames(top_landkreise) <- rownames(matrix_vart_icd_landkreise)
80  # "top_landkreise" transponieren um ICDs als Spalten zu bekommen
81  top_landkreise <- t(top_landkreise)
82  # Inf (Unendlich) mit Nullen ersetzen
83  top_landkreise[which(top_landkreise==Inf)] <- 0
84  # Prozentuale Tabelle erstellen
85  top_landkreise_proz <- prop.table(x=top_landkreise,margin=2)*100
86  ### PLOT BEGINNT ###
87  # Fenster in zwei gleichgroße Teile einteilen
88  layout(mat=matrix(c(1,1,2,2),nrow=2,ncol=2,byrow=TRUE))
89  # Seitenabstand oberes Diagramm
90  par(mar=c(2,3,5,7))
91  # Y-Achse Markierung
92  ybereich <- pretty(x=c(0,100),n=20)
93  # Plotten oben
94  top_st_abw_plot <- barplot(height=top,main="ICD-Codes nach den
        Versicherungsarten",col=farben_2,axes=FALSE,axisnames=FALSE)
95  # 1. Legende anzeigen
96  legend(x=(max(top_st_abw_plot)+1),y=100,legend=rownames(top),fill=farben_2,
        cex=0.7,xpd=TRUE,border=NA)
97  # 1. Y-Achse anzeigen
98  axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)
99  # 1. Y-Achsen-Beschriftung
100 mtext(text="Prozent",side=2,line=1.7,cex=0.8)
101 # Absolute Häufigkeit anzeigen
102 text(x=top_st_abw_plot+0.3,y=104,adj=1,xpd=TRUE,labels=matrix_vart_icd_pro_
        patient[1:anzahl_icd,5],cex=0.65,col=farben_2[2])
103 text(x=-0.5,y=104,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.55,col=
        farben_2[2])
104 # X-Achse anzeigen
105 text(x=top_st_abw_plot+0.1,y=-2,srt=90,adj=1,xpd=TRUE,labels=colnames(top),
        cex=0.6)
106 # Seitenabstand unteres Diagramm
107 par(mar=c(2,3,0,7))
108 # Plotten unten
109 top_st_abw_plot_landkreise <- barplot(height=top_landkreise_proz,col=farben_
        2,axes=FALSE,axisnames=FALSE)
110 # 2. Legende anzeigen
111 legend(x=(max(top_st_abw_plot)+1),y=100,legend=rownames(top_landkreise),fill=
        farben_2,cex=0.7,xpd=TRUE,border=NA)
112 # 2. Y-Achse anzeigen
113 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)
```

```
114  # 2. Y-Achsen-Beschriftung
115  mtext(text="Prozent",side=2,line=1.7,cex=0.8)
116  # X-Achsen-Beschriftung
117  mtext(text="ICD Codes",side=1,line=0.5,cex=0.8)
```

In den Zeilen 4 und 7 können vom Benutzer die Anzahl der darzustellenden ICD-10-Codes (`anzahl_icd`) und deren Mindestvorkommen im Bundesland (`mindest_haeufigkeit`) definiert werden. Für die Berechnung des oberen Balkendiagramms sind die Zeilen 16-50 zuständig. Der erste Schritt besteht in der Aggregation des Versicherungsstatus pro ICD-10-Code. Dazu wird in Zeile 18 die `aggregate()`-Funktion verwendet, deren erster Parameter das Objekt enthält, auf den die im dritten Parameter `FUN=` übergebene Funktion `table()` angewendet wird. Der Parameter `by=` legt die Werte fest, nach denen die Gruppierung erfolgt. In den Zeilen 20-32 werden die vier Hauptversicherungsarten in einer Matrix extrahiert (`matrix_vart_icd_absolut`). Deren Werte werden als neue Spalte zeilenweise summiert, beschriftet und sortiert. Basierend auf den absoluten Häufigkeitswerten wird eine Matrix in der Form *1 Versicherungsart pro X Versicherte* erstellt, um in Zeile 46-50 einen prozentualen Anteil der Versicherungsarten auf einen Diagnoseschlüssel berechnen zu können. In einem Zwischenschritt wird die Matrix im Vorfeld bereinigt, um die Diagnoseschlüssel, deren Anzahl unterhalb der gesetzten Mindesthäufigkeit (`mindest_haeufigkeit`) liegt und wird dann absteigend nach der Standardabweichung sortiert (Zeile 40-44).
Der Codeabschnitt von 51-85 beinhaltet `R`-Befehle zur Berechnung der unteren Darstellung. Um die Kompatibilität mit dem oberen Diagramm zu gewährleisten, muss zuerst der *data.frame* `ambu_icd`, der die Pseudonyme und ICD-Codes des Datenbestandes enthält und anknüpfend daran die einzelnen *data.frames* der vierzehn Regionen, entsprechend gefiltert werden (Zeile 54-57). Für die grafische Darstellung müssen die Regionen spaltenweise und die Diagnoseschlüssel zeilenweise vorliegen, sodass in Zeile 64 eine entsprechende Matrix (`matrix_vart_icd_landkreise`) mit Nullen initialisiert und in Zeile 69-75 mit den jeweiligen Werten aus der Liste `lk` belegt wird. Nachfolgend werden die prozentualen Werte berechnet (Zeile 77-85).
Die Ausgabebefehle erstrecken sich von Zeile 86-117. Eine Besonderheit im Gegensatz zu allen vorherigen Grafiken ist der Befehl `layout()`, in der das Blatt in zwei Grafikfenster geteilt wird, um zwei verschiedene `barplot()`-Aufrufe als Kombination auszugeben.

5.6. Medikationen in Sachsen-Anhalt

Das Anatomisch-Therapeutisch-Chemische Klassifikationssystem (ATC) ist ein von der Weltgesundheitsorganisation (WHO) herausgegebenes Codiersystem für Arzneistoffe mit definierten Tagesdosen (DDD), wobei die Gliederung der Arzneiwirkstoffe innerhalb des Klassifikationssystems nach der organischen Einwirkung und den therapeutischen, pharmakologischen und chemischen Eigenschaften geschieht [5].
Das ATC-System klassifiziert Substanzen und ermöglicht es somit, verschiedene wirkstoffgleiche Präparate zu identifizieren und zu vergleichen. Die Schaffung eines einheitlichen Bezugs bedingt einen strukturierten Aufbau. Die Wirkstoffe gruppieren sich in fünf hierarchisch aufgebauten Ebenen. Die erste Ebene gliedert sich in vierzehn anatomische Hauptgruppen und enthält chemische/pharmakologische/therapeutische Untergruppen. Die fünfte Ebene spezifiziert den konkreten chemischen Wirkstoff [29].
In der unten stehenden Tabelle 5.7 ist exemplarisch der Wirkstoff *Acetylsalicylsäure* (ASS), auch bekannt als Aspirin, nach den fünf Ebenen aufgeschlüsselt.

ATC-Code	ATC-Bedeutung	ATC-Ebene
B	Blut und blutbildende Organe	1. Ebene (anatomische Hauptgruppe)
B `01`	Antithrombotische Mittel	2. Ebene (therapeutische Untergruppe)
B01 `A`	Antithrombotische Mittel	3. Ebene (pharmakologische Untergruppe)
B01A `C`	Thrombozytenaggregations-hemmer ohne Heparin	4. Ebene (chemische Untergruppe)
B01AC `06`	Acetylsalicylsäure	5. Ebene (chemische Substanz)

Tab. 5.7.: Aufschlüsselung des ATC-Codes B01AC06

Die erste Ebene ist durch einen Buchstaben gekennzeichnet und bildet eine der vierzehn Hauptgruppen ab. Die zweite Ebene wird mit zwei Ziffern beschrieben und weist auf die therapeutische Untergruppe des Wirkstoffes hin, wohingegen die dritte und vierte Ebene durch einen Buchstaben spezifiziert werden und die pharmakologische bzw. chemische Untergruppe festlegen. Die unterste Ebene umfasst zwei Ziffern und klassifiziert somit die konkrete chemische Substanz. Der vollständige ATC-Code `B01AC06` bezeichnet alle als Thrombozytenaggregationshemmer eingesetzten Medikamente, die den Wirkstoff ASS

enthalten. ASS kann auch als Schmerzmittel eingesetzt werden und hat dann den ATC-Code `N02BA01`.
In diesem Kapitel werden die Arzneiwirkstoffe bezüglich ihres Vorkommens aus dem kompletten Datenbestand, sowie ihres regionalen Vorkommens im Bundesland Sachsen-Anhalt untersucht. Anschließend werden bestimmte ATC-Codes, die aufgrund spezieller Eigenschaften in der sog. *PRISCUS*-Liste zusammengefasst sind, näher untersucht.

5.6.1. Absolute Verteilung der ATC-Codes

Die erste Aufgabe besteht darin, sämtliche im ambulanten Datenbestand verschriebenen Medikationen aus dem Jahr 2012 aufzulisten und entsprechend ihres Vorkommens zu sortieren. Das Ergebnis dieser Zusammenstellung erlaubt einen ersten Blick auf die Verteilungsstruktur der verordneten Medikamente.
Hierzu wird die `Wohlfahrt_Arzneimittel.txt` herangezogen, in der sämtliche Medikationen in Form der ATC-Codes für jeden Patienten festgehalten sind.

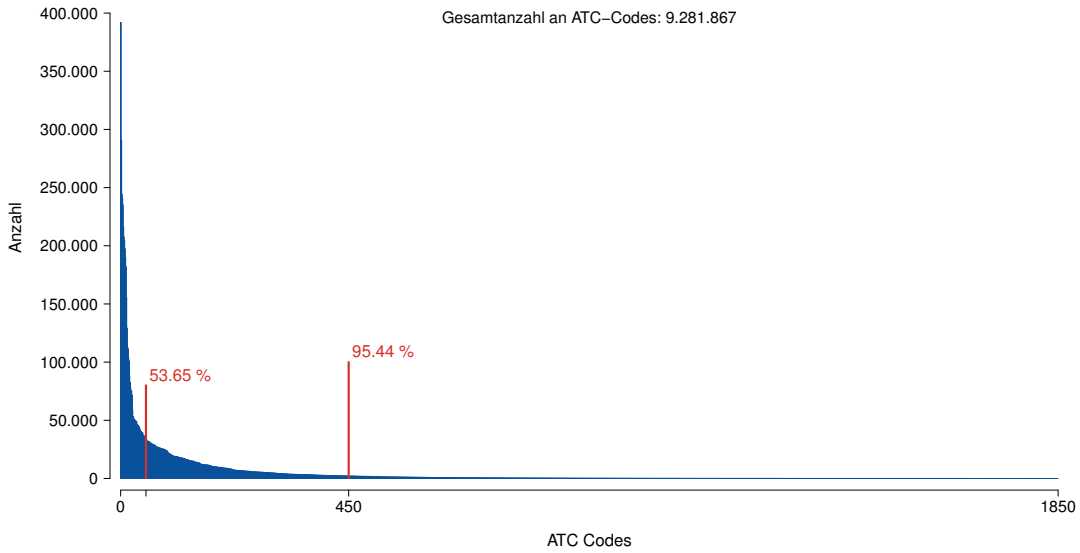

Abb. 5.31.: Gesamtübersicht ATC-Code-Verteilung

Die ATC-Codes bilden in Abbildung 5.31 die Einheiten der X-Achse und deren Häufigkeitsanzahl skaliert die Y-Achse. Die im Jahr 2012 verschriebenen ca. 9,28 Mio. Rezepte,

beruhen auf insgesamt 1850 verschiedenen Wirkstoffen. Während ein geringer Anteil an Substanzen ausgesprochen häufig verordnet wird, kommt ein Großteil der ATC-Codes eher selten vor. Die beiden roten Markierungen verdeutlichen dies: die 50 häufigsten ATC-Codes nehmen von den Gesamtverordnungen bereits über 53% ein, sodass die restlichen 1800 Codes auf die Differenz entfallen ($\approx 47\%$).
Die Grafik ähnelt stark dem Kurvenverlauf der Abbildung 5.8, in der die Diagnoseschlüsselhäufigkeiten dargestellt sind. Diese Übereinstimmung ist ein erstes Anzeichen für die mengenmäßige Relation zwischen den diagnostizierten Erkrankungen und den von den Ärzten verordneten Medikamenten.
Entsprechend dem Vorgehen bei den Abbildungen der Diagnoseschlüssel wird in gleicher Weise der Maßstab in der Folgeabbildung vergrößert.

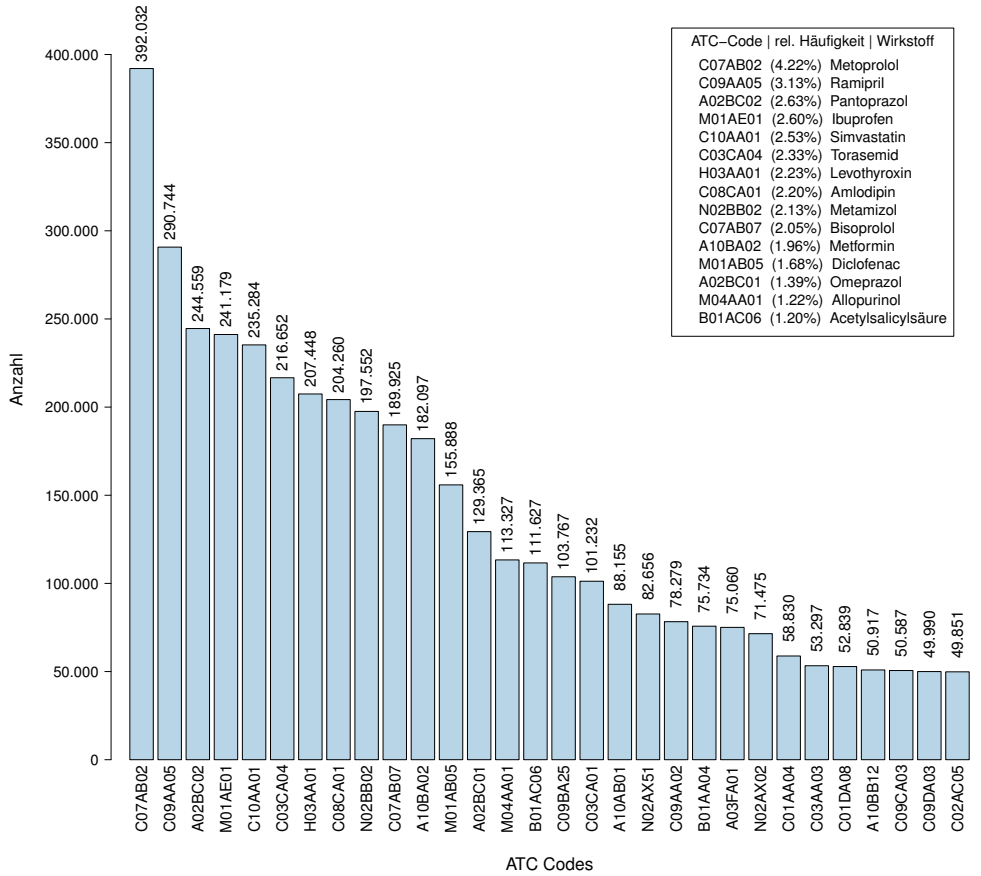

Abb. 5.32.: Die 30 häufigsten ATC-Codes

Die Abbildung 5.32 zeigt auf der X-Achse die häufigsten 30 ATC-Codes in Säulenform an, während die Y-Achse unverändert die absolute Anzahl darstellt.
Der am häufigsten verordnete Wirkstoff ist *Metoprolol* (`C07AB02`) mit 392.032 Medikationen im Jahr 2012, sodass der Anteil an der Gesamtanzahl von 9.281.867 ca. 4.22% beträgt. *Metoprolol* ist ein sog. Betablocker und wird zur Behandlung von Patienten mit Bluthochdruck verschrieben. Die hohe Anzahl an Verordnungen steht in einem plausiblen Verhältnis zu dem am häufigsten diagnostizierten Diagnoseschlüssel `I1090` (*Essentielle Hypertonie, nicht näher bezeichnet: Ohne Angabe einer hypertensiven Krise*) aus Abbildung 5.9. Platz zwei in der Grafik, mit knapp über einem Viertel weniger Verschreibungen, nimmt *Ramipril* (`C09AA05`) ein, das ebenfalls bei arterieller Hypertonie eingesetzt wird.
Der Balkenverlauf verzeichnet einen weiteren stärkeren Abfall von knapp 46.000 ausgestellten Rezepten zum ATC-Code `A02BC02`. Der restliche Verlauf zeigt eine stetig abfallende Kurve mit kleineren Sprüngen.

Der Quelltext der ATC-Häufigkeitsverteilung (Abbildung 5.31):

```
1  # Variablen setzen (rote Markierungen)
2  # default: 50
3  mark_1 <- 50
4  # default: 450
5  mark_2 <- 450
6  # initial laden und ausführen
7  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
8  # Arzneidaten einlesen
9  arznei <- arzneimittel_laden()
10 # Absolute Häufigkeitstabelle der ATC-Codes erstellen
11 arznei_absh <- data.frame(table("ATC"=arznei$ATC))
12 # Häufigkeitstabelle absteigend sortieren
13 arznei_absh <- data.frame(arznei_absh[order(-arznei_absh$Freq),])
14 ### PLOT BEGINNT ###
15 # Seitenabstand
16 par(mar=c(6,6,4,2))
17 # Plotten
18 barplot(height=arznei_absh[,2],yaxt="n",main="Häufigkeitsverteilung der ATC-
       Codes",xlab="ATC Codes",ylab="Anzahl",ylim=c(0,1.1*max(arznei_absh[,2])),
       space=0,col=farben[1],border=farben[1])
19 # Gesamtanzahl ATC-Codes anzeigen
```

```
20 mtext(text=paste("Gesamtanzahl an ATC-Codes: ",format(x=length(arznei$ATC),
       big.mark=".",scientific=FALSE),sep=""),side=3,cex=0.7,line=-1)
21 # Y-Achse in 50.000 Schritten definieren
22 ybereich <- pretty(x=c(0,max(arznei_absh[,2])),n=10)
23 # Y-Achse anzeigen
24 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=
       FALSE),las=2,cex.axis=0.8,pos=-17)
25 # X-Achse definieren
26 xbereich <- c(0,mark_1,mark_2,length(arznei_absh[,1]))
27 # X-Achse anzeigen
28 axis(side=1,at=xbereich,cex.axis=0.8,pos=-10000)
29 # Aufsummierte relative Häufigkeit berechnen
30 rel <- arznei_absh$Freq * 100 / sum(arznei_absh$Freq)
31 sum_mark_1 <- sum(rel[1:mark_1])
32 sum_mark_2 <- sum(rel[1:mark_2])
33 # Rote Striche anzeigen
34 lines(x=mark_1,y=80000,col=farben_2[2],type="h")
35 lines(x=mark_2,y=100000,col=farben_2[2],type="h")
36 text(x=mark_1+35,y=88000,labels=paste(round(x=sum_mark_1,digits=2),"%"),cex
       =0.75,xpd=TRUE,col=farben_2[2])
37 text(x=mark_2+35,y=108000,labels=paste(round(x=sum_mark_2,digits=2),"%"),cex
       =0.75,xpd=TRUE,col=farben_2[2])
38 ### Tabelle als CSV auf Festplatte speichern (abs. Häufigkeit der ATC-Codes)
39 write.table(x=arznei_absh,file="CSV/ATC-Codes_absolute_haeufigkeit.csv",sep="
       ;",row.names=FALSE,col.names=TRUE,fileEncoding="latin1")
```

In Zeile 3 bzw. 5 können die X-Werte für die roten Markierungslinien gesetzt werden. Anschließend wird die `initial.R` geladen und auf Basis der Datei `Wohlfarth_Arzneimittel.txt` eine absolute Häufigkeitstabelle (`table("ATC"=arznei$ATC)`) der ATC-Codes erstellt und entsprechend sortiert (`order()`). Ab der Zeile 16 beginnen die üblichen `R`-Befehle für die Grafikausgabe. Der zusätzliche Befehl `lines()` in Zeile 34 und 35 erzeugt die roten Markierungslinien an der speziellen X-Position (`x=`). Der Parameter `y=` gibt die Höhe der Linien an und `type=` bestimmt die Art der Kennzeichnung. Die Codierung durch den Buchstaben `h` definiert eine vertikale Linie. Mit dem letzten Funktionsaufruf (`write.table()`) in Zeile 39 wird eine Liste aller ATC-Codes mit ihren absteigend sortierten Häufigkeiten als CSV-Datei auf die Festplatte gespeichert.

Die R-Befehle der Abbildung 5.32:

```
# Variable setzen (Anzahl der häufigsten ATC-Codes)
# default: 30
anzahl_atc_codes <- 30
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Arzneidaten einlesen
arznei <- arzneimittel_laden()
# Absolute Häufigkeitstabelle der ATC-Codes erstellen
arznei_absh <- data.frame(table("ATC"=arznei$ATC))
# Häufigkeitstabelle absteigend sortieren
arznei_absh <- data.frame(arznei_absh[order(-arznei_absh$Freq),])
# Die häufigsten ATC-Codes extrahieren
atc_bereich <- arznei_absh[1:anzahl_atc_codes,]
### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(6,7,3,2))
# Plotten
plot <- barplot(height=atc_bereich[,2],main=paste("Die häufigsten ",anzahl_
    atc_codes," ATC-Codes",sep=""),axes=FALSE,axisnames=FALSE,col=farben[4],
    ylim=c(0,1.1*max(atc_bereich[,2])))
# Legende erzeugen
  # Wirkstoffe
  wirkstoff <- c("Metoprolol","Ramipril","Pantoprazol","Ibuprofen","
      Simvastatin","Torasemid","Levothyroxin","Amlodipin","Metamizol","
      Bisoprolol","Metformin","Diclofenac","Omeprazol","Allopurinol","
      Acetylsalicylsäure")
  tabelle_atc <- data.frame("ATC_Code"=atc_bereich[1:15,1],"Wirkstoff"=
      wirkstoff)
  # Relative Häufigkeit berechnen
  rel <- atc_bereich$Freq[1:15] * 100 / length(arznei$ATC)
  rel <- round(x=rel,digits=2)
  rel_legende <- paste(atc_bereich[1:15,1],"  (",format(x=rel,nsmall=2),"%","
      )  ",sep="",wirkstoff)
legend('topright',legend=rel_legende,title="ATC-Code | rel. Häufigkeit |
    Wirkstoff",cex=0.7)
# Anzahl jeder Säule anzeigen
text(x=plot,y=(atc_bereich[,2]+18000),srt=90,labels=format(x=atc_bereich[,2],
    big.mark=".",scientific=FALSE),cex=0.7)
# ATC Codes jeder Säule anzeigen
text(x=plot,y=-22000,labels=atc_bereich[,1],cex=0.6,srt=90,xpd=TRUE)
# Y-Achse definieren
ybereich <- pretty(x=c(0,max(arznei_absh[,2])),n=10)
# Y-Achse anzeigen
```

```
35 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=
       FALSE),las=2,cex.axis=0.9,pos=-0.6)
36 # Y-Achsen-Beschriftung
37 mtext(text="Anzahl",side=2,line=4,cex=0.9)
38 # X-Achsen-Beschriftung
39 mtext(text="ATC Codes",side=1,line=4,cex=0.9)
```

Die ersten dreizehn Zeilen berechnen, ähnlich wie im Quellcode der Abbildung 5.31, die Häufigkeiten der ATC-Codes, mit dem Unterschied, dass in Zeile 3 die Anzahl der darzustellenden Codes festgelegt wird. Ab der Zeile 14 bis einschließlich 39 werden die `plot()`-, `legend()`-, `text()`-, `mtext()`- und `axis()`-Funktionen für die grafische Ausgabe eingesetzt.

5.6.2. Regionale Verteilung der ATC-Codes in Sachsen-Anhalt

Ergänzend zu der Ausarbeitung der Abbildung 5.31 bezieht sich die nachfolgende Untersuchung auf die regionale Verteilung der ATC-Codes. Es soll festgestellt werden, inwieweit die Medikationen homogen verteilt sind, bzw. inwieweit lokale Schwerpunkte bei den Verordnungen auftreten.
Für diesen Zweck wird wiederum die Heatmapdarstellung auf Basis der Postleitzahlengebiete herangezogen, sodass detaillierte Aussagen getroffen werden können. Im Gegensatz zu der Analyse im Kapitel 5.6.1 werden nur die Verschreibungen der Versicherten betrachtet, die im Bundesland Sachsen-Anhalt ansässig sind. In Folge dessen reduziert sich der Datenbestand von ca. 9,28 Mio. um ca. 300.000 ATC-Codes.
Für die Analyse werden zwei Arten der grafischen Darstellung gewählt. Die erste Abbildung (5.33) beruht auf einer absoluten Betrachtung, d.h. es wird die Anzahl der ATC-Codes des jeweiligen Postleitzahlgebietes auf der Heatmap präsentiert. Die zweite Abbildung (5.34) stellt die durchschnittliche Anzahl der ATC-Codes pro Versicherten dar. Dazu werden die Ergebniswerte aus der Abbildung 5.33 (Anzahl ATC-Codes pro PLZ-Gebiet) durch die Werte der Abbildung 5.1 (Anzahl Versicherter pro PLZ-Gebiet), siehe Seite 43, dividiert.

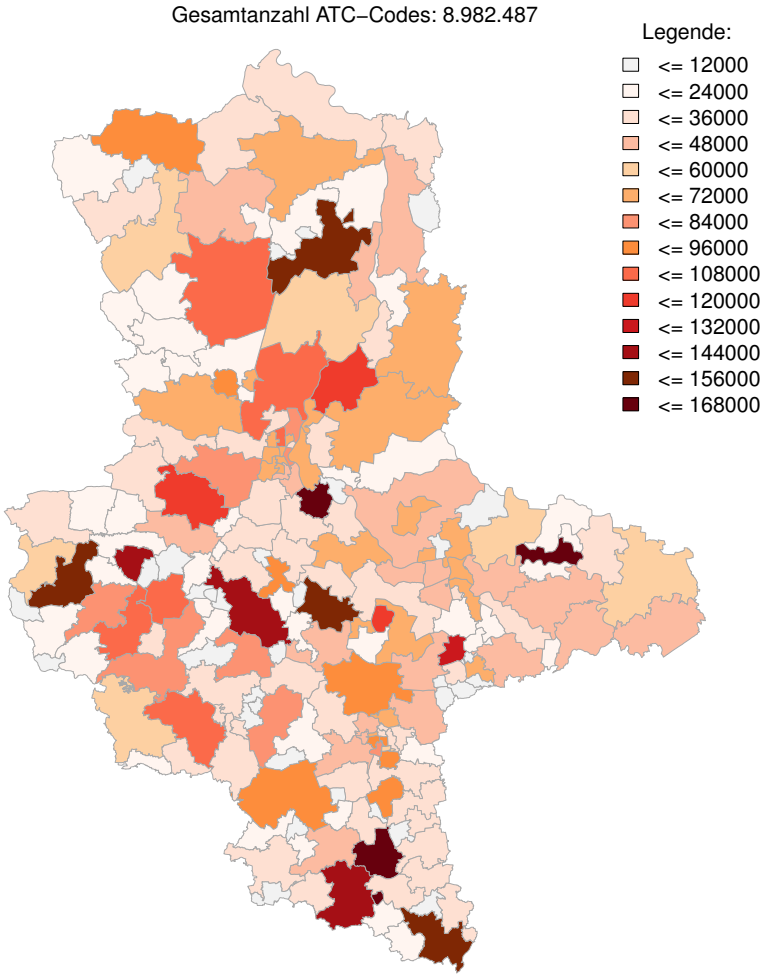

Abb. 5.33.: Absolute Verteilung der ATC-Codes

Erwartungsgemäß variiert die Anzahl der verordneten Medikationen bei der absoluten Betrachtung aufgrund der unterschiedlichen Bevölkerungsdichte pro Postleitzahlgebiet (siehe S. 43) erheblich. Mit 1.024 Verschreibungen hat die Region mit der Postleitzahl 06255, in der gleichzeitig die wenigsten Versicherten ansässig sind, den geringsten Anteil an der Gesamtanzahl. Im Gegenzug vereint die Postleitzahl 39218 (Schönebeck (Elbe), Salzlandkreis) die meisten ATC-Codes mit 167.299, dicht gefolgt von einem Stadtteil in Wittenberg, in dem die meisten Versicherten leben (vgl. S. 43), mit 163.107 ATC-Codes.

Die kumulierte Gesamtanzahl an Verordnungen im Jahr 2012 beträgt 8.982.487 und verteilt sich auf die in Sachsen-Anhalt lebenden Versicherten. Bei insgesamt 580.529 Versicherten ergibt sich im landesweiten Durchschnitt ein Wert von 15,47 Medikationen pro Versicherungsnehmer. Eine genauere Betrachtung folgt in Abbildung 5.34.

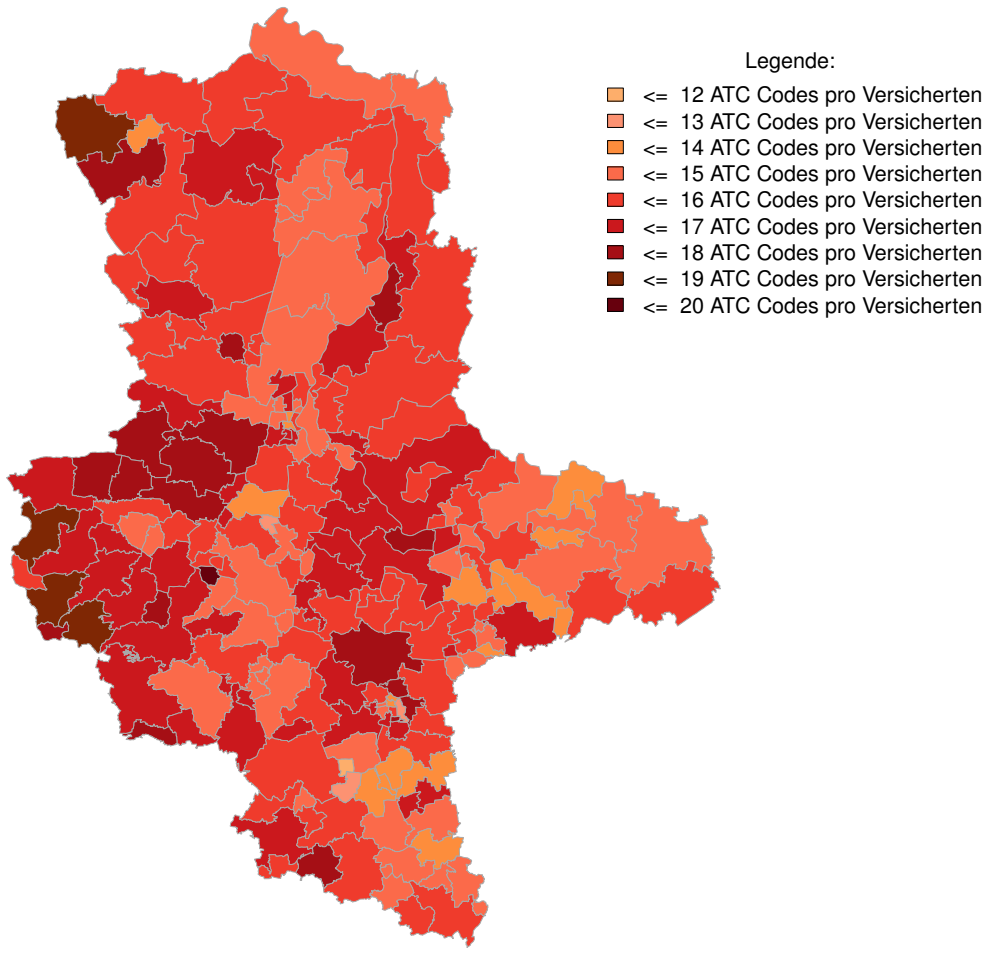

Abb. 5.34.: Anzahl ATC-Codes pro Versicherten

Die Karte lässt erkennen, dass im Harz und dessen nördlich angrenzenden Gebieten, sowie in einzelnen Gemeinden des Landkreises Altmarkkreis-Salzwedel mehr Medikationen verschrieben werden als in anderen Regionen. Beispielsweise liegt der Ortsteil

Hoym in der Stadt Seeland im Salzlandkreis (Postleitzahl 06467) mit 19.5 und der Gemeindeverbund *Beetzendorf-Diesdorf* (Postleitzahl 29413) mit 18.9 ATC-Codes pro Versicherten deutlich oberhalb des Gesamtdurchschnitts. Das Gebiet mit den niedrigsten Verschreibungen pro Versicherungsnehmer ist wiederum der Ort *Wünsch* (Saalekreis) mit der Postleitzahl 06255 aus obiger Untersuchung. Für die dort lebenden 93 AOK-Versicherten wurden 1024 Verordnungen ausgestellt, sodass sich für diesen Postleitzahlenbezirk 11 ATC-Codes pro Versicherten ergeben.

Der für die Abbildungen 5.33 und 5.34 zuständige Quelltext besitzt zahlreiche Überschneidungen, wodurch das erste Code-Segment für beide Grafiken genutzt wird.

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")
# Arzneidaten einlesen
arznei <- arzneimittel_laden()
# Die Spalten "PSEUDONYM", "ATC" extrahieren
arznei_atc <- arznei[c(1,4)]
# Verbinden aller Landkreise mit den ATC-Codes der arznei_daten
all_atc <- merge(alle_kreise,arznei_atc,by="PSEUDONYM")
# Absolute Häufigkeitstabelle erzeugen (nach den PLZ)
all_atc_freq <- data.frame(table(all_atc$PLZ))
colnames(all_atc_freq) <- c("PLZ","Freq")
# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
all_atc_freq$PLZ <- str_pad(string=all_atc_freq$PLZ,width=5,pad="0")
# Data.frame für die Darstellung (maptools) erzeugen
anzahl_atc_lk <- data.frame(shapes$plz)
anzahl_atc_lk <- cbind(anzahl_atc_lk,0)
# Spaltenüberschriften vergeben
colnames(anzahl_atc_lk) <- c("PLZ","Freq")
# Anzahl der vorkommenden ATC-Codes in data.frame "anzahl_atc_lk" schreiben
for (i in 1:length(anzahl_atc_lk$PLZ)) {tmp <- subset(x=all_atc_freq$Freq,
    subset=(all_atc_freq$PLZ == anzahl_atc_lk[i,1]))
  if (length(tmp) == 0) {
    anzahl_atc_lk[i,2] <- 0
  }
  else {
    anzahl_atc_lk[i,2] <- tmp
  }
}
```

Beginnend mit dem Einlesen des Datenmaterials in R (Zeile 1-8) werden in Zeile 10-12 die ATC-Codes den Versicherten zugeordnet (`merge()`-Befehl) und eine Häufigkeitstabelle der vorkommenden ATC-Codes, gruppiert nach Postleitzahlen, erstellt (`table(all_atc$PLZ)`). Nach anschließender Beschriftung und Konvertierung der Postleitzahlen in ein 5-stelliges Format (Zeile 13-15) wird ein neuer *data.frame* `anzahl_atc_lk`, der die benötigte Reihenfolge der Postleitzahlen aus dem *shapefile* besitzt, mit Nullen initialisiert (Zeile 17-18) und mit den entsprechenden Häufigkeitswerten in der `for()`-Schleife gefüllt (siehe Zeile 22-29).

Der verbleibende R-Code für die Abbildung 5.33 begrenzt sich auf die Grafikausgabe. Der komplette Quelltext befindet sich ebenfalls im Quellcode-Anhang auf Seite 235.

```
1  # Einteilung der Schritte für die Heatmap
2  break_points <- seq(from=0,to=168000,by=12000)
3  class <- cut(anzahl_atc_lk$Freq,breaks=break_points)
4  levels(class) <- break_points[2:length(break_points)]
5  ### PLOT BEGINNT ###
6  # Plotten
7  plot(x=shapes,border="darkgrey",col=farben_shape[class],main="Absolute
       Verteilung aller ATC Codes")
8  # Legende anzeigen
9  legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
       black",cex=1.2,title="Legende:",bty="n")
10 # Gesamtanzahl an ATC-Codes anzeigen
11 mtext(text=paste("Gesamtanzahl ATC-Codes: ",sum(all_atc_freq$Freq),sep=""),
       side=1,cex=0.7,line=1)
```

Von Zeile 2 bis 4 werden die Farbbereiche der Heatmap festgelegt. Die Einteilung beginnt mit 0 ATC-Codes pro Postleitzahlgebiet und durch die feste Schrittweite von 12.000 endet der Bereich bei 168.000. Die Zeilen 7-11 erzeugen mit den bekannten R-Befehlen `plot()`, `legend()` und `mtext()` die Grafikausgabe.

Der nachfolgende Code zeigt für die Darstellung der Abbildung 5.34 zusätzlich benötigte Befehle. (Siehe dazu auch Seite 237 im Anhang).

```
1  ### Anzahl Einwohner pro PLZ ermitteln
2  # Data.frame für die Darstellung (maptools) erzeugen
3  anzahl_ver_lk <- data.frame(shapes$plz)
4  anzahl_ver_lk <- cbind(anzahl_ver_lk,0)
```

```
5  colnames(anzahl_ver_lk) <- c("PLZ","Freq")
6  # Absolute Häufigkeitstabelle der Versicherten erzeugen (nach den PLZ)
7  all_ver_freq <- data.frame(table(alle_kreise$PLZ))
8  colnames(all_ver_freq) <- c("PLZ","Freq")
9  # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
10 all_ver_freq$PLZ <- str_pad(string=all_ver_freq$PLZ,width=5,pad="0")
11 # Anzahl der Versicherten in den data.frame "anzahl_ver_lk" schreiben
12 for (i in 1:length(anzahl_ver_lk$PLZ)) {tmp <- subset(x=all_ver_freq$Freq,
       subset=(all_ver_freq$PLZ == anzahl_ver_lk[i,1]))
13   if (length(tmp) == 0) {
14     anzahl_ver_lk[i,2] <- 0
15   }
16   else {
17     anzahl_ver_lk[i,2] <- tmp
18   }
19 }
20 # Anzahl ATC-Codes durch die Anzahl der Versicherten dividieren (1 ATC-Code
       pro Versicherten)
21 plot_pro_patient <- data.frame("PLZ"=anzahl_atc_lk$PLZ,"pro_patient"=0)
22 for (i in 1:length(anzahl_ver_lk$PLZ)) {plot_pro_patient$pro_patient[i] <-
       anzahl_atc_lk$Freq[i] / anzahl_ver_lk$Freq[i]}
23 # Einteilung der Schritte für die Heatmap
24 break_points <- c(0,12:20)
25 class <- cut(plot_pro_patient$pro_patient,breaks=break_points)
26 levels(class) <- break_points[2:length(break_points)]
27 # Nur 9 statt der 14 Farben nötig
28 farben_shape_9 <- farben_shape[6:14]
29 ### PLOT BEGINNT ###
30 # Plotten
31 plot(x=shapes,border="darkgrey",col=farben_shape_9[class],main="ATC Codes pro
        Versicherten")
32 # Legende anzeigen
33 legend('topright',fill=farben_shape_9,legend=paste("<= ",levels(class),"ATC
       Codes pro Versicherten"),border="black",cex=0.8,title="Legende:",bty="n")
```

Basierend auf dem Quellcode der Seite 135 ist es für die Darstellung der durchschnittlichen ATC-Codes pro Versicherten erforderlich, die Gesamtanzahl ansässiger AOK-Versicherter pro Postleitzahlgebiet zu ermitteln. Dazu wird in Zeile 3-19 ein neuer *data.frame* `anzahl_ver_lk` angelegt, dessen Erzeugung identisch mit dem Vorgehen des *data.frame* `anzahl_atc_lk` ist (vgl. S. 135), jedoch keine ATC-Codes, sondern die Anzahl Versicherter pro Postleitzahlgebiet beinhaltet. Nachfolgende drei Zeilen erzeugen den für die Grafikausgabe notwendigen *data.frame* `plot_pro_patient`.

Dieser enthält durch die Division von `anzahl_atc_lk` und `anzahl_ver_lk` das gewünschte Ergebnis. Die restlichen Befehle ab Zeile 24 sind für die Darstellung der Heatmap verantwortlich.

5.6.3. Die PRISCUS-Liste

Die sog. *PRISCUS*-Liste ist ein im Forschungsprojekt des Projektverbandes *PRISCUS* erstelltes Dokument, in dem potenziell inadäquate Medikationen für ältere Menschen (ab 65 Jahren) aufgelistet sind. Ziel des Projektes ist es, die Arzneimittelsicherheit zu steigern, indem u.a. günstige und ungünstige Arzneimittelkombinationen sowie geeignete Dosierungsmengen von Medikationen für Menschen höheren Alters erforscht und optimiert werden. Zudem wird versucht, neue Behandlungsansätze für ältere Patienten zu finden, bei denen mehrere Erkrankungen gleichzeitig und zusammenhängend auftreten (Multimorbidität) [20].
Die dieser Arbeit zugrunde liegende Liste wurde 2011 veröffentlicht und beinhaltet insgesamt 83 Wirkstoffe aus 18 Arzneistoffklassen, bei denen ein erhöhtes Risiko an unerwünschten Nebenwirkungen auftreten kann. Weiterhin enthält die Liste Maßnahmen für die “klinische Praxis“ und Therapiealternativen der *PRISCUS*-Medikationen [14].

Ziel dieser Untersuchung ist die Ermittlung der betroffenen Personen, denen *PRISCUS*-Medikamente verordnet wurden. Dabei wird die Anzahl der Betroffenen pro einzelner Region, sowie die Gesamtanzahl für das Bundesland Sachsen-Anhalt bestimmt. Mehrfachverschreibungen der Präparate an einzelne Versicherte bleiben unberücksichtigt.
Die Ergebnisse können als Diskussionsgrundlage dienen und in weiteren Schritten zu Therapieverbesserungen und zu einer Senkung der unerwünschten Arzneimittelereignisse (UAE) zugunsten der Patienten, beitragen. Verbunden damit resultiert für den Versicherungsträger eine eventuelle Kosteneinsparung.
Für die Analyse müssen die 83 Arzneistoffe mit ihren entsprechenden ATC-Codes vorliegen, um eine Verbindung zum Datenbestand zu ermöglichen. Die Zuordnung geschieht über die zur Verfügung gestellte Liste `Priscus-Liste_Joerg.xlsx` (siehe S. 33), in der alle Wirkstoffe in ihre 97 ATC-Codes aufgeschlüsselt sind. Die Tatsache, dass einige Stoffe mehrere ATC-Codes besitzen, führt zu einer unterschiedlichen Anzahl.

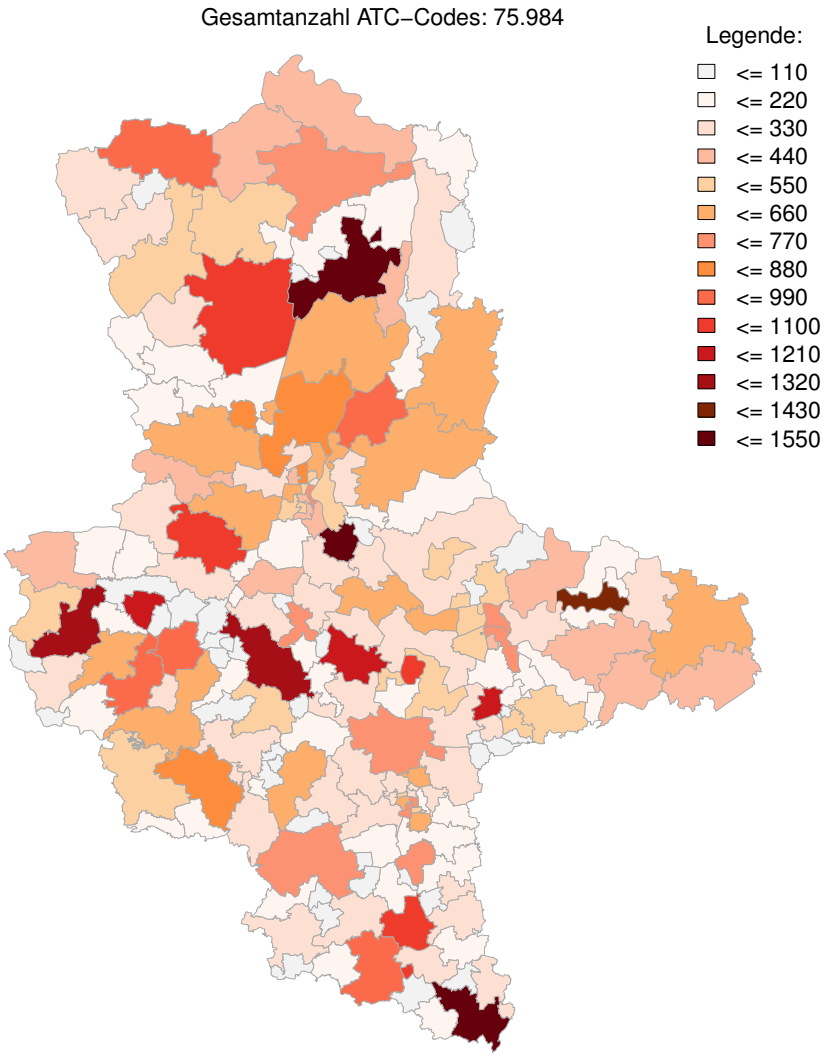

Abb. 5.35.: ATC-Codes mit potenziell inadäquaten Medikationen

Die Abbildung 5.35 stellt die absoluten Häufigkeiten der indizierten ATC-Codes als farbcodierte Karte dar. Die Anzahl vorkommender Codes pro Postleitzahlgebiet bewegt sich zwischen 7 im Postleitzahlgebiet 06255 und 1.547 im Postleitzahlbereich 39576, Stendal. Der Gesamtdurchschnitt beläuft sich auf 376 Codes pro Gebiet und die Gesamtanzahl für Sachsen-Anhalt beträgt 75.984 Codes, wobei diese auf insgesamt

61.619 Versicherte entfallen. Bei einer Anzahl von 256.770 Versicherungsnehmern, die im Jahr 2012 älter als 65 waren, erhielten demnach ca. 24% der AOK-Versicherten, eine nach den Kriterien der *PRISCUS*-Liste potentiell inadäquate Medikation.

Die obige Heatmap ist aus folgendem Quelltext entstanden:

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Priscus-Liste einlesen
priscus_csv <- priscus_laden()
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Arzneidaten einlesen
arznei <- arzneimittel_laden()
# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden("plz")
# Data.frame aus den ATC-Codes erstellen
priscus_liste <- data.frame("ATC"=priscus_csv$ATC)
# Die Spalten "PSEUDONYM", "ALTER" werden extrahiert
ambu_age <- ambudaten[c(1,6)]
# Die Spalten "PSEUDONYM", "ATC" werden extrahiert
arznei_atc <- arznei[c(1,4)]
# In den Arzneidaten ATCs raussuchen, die in der Priscus_liste vorkommen
# Unique() anwenden, da ein Versicherter pro Jahr ein Medikament mehrmals
    verschrieben bekommen könnte
priscus_atc <- unique(arznei_atc[arznei_atc$ATC %in% priscus_liste$ATC,])
# In den Ambudaten die Versicherten raussuchen, die > 65 Jahre
# Unique() anwenden, da ein Versicherter in den Ambudaten mehrmals vorkommt
priscus_age <- unique(subset(x=ambu_age,subset=(ambu_age$AGE > 65)))
# priscus_atc  und  priscus_age  verbinden
age_atc <- merge(priscus_atc,priscus_age,by="PSEUDONYM")
### HEATMAP BERECHNUNG BEGINNT ###
alle_kreise_priscus <- unique(merge(alle_kreise,age_atc,by="PSEUDONYM"))
# Data.frame für die Darstellung (maptools) erzeugen
plz_priscus <- data.frame(shapes$plz)
plz_priscus <- cbind(plz_priscus,0)
# Spaltenüberschriften vergeben
colnames(plz_priscus) <- c("PLZ","Freq")
# Häufigkeitstabelle über alle vorkommenden PLZ der Landkreise
plz_lk <- data.frame(table(alle_kreise_priscus$PLZ))
# Spaltenüberschriften vergeben
colnames(plz_lk) <- c("PLZ","Freq")
# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
```

```
37 plz_lk$PLZ <- str_pad(string=plz_lk$PLZ,width=5,pad="0")
38 # Anzahl der vorkommenden PLZ der Landkreise in den data.frame "plz_priscus"
       schreiben
39 for (i in 1:length(plz_priscus$PLZ)) {tmp <- subset(x=plz_lk$Freq,subset=(plz
       _lk$PLZ == plz_priscus[i,1]))
40   if (length(tmp) == 0) {
41     plz_priscus[i,2] <- 0
42   }
43   else {
44     plz_priscus[i,2] <- tmp
45   }
46 }
47 # Einteilung der Schritte für die Heatmap
48 break_points <- c
       (0,110,220,330,440,550,660,770,880,990,1100,1210,1320,1430,1550)
49 class <- cut(plz_priscus$Freq,breaks=break_points)
50 levels(class) <- break_points[2:length(break_points)]
51 ### PLOT BEGINNT ###
52 # Plotten
53 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="PRISCUS")
54 # Gesamtanzahl an ATC-Codes anzeigen
55 mtext(text=paste("Gesamtanzahl ATC-Codes: ",format(x=length(alle_kreise_
       priscus$ATC),big.mark=".",scientific=FALSE),sep=""),side=1,cex=0.7,line
       =1)
56 # Legende anzeigen
57 legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
       black",cex=0.8,title="Legende:",bty="n")
58 ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit der ATC-
       Codes der PRISCUS-Liste)
59 # Tabelle beinhaltet gesamten Datenbestand (alle_kreise und alle_anderen)
60 tabelle_priscus <- data.frame(table("ATC"=age_atc$ATC))
61 tabelle_priscus <- rbind(tabelle_priscus,c(NA,sum(tabelle_priscus$Freq)))
62 write.table(x=tabelle_priscus,file="/Users/Florian/Documents/Bachelorarbeit/
       CSV/priscus_atc_haeufigkeit.csv",sep=";",row.names=FALSE,col.names=TRUE,
       fileEncoding="latin1")
```

Ab Zeile 1 bis einschließlich 16 werden die erforderlichen Dateien geladen. Dazu zählen die `initial.R`, `Priscus-Liste_Joerg.csv`, `Wohlfahrt_ambuDaten.txt`, `Wohlfarth_Arzneimittel.txt` und die `sa_plz.shp` Datei. Daran anknüpfend werden die übereinstimmenden ATC-Codes der *PRISCUS*-Liste aus der Arzneimittel-Datei (`arznei_atc`) extrahiert und mit dem gefilterten ambulanten Datensatz (Versicherte ab 65 Jahre) verbunden (siehe Zeile 19-24). Ausgehend von diesem Ergebnis wird in Zeile

26 eine Zuordnung der Versicherten und den Postleitzahlen mithilfe des *data.frame* `alle_kreise` hergestellt. In den Zeilen 28 bis 57 wird, ähnlich wie in den vorherigen Grafiken, eine Häufigkeitstabelle der ATC-Codes erstellt, den Postleitzahlen zugewiesen und als Heatmap dargestellt. Der abschließende Befehl `write.table()` erzeugt eine im CSV-Format absteigend sortierte Liste der ATC-Codes und die Anzahl der von diesem Code betroffenen Personen. Von den 97 ATC-Codes in der *PRISCUS*-Liste sind 78 Codes in Sachsen-Anhalt wiederzufinden und in Tabelle 5.8 aufgelistet.

ATC-Code	Anzahl	ATC-Code	Anzahl	ATC-Code	Anzahl	ATC-Code	Anzahl	ATC-Code	Anzahl
A06AA01	33	D04AA13	19	M03BX01	861	N05BA04	312	N05CF01	1067
B01AC05	114	G04BD04	958	M03BX07	4275	N05BA05	21	N05CF02	1664
B01AC22	199	G04BD07	375	N02AB02	69	N05BA06	3835	N06AA02	165
C01AA02	329	G04BD08	2697	N02CA02	3	N05BA08	287	N06AA04	277
C01AA05	186	G04CA03	884	N03AA02	92	N05BA09	32	N06AA06	2189
C01AA08	36	J01XE01	2226	N04BC03	27	N05BA11	1	N06AA09	5344
C01BC04	385	M01AA01	112	N05AA02	476	N05BA12	1031	N06AA12	2181
C02AB01	172	M01AB01	2643	N05AB02	152	N05BB01	305	N06AA21	89
C02AC01	618	M01AB11	908	N05AB03	21	N05CC01	75	N06AB03	311
C02CA01	3	M01AC01	492	N05AC02	41	N05CD01	17	N06AF04	8
C02CA04	2932	M01AC06	1551	N05AD01	16	N05CD02	784	N06BX03	1608
C02LA01	107	M01AE03	190	N05AH02	258	N05CD03	45	R06AA04	174
C04AD03	391	M01AE17	1485	N05AH03	781	N05CD05	6	R06AB03	553
C04AX21	1524	M01AH05	11612	N05BA01	3	N05CD06	180	S01EA04	1532
C07AA07	944	M02AA10	4	N05BA02	341	N05CD07	28		
C08CA05	5208	M02AA23	3	N05BA03	1760	N05CD09	322		

Tab. 5.8.: Häufigkeit der *PRISCUS*-Wirkstoffe

5.7. Arznei- und Hilfsmittelverschreibungen in Sachsen-Anhalt

In Deutschland ist die Aufnahme eines neuen Arznei- bzw. Hilfsmittels mit der Vergabe eines Identifikationsschlüssels durch die *Informationsstelle für Arzneispezialitäten* (IFA) verbunden. Dieser Schlüssel, auch Pharmazentralnummer (PZN) genannt, ist ein einheitlicher 7 bzw. 8-stelliger numerischer Code, der wie bereits auf Seite 29 erläutert,

Produkte nach ihrer Packungsgröße, Darreichungsform, Farbe, Form und Größe eines bestimmten Anbieters unterscheidet [16].

Die Codierung der PZN-Codes ist eindeutig, erfolgt maschinell und folgt keiner logischen Struktur hinsichtlich der Artikelzuordnung. D.h., der Code beinhaltet keine Informationen über den Artikel, wodurch eine direkte Aufschlüsselung, wie bei den ATC- bzw. ICD-10-Codes, nicht möglich ist [15]. Eine weitere wichtige Funktion der PZN besteht in der Aufgabe einer einheitlichen Abrechnung zwischen den Apotheken und den Krankenkassen.

In diesem Abschnitt werden mit Hilfe der PZN-Codes kostenbasierende Analysen vorgestellt. Die Grundlage für die Untersuchungen bildet die auf Seite 32 beschriebene Datei `pzn-kosten-tabelle.xlsx`.

5.7.1. Packungspreise und Gesamtkosten pro Arzneimittel

Die erste Aufgabe besteht darin, die Menge der verordneten Präparate dem jeweiligen Einzelpreis gegenüberzustellen. Dadurch soll die Vermutung bestätigt bzw. widerlegt werden, dass häufig verordnete Artikel wesentlich kostengünstiger gehandelt werden als seltene Verschreibungen.
Für dieses Vorhaben wird die Arzneimittel-Datei herangezogen und auf Basis des gesamten Datenbestandes eine absteigende Häufigkeitstabelle der PZN-Codes erstellt. Anschließend wird ein Abgleich mit der Kostentabelle vorgenommen und das Ergebnis in Abbildung 5.36 dargestellt.

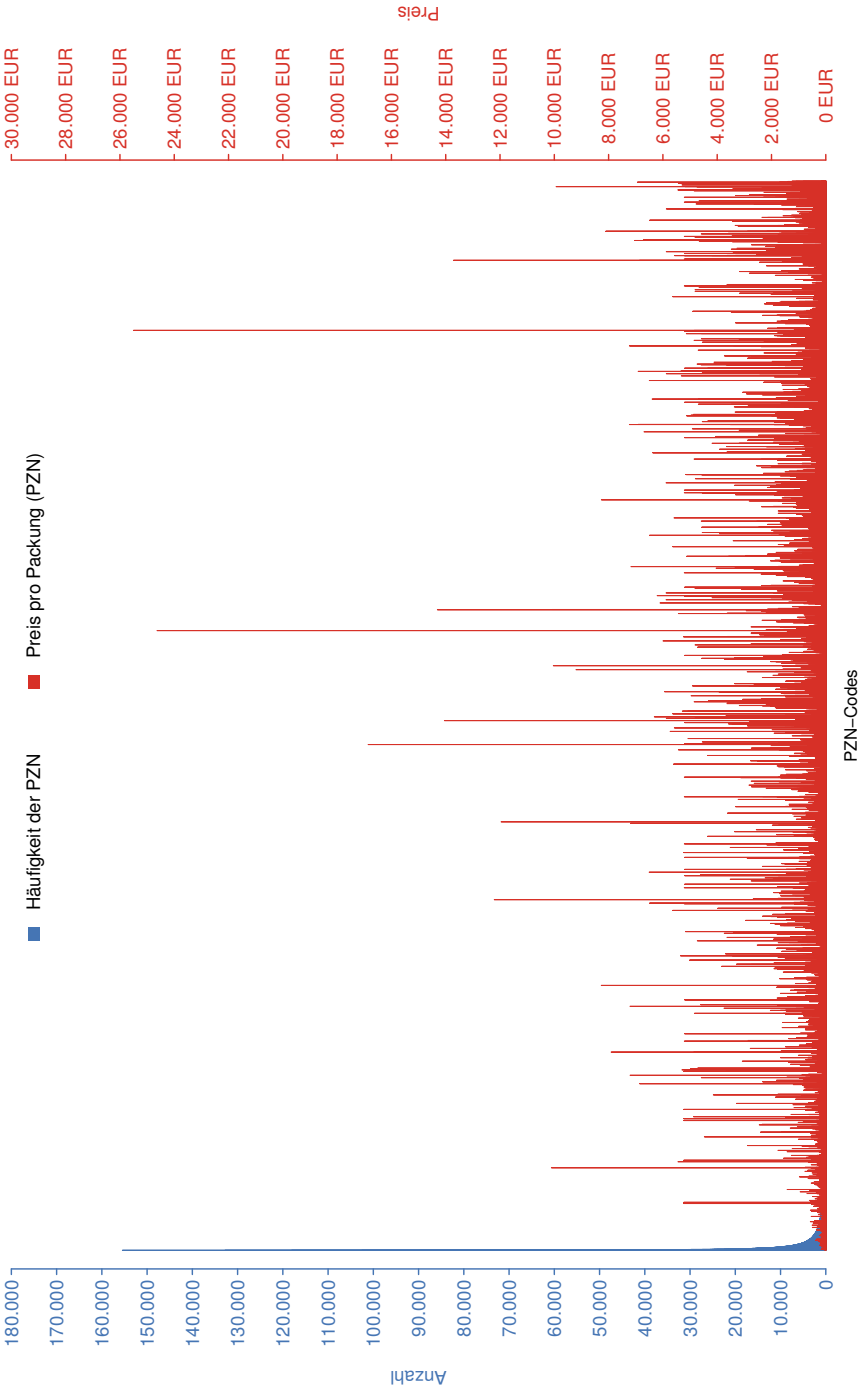

Abb. 5.36.: Packungspreise pro PZN-Code

Die Abbildung 5.36 zeigt zwei überlagerte Graphen an, wobei die X-Achse (PZN-Codes) die Relation zwischen der Anzahl (blaue Y-Achse) und dem Packungspreis (rote Y-Achse) bildet. Von den in der Arzneimittel-Datei verschiedenen vorhandenen 40.528 PZN-Codes ist zu annähernd jedem Code ein Preis verfügbar, sodass auf der Grafik insgesamt 40.120 PZN-Codes dargestellt sind. Für die Differenz von 408 PZN-Codes ist entweder kein Preis verfügbar oder es existiert keine Verordnung dieses Handelspräparates für das Jahr 2012.
Beginnend mit dem am häufigsten verordneten Artikel von 155.475 Stück (*RAMILICH 5MG Tabletten - 100 Stk. - PZN: 1983648 - Packungspreis: 13,63 €*) zeigt die Verteilung einen stark abfallenden Verlauf. Die Preise der 5.598 Präparate, die einmalig verschrieben wurden, variieren zwischen 1,24 € und 13.720,00 €.
Im Gegensatz dazu weist der Graph der Packungspreise einen sehr sprunghaften Verlauf auf. Hier schwanken die Preise zwischen 0,56 € und 25.504,20 € pro Packung.
Betrachtet man den Durchschnittspreis der häufigsten 500 Präparate (38,61 €) liegt dieser im Vergleich zu den Restlichen mit durchschnittlich 141,25 € um das ca. 3,7-fache niedriger, wodurch bei gesamtheitlicher Betrachtung in Verbindung mit der Grafik, die obere Aussage bestätigt wird.

Als Erweiterung der ersten Analyse, stellt die zweite Abbildung (5.37) die Gesamtkosten bezüglich der Häufigkeit pro PZN-Code dar. Im Gegensatz zu der Abbildung 5.36 wird erkenntlich, dass die höchsten Ausgaben bei den niedrigpreisigen Präparaten vorkommen und die teureren Produkte weit weniger ins Gewicht fallen. Der Balkenverlauf der Gesamtkosten ist im Vergleich des Verlaufs der Packungspreise, unter Vernachlässigung einiger Ausreißer, stetig fallend. Beispielsweise haben die häufigsten 500 Artikel einen Anteil von ca. 35% mit 156.016.257,- € an den Gesamtkosten von 445.963.321,34 €.
Das oben erwähnte Medikament *RAMILICH*, welches Kosten von 2.119.124,25 € verursacht, ist, wie in der Grafik erkennbar, nicht das kostenintensivste Produkt, sondern der Artikel *SPIRIVA 18UG (Kapseln - 90 Stk. - PZN: 03649221 - Packungspreis: 178,15 €)* mit 4.293.236,85 €, gefolgt von *HUMIRA 40MG PEN (6 Stk. - PZN: 00281795 - Packungspreis: 5.230,87 €)* mit Gesamtkosten in Höhe von 4.069.616,86 €.

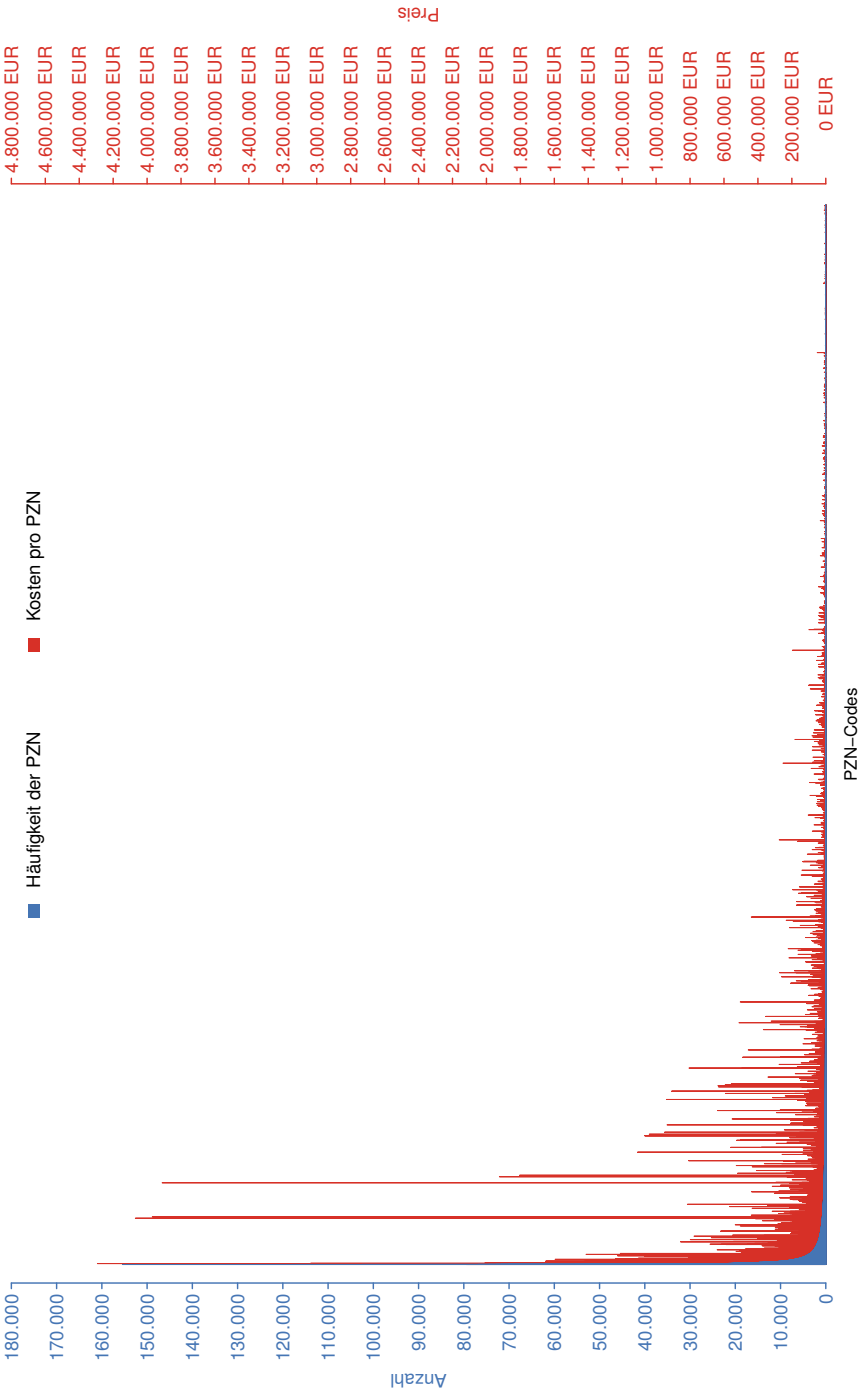

Abb. 5.37.: Gesamtkosten pro PZN-Code

Im Folgenden ist der Quelltext für die Abbildung 5.36 aufgelistet.

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Arzneidaten einlesen
arznei <- arzneimittel_laden()
# PZN_Preise einlesen
pzn_preise <- pznkosten_laden()
colnames(pzn_preise) <- c("PZN","Preis")
# Absolute Häufigkeitstabelle der PZN-Codes erstellen
pzn_absh <- data.frame(table("PZN"=arznei$PZN))
# Verbinden der PZN-Codes mit den Preisen
pzn_absh_preise <- merge(pzn_absh,pzn_preise,by="PZN")
# PZN-Codes absteigend nach Häufigkeit sortieren
pzn_absh_preise <- pzn_absh_preise[order(-pzn_absh_preise$Freq),]
## PLOT BEGINNT ###
# Seitenabstand
par(mar=c(3,7,5,7))
# Y-Achsen Markierungen
ybereich_1 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Freq)),n=20)
ybereich_2 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Preis)),n=20)
# Plotten
barplot(height=pzn_absh_preise$Freq,main="Packungspreise pro PZN-Code (nach
    Häufigkeit der PZN-Codes)",ylim=c(0,max(ybereich_1)),axes=FALSE,col=
    farben_2[12],border=farben_2[12],space=0)
# Y-Achse_1 anzeigen
axis(side=2,at=ybereich_1,labels=format(x=ybereich_1,big.mark=".",scientific=
    FALSE),col=farben_2[12],col.axis=farben_2[12],las=2,cex.axis=0.75)
# Y-Achse_1 Überschrift
mtext(text="Anzahl",side=2,line=4,col=farben_2[12],cex=0.7)
# Beide Y-Achsen gleich hoch
par(new=TRUE)
# Plotten
barplot(height=pzn_absh_preise$Preis,ylim=c(0,max(ybereich_2)),axes=FALSE,col
    =farben_2[2],border=farben_2[2],space=0)
# Y-Achse_2 anzeigen
axis(side=4,at=ybereich_2,labels=paste(format(x=ybereich_2,big.mark=".",
    scientific=FALSE),"EUR"),col=farben_2[2],col.axis=farben_2[2],las=2,cex.
    axis=0.75)
# Y-Achse_2 Überschrift
mtext(text="Preis",side=4,line=5,col=farben_2[2],cex=0.8)
# Legende anzeigen
legend(x=2000,y=29000,legend=c("Häufigkeit der PZN","Preis pro Packung (PZN)"
    ),fill=c(farben_2[12],farben_2[2]),cex=0.7,xpd=TRUE,bty="n",border=NA,
    horiz=TRUE)
```

```
36  # X-Achsen Überschrift
37  mtext(text="PZN-Codes",side=1,line=1,col="black",cex=0.7)
```

Zu Beginn wird die Arzneimitteldatei und die Kostentabelle eingelesen (Zeile 4-7) und auf Basis der verordneten PZN-Codes eine Häufigkeitstabelle im *data.frame* `pzn_absh` angelegt (Zeile 9). In der nachfolgenden Zeile werden den einzelnen PZN-Codes ihre Packungspreise zugewiesen, indem der *data.frame* `pzn_absh` mit den `pzn_kosten` verbunden wird (`merge()`-Funktion). Um eine absteigende Sortierung nach der Häufigkeit zu erhalten, ist in Zeile 13 der `order()`-Befehl notwendig.
Ab Zeile 14 beginnen die Funktionsaufrufe für die Grafik. Dazu werden in Zeile 18-19 die Achsenskalierungen jeweils für das absolute Vorkommen (`ybereich_1`) und den Packungspreisen (`ybereich_2`) der Codes definiert. Die Befehle der Achsen- und Balkenausgabe der Häufigkeiten befinden sich in den Codezeilen 20-25 und für die Ausgabe der Preise sind die Zeilen 28-37 zuständig. Das Anwenden von `par(new=TRUE)` zwischen den `barplot()`-Aufrufen in Zeile 27 bewirkt, dass die Balkendiagramme übereinander und die Skalierung proportional zueinander liegen.

Die Auflistung des Quellcodes für die Abbildung 5.37 beschränkt sich auf die wesentlich abweichenden Codezeilen, da der Programmablauf in den zentralen Teilen vergleichbar mit dem aus der Abbildung 5.36 ist. (Siehe Seite 242 für eine vollständige Auflistung).

```
1  { ... }
2  # Absolute Kosten pro PZN als Spalte hinzufügen
3  pzn_absh_preise <- cbind(pzn_absh_preise,"Kosten_pro_PZN"=pzn_absh_preise$
       Freq * pzn_absh_preise$Preis)
4  { ... }
5  ### Tabelle als CSV auf Festplatte speichern
6  write.table(x=pzn_absh_preise,file="CSV/kosten_pzn.csv",sep=";",row.names=
       FALSE,col.names=TRUE)
```

Um die Kosten pro PZN-Code zu visualisieren, muss in Ergänzung zum obigen Quelltext dem *data.frame* `pzn_absh_preise` eine weitere Spalte hinzugefügt werden. Dazu wird in Zeile 3 die Häufigkeit pro Präparatsverordnung mit den Packungspreisen multipliziert und in der neuen Spalte gespeichert. Am Ende wird der gesamte *data.frame* als Liste auf die Festplatte geschrieben, um Anzahl, Kosten und Packungspreise für jeden PZN-Code zu jeder Zeit einsehen zu können (Zeile 6).

5.7.2. Durchschnittskosten für Arzneimittel pro Postleitzahlgebiet

In einer weiteren Untersuchung wird eine Kostenanalyse unter Einbeziehung der Postleitzahlenbereiche und AOK-Versicherten durchgeführt. Es soll eine Übersicht über die Verteilung der in Anspruch genommenen Leistungen für Arzneiprodukte erstellt und auf diese Weise überprüft werden, ob sich regionale Unterschiede in Sachsen-Anhalt ergeben. Um einen Vergleich der einzelnen Gebiete zu ermöglichen, werden die Gesamtkosten für jede Region ermittelt und in ein Verhältnis zu den dort ansässigen Versicherungsnehmern gesetzt, sodass man die durchschnittlichen Kosten pro Versicherten, bezogen auf das jeweilige Gebiet, erhält.
Die Darstellung der Ergebnisse erfolgt in tabellarischer Form. Ein Ausschnitt des Resultats ist in unterer Tabelle (5.9) zu finden.

PLZ	Kosten pro PLZ	Anzahl Versicherte mit Leistungen	Gesamtanzahl Versicherte	Durchschnittskosten pro Versicherten
06198	2.108.624,13 €	2117	2130	989,96 €
06467	733.039,92 €	760	761	963,26 €
39599	172.935,73 €	178	184	939,87 €
...	...	...	...	...
06406	7.463.100,82 €	9462	9542	782,13 €
39624	2.092.131,87 €	2664	2678	781,23 €
39110	3.253.920,57 €	4135	4175	779,38 €
...	...	...	...	...
06259	152.070,50 €	262	263	578,21 €
06888	710.728,43 €	1230	1246	570,41 €
39217	297.285,17 €	540	543	547,49 €

Tab. 5.9.: Durchschnittskosten pro PLZ-Gebiet

Die Tabelle 5.9 beinhaltet insgesamt 202 Zeilen (Anzahl Postleitzahlen) und gliedert sich in vier Spalten, wobei eine absteigende Sortierung anhand der vierten Spalte *Durchschnittskosten pro Versicherten* erfolgt. Der Auszug lässt eine deutliche Differenz zwischen den höchsten und niedrigsten Durchschnittskosten erkennen. Die Spanne reicht

von ca. 550,- € in Schönebeck-Elbe (Salzlandkreis) bis ca. 990,- € in Salzatal (Saalekreis), welches einem prozentualen Unterschied von 81% entspricht.
Zu einer besseren Veranschaulichung der lokalen Positionen innerhalb des Bundeslandes, sind die Durchschnittskosten pro Postleitzahlgebiet in Abbildung 5.38 in einer Heatmap abgebildet.

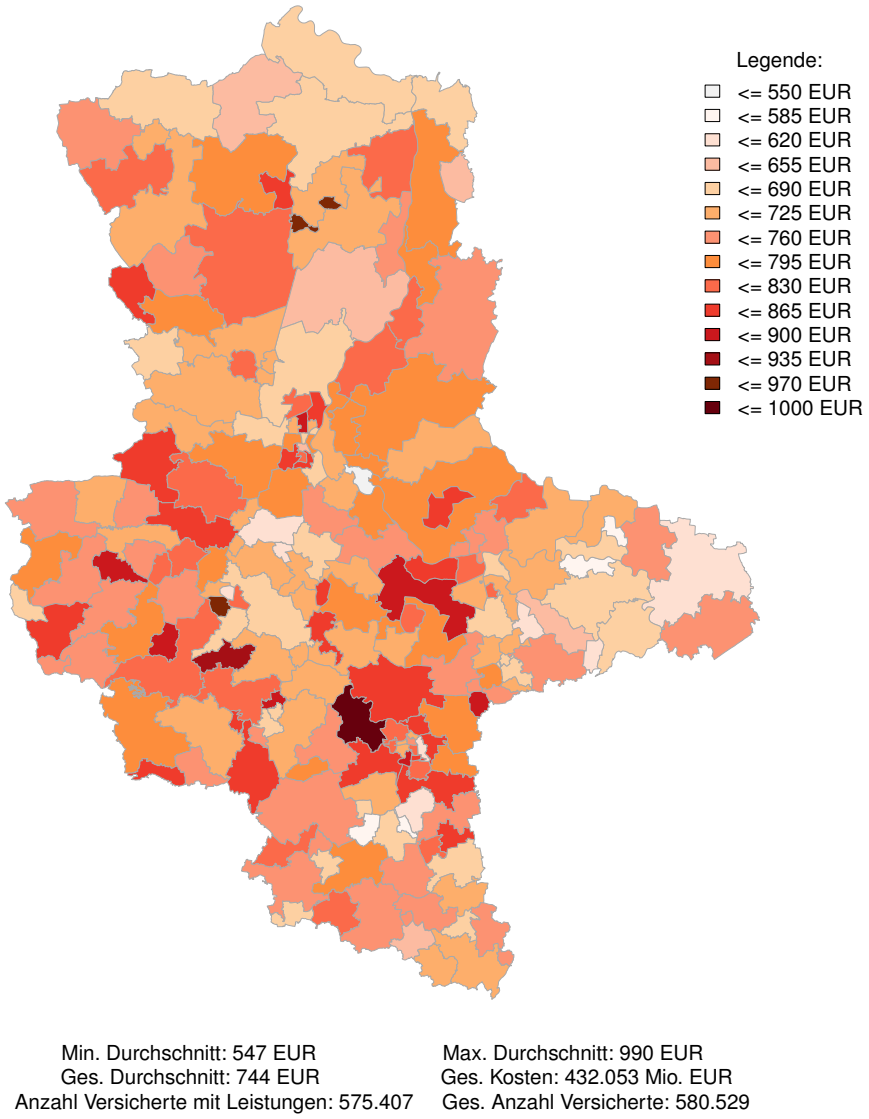

Abb. 5.38.: Durchschnittskosten pro PLZ-Gebiet

Die Abbildung 5.38 lässt erkennen, dass in der kreisfreien Stadt *Halle (Saale)* und in deren Umgebung die Durchschnittskosten tendenziell höher und im östlichen Teil Sachsen-Anhalts niedriger sind als im restlichen Bundesland. Ein Großteil ist farblich stark fragmentiert, sodass auf eine inhomogene Verteilung der Kosten geschlossen werden kann.
Der Grafik im unteren Teil sind zusätzliche Informationen über die minimal und maximal vorkommenden Durchschnittskosten pro Versicherten bezüglich der Postleitzahlengebiete zu entnehmen. Zudem ist die Gesamtanzahl der Versicherten aufgelistet, die Leistungen im Jahr 2012 in Form von Arzneiprodukten erhielten, sowie die durchschnittlichen und Gesamtkosten des Bundeslandes, wobei sich die Durchschnittskosten auf die Gesamtanzahl der AOK-Mitglieder von 580.529 beziehen.

Um einen detaillierteren Überblick über die Kostenverteilung zu erhalten, werden in Ergänzung zur Abbildung 5.38 die Versicherten in bestimmte Gruppen gegliedert. Die AOK-Versicherten werden geschlechtsspezifisch in sechs Altersklassen von *0-10 Jahre*, *11-20 Jahre*, *21-35 Jahre*, *36-50 Jahre*, *51-65 Jahre* und *66-120 Jahre* eingeteilt. Das Ziel ist die Überprüfung von signifikanten Kostenschwankungen zwischen den Gruppen, jeweils abhängig vom Geschlecht. Die Ergebnisse befinden sich in den Abbildungen 5.39-5.44.

Im Gegensatz zur Abbildung 5.38 beziehen sich alle Durchschnittskosten auf die jeweilige gesamte Anzahl Versicherter, die Leistungen in Anspruch genommen haben. Auf einigen Abbildungen sind wesentliche Unterschiede zwischen den maximalen Durchschnittskosten eines Postleitzahlgebietes und den gesamten Durchschnittskosten des Bundeslandes zu erkennen. Dies resultiert aus sog. "Ausreißern" (Einzelpatienten), die das Gesamtbild verzerren. Des Weiteren wird ersichtlich, dass Männer zumeist höhere Kosten im Mittel verursachen als Frauen der gleichen Altersgruppe. Eine Ausnahme bildet hier die Gruppe *11-20 Jahre*, in der die Frauen mit 211,- € im Durchschnitt höher liegen als Männer mit 190,- €. Zudem ist an den Kosten erkennbar, dass diese, pro Altersgruppe, erwartungsgemäß kontinuierlich ansteigen. Eine deutliche Zunahme zeigt sich bei der ungefähren Verdoppelung der Kosten zwischen den Altersgruppen *21-35* und *36-50 Jahren*.

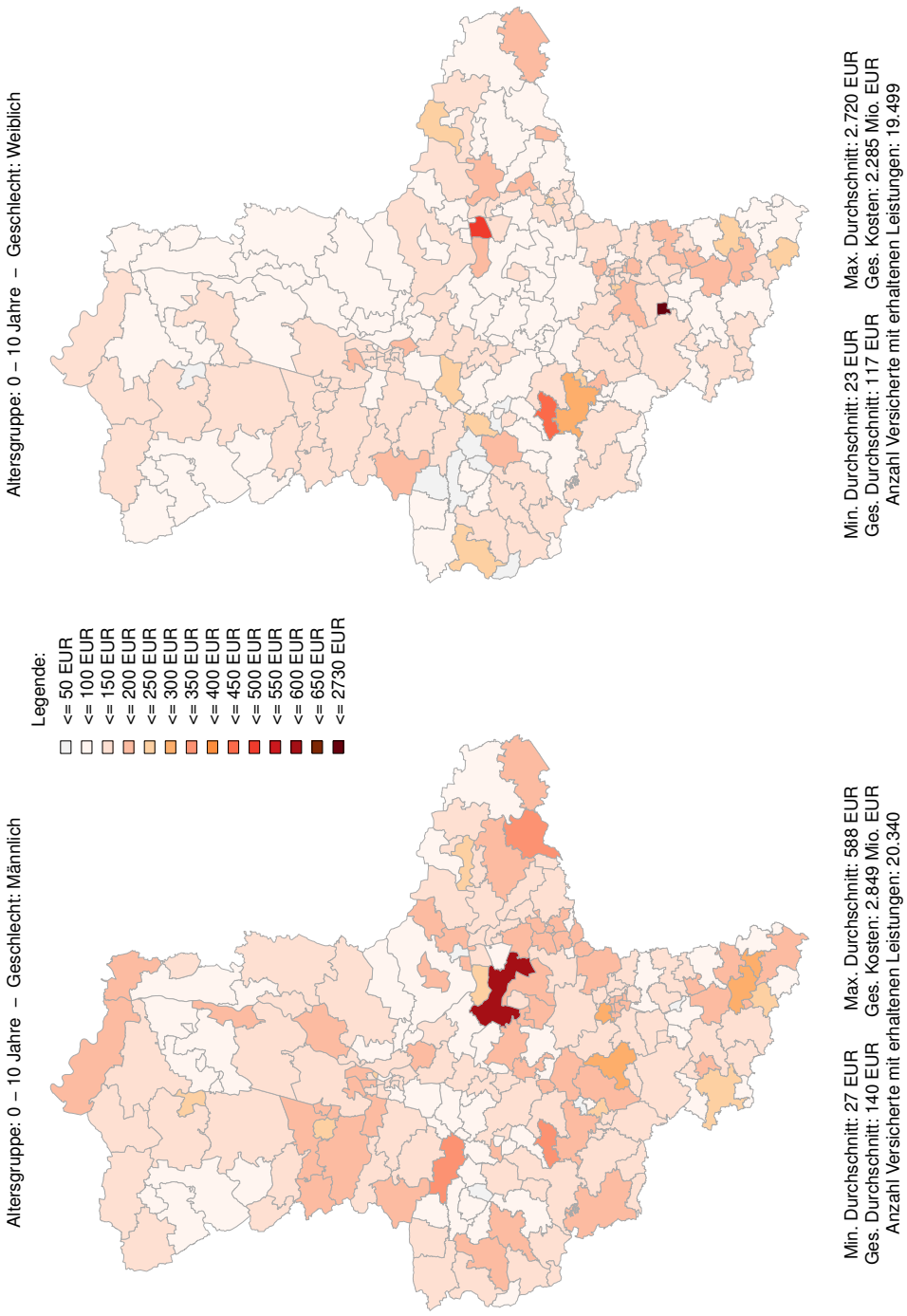

Abb. 5.39.: Durchschnittskosten pro PLZ-Gebiet (0-10 Jahre)

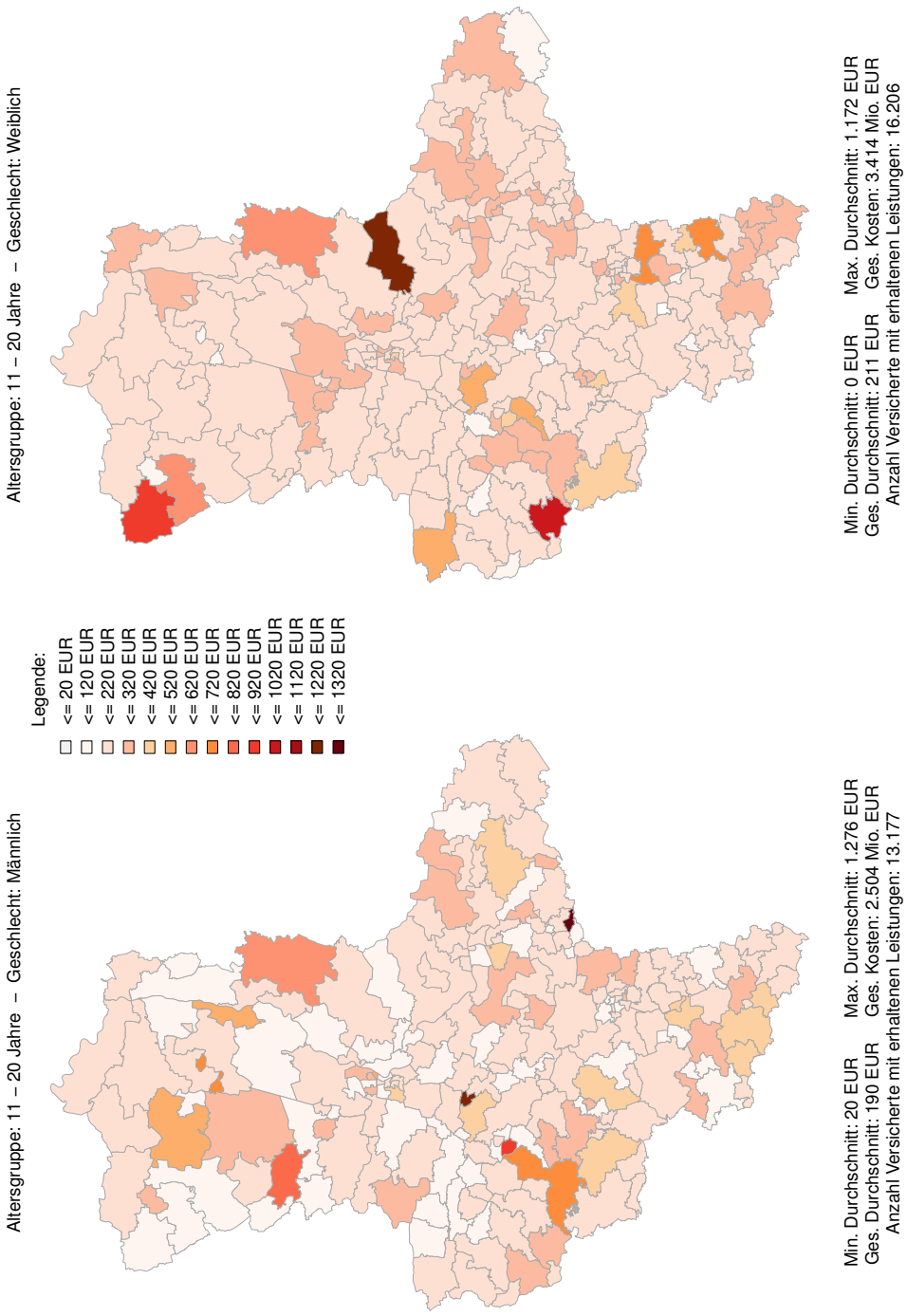

Abb. 5.40.: Durchschnittskosten pro PLZ-Gebiet (11-20 Jahre)

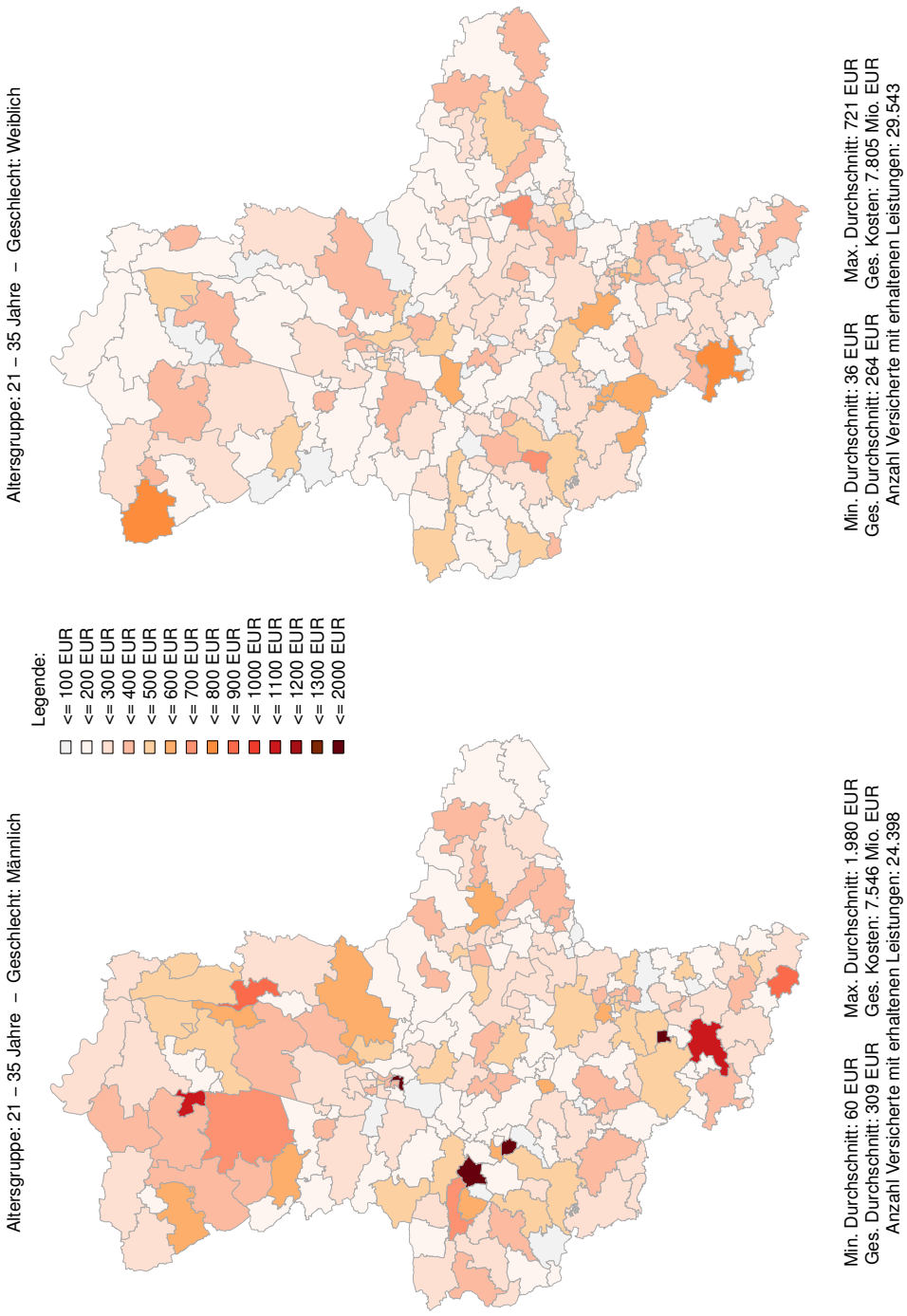

Abb. 5.41.: Durchschnittskosten pro PLZ-Gebiet (21-35 Jahre)

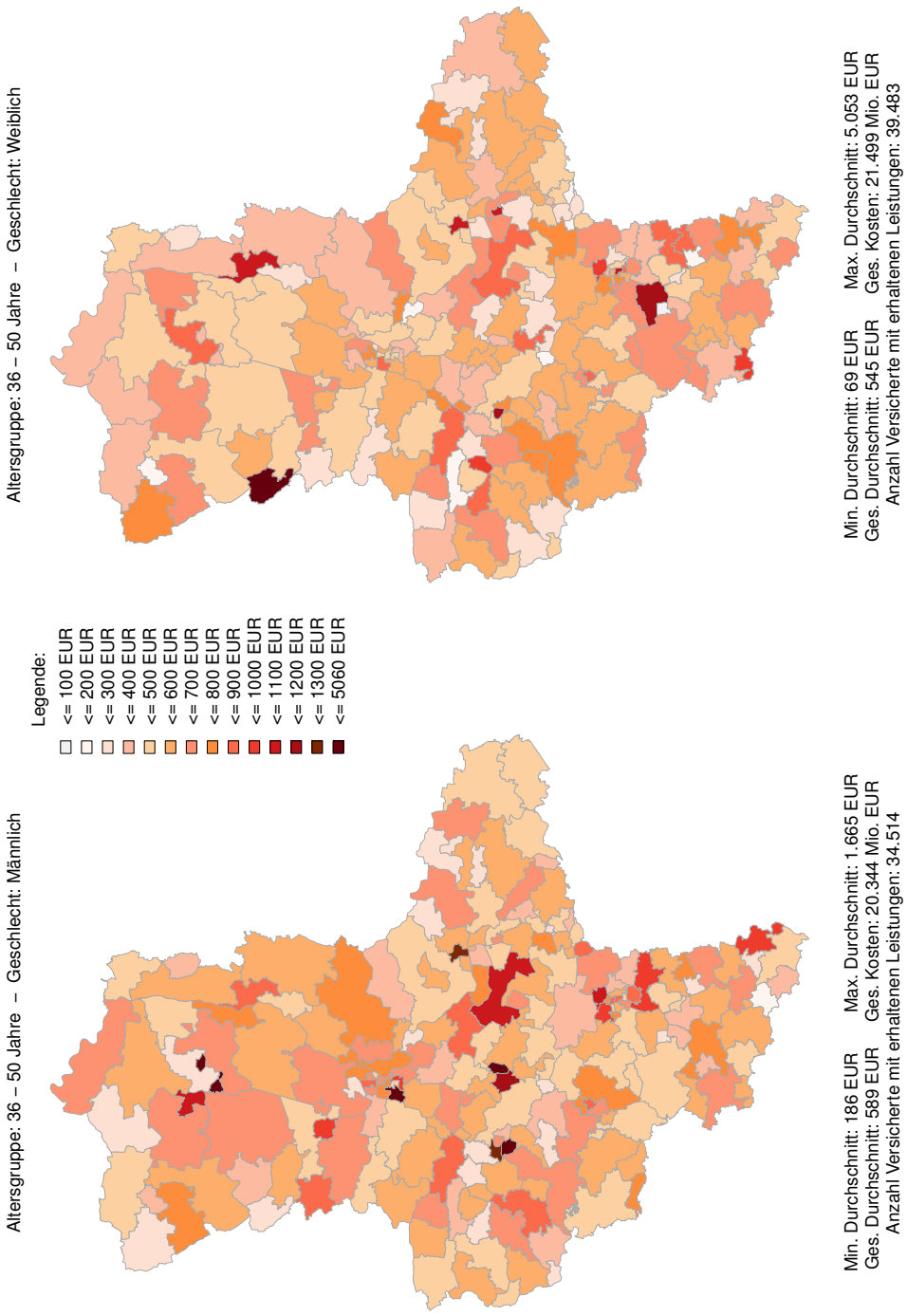

Abb. 5.42.: Durchschnittskosten pro PLZ-Gebiet (36-50 Jahre)

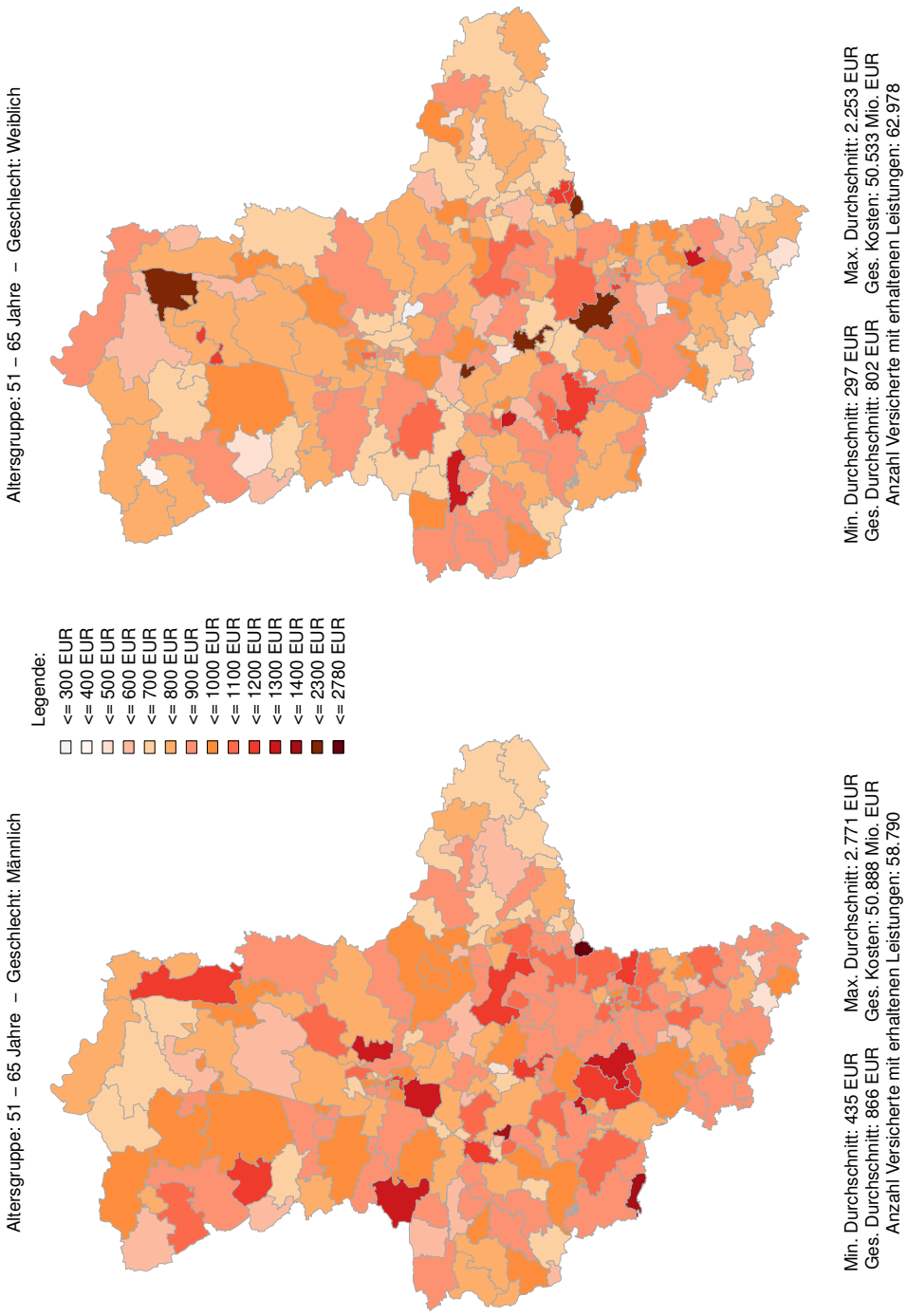

Abb. 5.43.: Durchschnittskosten pro PLZ-Gebiet (51-65 Jahre)

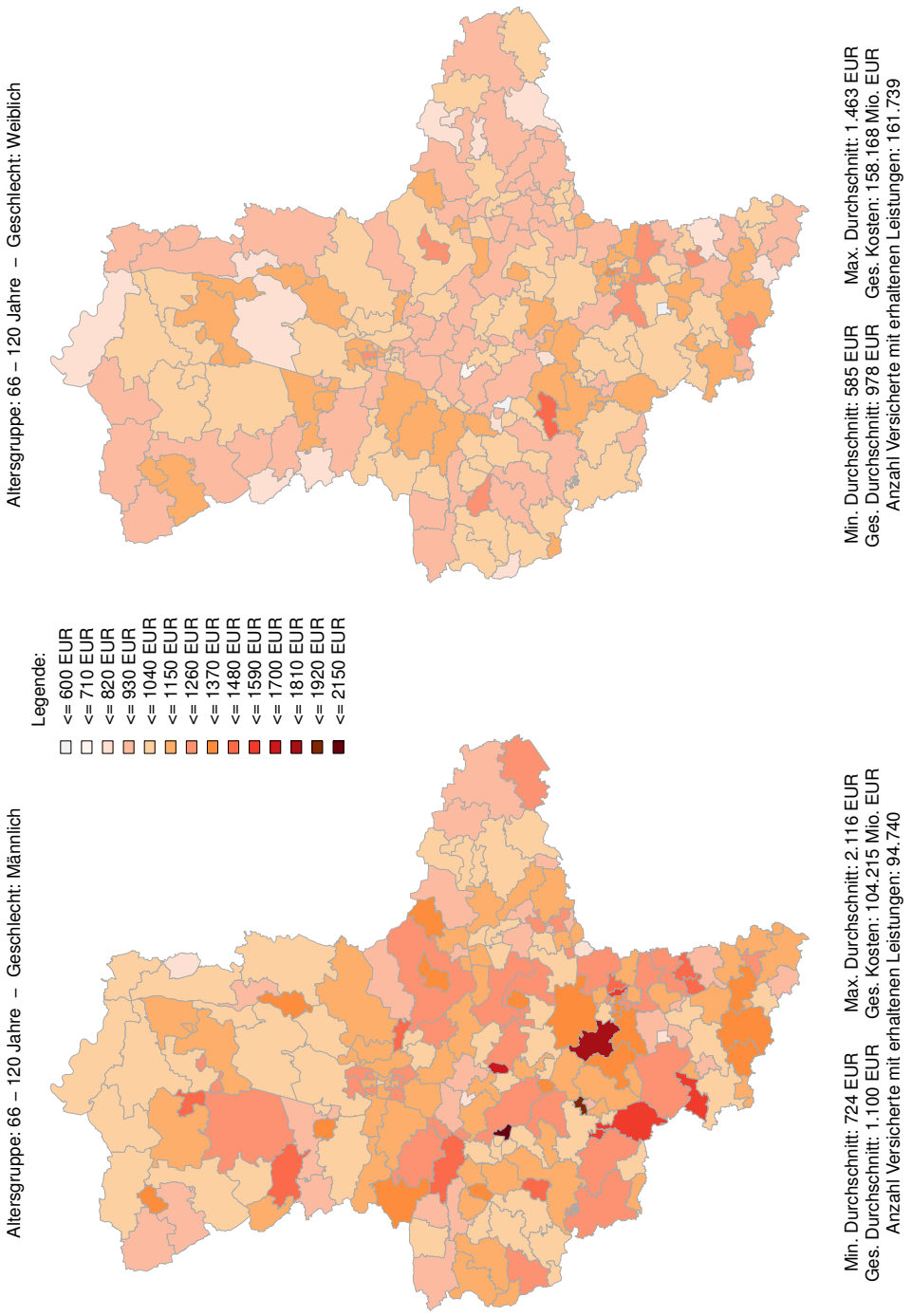

Abb. 5.44.: Durchschnittskosten pro PLZ-Gebiet (66-120 Jahre)

Nachfolgende Codezeilen beziehen sich auf die Abbildung 5.38.

```
# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
# Arzneidaten einlesen
arznei <- arzneimittel_laden()
# Die Spalten "PSEUDONYM" und "PZN" werden extrahiert
arznei_pzn <- arznei[c(1,5)]
# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()
# Die Spalten "PSEUDONYM", "AGE" und "GENDER" werden extrahiert
ambu_age_gender <- unique(ambudaten[c(1,6,7)])
# PZN_Preise einlesen
pzn_preise <- pznkosten_laden()
colnames(pzn_preise) <- c("PZN","Preis")
# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")
# Verbinden der PZN-Codes und deren Preise mit den Pseudonymen
arznei_pzn_preise <- merge(arznei_pzn,pzn_preise,by="PZN")
# Aggregation der Preise nach den Pseudonymen
kosten_pro_ver <- aggregate(arznei_pzn_preise[c(3)],by=list(arznei_pzn_preise
    $PSEUDONYM),FUN=sum)
colnames(kosten_pro_ver) <- c("PSEUDONYM","Kosten_pro_V")
# Verbinden aller Pseudonyme in SA (alle_kreise) mit dessen Geschlecht/Alter
ambu_alle_kreise <- merge(alle_kreise[c(1,2)],ambu_age_gender,by="PSEUDONYM")
# Verbinden der Ambu-Daten mit den Kosten-Pro-Versicherten
# INFO: Es fallen einige Pseudonyme weg, da in Ambudaten nicht vorhanden
sa_kosten_pro_ver <- merge(ambu_alle_kreise,kosten_pro_ver,by="PSEUDONYM")
# Gesamtkosten pro PLZ-Gebiet ermitteln
kosten_plz <- aggregate(sa_kosten_pro_ver[c(5)],by=list(sa_kosten_pro_ver$PLZ
    ),FUN=sum)
colnames(kosten_plz) <- c("PLZ","Kosten_pro_PLZ")
# Anzahl Versicherte pro PLZ ermitteln, die Leistungen erhalten haben
anzahl_plz <- sa_kosten_pro_ver[c(1,2)]
anzahl_plz <- data.frame(table(anzahl_plz$PLZ))
# Gesamtanzahl Versicherte pro PLZ-Gebiet ermitteln
anzahl_plz <- cbind(anzahl_plz,data.frame(table(alle_kreise$PLZ))[,2])
colnames(anzahl_plz) <- c("PLZ","Anzahl_Versicherte_Leistungen","Anzahl_
    Versicherte_Ges")
# Gesamtkosten und Anzahl Versicherte pro PLZ-Gebiet verbinden
sa_gesamt <- merge(kosten_plz,anzahl_plz,by="PLZ")
# Spalte hinzufügen: Durchschnittskosten pro Versicherten
sa_gesamt <- cbind(sa_gesamt,"Durchschnittskosten_pro_V"=sa_gesamt$Kosten_pro
    _PLZ / sa_gesamt$Anzahl_Versicherte_Ges)
# Data.frame für die Darstellung (maptools) erzeugen
```

```
40 shapes_plz <- data.frame(shapes$plz)
41 shapes_plz <- cbind(shapes_plz,0)
42 # Spaltenüberschriften vergeben
43 colnames(shapes_plz) <- c("PLZ","Durchschnittskosten_pro_V")
44 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
45 sa_gesamt$PLZ <- str_pad(string=sa_gesamt$PLZ,width=5,pad="0")
46 # Durchschnittskosten pro Ver. in data.frame "shapes_plz" schreiben
47 for (i in 1:length(shapes_plz$PLZ)) {tmp <- subset(x=sa_gesamt$
       Durchschnittskosten_pro_V,subset=(sa_gesamt$PLZ == shapes_plz[i,1]))
48   if (length(tmp) == 0) {
49     shapes_plz[i,2] <- 0
50   }
51   else {
52     shapes_plz[i,2] <- tmp
53   }
54 }
55 # Einteilung der Schritte für die Heatmap
56 break_points <- c(0,550,585,620,655,690,725,760,795,830,865,900,935,970,1000)
57 class <- cut(shapes_plz$Durchschnittskosten_pro_V,breaks=break_points)
58 levels(class) <- break_points[2:length(break_points)]
59 ### PLOT BEGINNT ###
60 # Plotten
61 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="
       Durchschnittliche Kosten für Arzneimittel pro Versicherten")
62 # Informationen unterhalb der Heatmap anzeigen
63 mtext(text=paste("Min. Durchschnitt: ",round(min(shapes_plz$
       Durchschnittskosten_pro_V),digits=0)," EUR"," ","Max. Durchschnitt: ",
       round(max(shapes_plz$Durchschnittskosten_pro_V),digits=0)," EUR"," ","Ges
       . Durchschnitt: ",round(sum(sa_gesamt$Kosten_pro_PLZ) / sum(sa_gesamt$
       Anzahl_Versicherte_Ges),digits=0)," EUR","\nGes. Kosten: ",round(sum(sa_
       gesamt$Kosten_pro_PLZ) / 1000000,digits=3)," Mio. EUR "," ","\n\nAnzahl
       Versicherte mit Leistungen: ",sum(sa_gesamt$Anzahl_Versicherte_Leistungen
       ),"\nGes. Anzahl Versicherte: ",sum(sa_gesamt$Anzahl_Versicherte_Ges),sep
       =""),side=1,cex=0.7,line=+2.5)
64 # Altersgruppe und Geschlecht als Text hinzufügen
65 mtext(text="Alle Altersgruppen  -  Geschlecht: Weiblich & Männlich",side=3,
       cex=0.7)
66 # Legende anzeigen
67 legend('topright',fill=farben_shape,legend=paste("<=",levels(class),"EUR"),
       cex=0.85,title="Legende:",xpd=TRUE,bty="n")
68 ### Tabelle als CSV auf Festplatte speichern
69 ## Liste mit den teuersten Versicherten
70 # Anzahl ICD-10-Codes ermitteln pro Versicherten
71 anzahl_icd_codes <- data.frame(table(unique(ambudaten[c(1,3)])[,1]))
```

```
72 colnames(anzahl_icd_codes) <- c("PSEUDONYM","Anzahl_ICD")
73 # Anzahl ATC-Codes ermitteln pro Versicherten
74 anzahl_atc_codes <- data.frame(table(unique(arznei[c(1,4)])[,1]))
75 colnames(anzahl_atc_codes) <- c("PSEUDONYM","Anzahl_ATC")
76 tabelle_sa_kosten_pro_ver <- merge(sa_kosten_pro_ver,anzahl_icd_codes,by="
       PSEUDONYM")
77 tabelle_sa_kosten_pro_ver <- merge(tabelle_sa_kosten_pro_ver,anzahl_atc_codes
       ,by="PSEUDONYM")
78 tabelle_sa_kosten_pro_ver <- tabelle_sa_kosten_pro_ver[order(-tabelle_sa_
       kosten_pro_ver$Kosten_pro_V),]
79 tabelle_sa_kosten_pro_ver$Kosten_pro_V <- format(x=as.numeric(tabelle_sa_
       kosten_pro_ver$Kosten_pro_V),nsmall=2,justify="right")
80 write.table(x=tabelle_sa_kosten_pro_ver,file="CSV/arzneikosten_pro_
       versicherten.csv",sep=";",row.names=FALSE,col.names=TRUE,fileEncoding="
       latin1",dec=".")
81 # Liste der Durchschnittkosten pro Versicherten nach PLZ
82 tabelle_sa_gesamt <- sa_gesamt
83 tabelle_sa_gesamt$Durchschnittskosten_pro_V <- round(x=tabelle_sa_gesamt$
       Durchschnittskosten_pro_V,digits=2)
84 tabelle_sa_gesamt$Durchschnittskosten_pro_V <- format(x=as.numeric(tabelle_sa
       _gesamt$Durchschnittskosten_pro_V),nsmall=2,justify="right")
85 tabelle_sa_gesamt$Kosten_pro_PLZ <- format(x=as.numeric(tabelle_sa_gesamt$
       Kosten_pro_PLZ),nsmall=2,justify="right")
86 write.table(tabelle_sa_gesamt,file="CSV/durchschnitts-arzneikosten_pro_
       versicherten_pro_plz.csv",sep=";",row.names=FALSE,col.names=TRUE,
       fileEncoding="latin1",dec=".")
```

Von Zeile 1 bis 15 werden die zur Analyse benötigten Daten importiert und beginnend ab Zeile 16 bis einschließlich 27 miteinander verbunden. Zuerst wird den verordneten PZN-Codes der Versicherten der entsprechende Preis zugeordnet (Zeile 17) und anschließend, durch die `aggregate()`-Funktion, die Gesamtkosten pro Pseudonym mit Hilfe der `sum()`-Funktion berechnet. Im nächsten Schritt wird den Versicherten ihr Alter und Geschlecht zugeordnet und mit der Kostentabelle (`kosten_pro_ver`) verknüpft (Zeile 22-25). Um die Gesamtkosten pro Postleitzahlbereich zu erhalten, wird in Zeile 27 erneut ein `aggregate()`-Aufruf eingesetzt.
Für die Ermittlung der Personen, die im Jahr 2012 Leistungen erhalten haben, sowie die Gesamtanzahl der Versicherten pro Gebiet, wird in den Zeilen 30-34 jeweils die `table()`-Funktion angewandt. Um die Durchschnittskosten zu erhalten, wird in einem letzten Berechnungsschritt der *data.frame* `kosten_plz` mit dem *data.frame* `anzahl_plz`

über die Spalte *PLZ* mittels `merge()` verbunden und dividiert (Zeile 36-38). Die Codezeilen 40-67 beinhalten die üblichen Befehle für die Darstellung der Heatmap, die in den vorherigen Kapiteln schon ausführlich beschrieben wurden.
Der letzte Quelltextabschnitt (Zeile 68-86) erstellt zwei Dateien im CSV-Format. Zum einen werden in tabellarischer Form die Durchschnittskosten pro PLZ-Gebiet erzeugt (siehe Tabelle 5.9) und zum anderen eine absteigend sortierte Liste der Versicherten nach ihren in Anspruch genommenen Leistungen, die im Kapitel 5.7.3 erläutert wird (siehe Tabelle 5.10).

Für die Abbildungen der verschiedenen Altersklassen ist der obige Quellcode überwiegend identisch, sodass nachfolgend nur auf die Differenzen eingegangen wird.

```
1  # Funktion für die Berechnung und Grafikausgabe der Durchschnittskosten pro
       PLZ-Gebiet für definierte Altersgruppen und Geschlecht
2  func_pzn_kosten <- function(geschlecht,alter_von,alter_bis,break_points,
       untertitel) {
3    # Filterung der Daten nach dem Geschlecht und dem Altersbereich
4    sa_kosten_pro_ver_auswahl <- subset(x=sa_kosten_pro_ver,subset=(sa_kosten_
         pro_ver$GENDER == geschlecht & sa_kosten_pro_ver$AGE >= alter_von & sa_
         kosten_pro_ver$AGE <= alter_bis))
5    { ... }
6    # Altersgruppe und Geschlecht als Text hinzufügen
7    mtext(text=untertitel,side=3,cex=0.7)
8    { ... }
9  }
10 # Selbstdefinierter Bereich: 0-10 Jahre Mann
11 break_points <- c(0,50,100,150,200,250,300,350,400,450,500,550,600,650,2730)
12 untertitel <- "Altersgruppe: 0 - 10 Jahre  -  Geschlecht: Männlich"
13 func_pzn_kosten("M",0,10,break_points,untertitel)
14 { ... }
```

Um eine Redundanz des Codes zu vermeiden, wurde die Berechnung und Ausgabe für die sechs Altersgruppen in eine parametrisierte Funktion (`func_pzn_kosten`) überführt. Der erste Parameter `geschlecht` erwartet als String "M" bzw. "F", die beiden Argumente `alter_von` und `alter_bis` legen den Altersbereich fest. Mit den `break_points` wird die Farbeinteilung der Heatmap bestimmt und durch den Parameter `untertitel` erhält die Grafik eine Überschrift. In Zeile 4 werden mit Hilfe der übergebenen Funktionsargumente und der `subset()`-Funktion die Datensätze entsprechend den Eingaben gefiltert.

Nachfolgende Programmzeilen sind identisch mit dem Code aus Abbildung 5.38, zuzüglich eines weiteren Befehls (`mtext()`) für die Überschrift der Grafik (Zeile 7).
Ab Zeile 11 beginnen die Aufrufe der selbstdefinierten Funktion, um die sechs Altersklassen auszugeben. Exemplarisch sind in den Zeilen 11-13 die Befehle für den Altersbereich 0-10 Jahre (Männer) angegeben.

5.7.3. Individuelle Arzneimittelkosten pro Versicherten

Eine letzte Auswertung zeigt eine Statistik über die personenbezogenen Kosten. Wie bereits auf Seite 161 erwähnt, ist der Programmcode Bestandteil der vorherigen Analyse und listet für jeden Versicherten die Kosten in absteigender Reihenfolge auf.

PSEUDONYM	PLZ	Alter	Geschlecht	Anzahl ICD-Codes	Anzahl ATC-Codes	Kosten pro Versicherten
1622386086	06198	54	F	37	9	281.332,33 €
2741174179	39646	43	F	19	10	253.974,27 €
4204603461	39116	41	M	33	14	178.506,00 €
3684629091	06193	51	F	40	17	111.859,97 €
2287153271	39120	29	M	13	12	108.293,81 €
1415382107	39393	74	M	26	18	108.092,35 €
...	...	...	...	...	...	...
...	...	...	...	...	...	...
4246180142	38489	10	M	3	1	1,00 €
3776314128	39106	12	M	3	1	0,94 €
4025373346	38836	10	F	2	1	0,94 €

Tab. 5.10.: In Anspruch genommene Leistungen pro AOK-Versicherten

Die Tabelle 5.10 zeigt einen Auszug der insgesamt 575408 Zeilen und beinhaltet neben den personenbezogenen Kosten, den Wohnort, Alter, Geschlecht, sowie die Anzahl der Erkrankungen und der verordneten Medikamente, wobei keine Mehrfachdiagnosen bzw. Verschreibungen eingehen. Der Ausschnitt bestätigt die Vermutung, dass die älteren Patienten die höchsten Kosten verursachen und eine Vielzahl an Diagnoseschlüsseln aufweisen.

6. Fazit und Ausblick

Das Ziel dieser Bachelorarbeit bestand in der deskriptiven Auswertung des vorliegenden Datenbestandes der AOK, sowie in der Beantwortung einzelner Fragestellungen.

Die Erhebung sollte eventuelle Zusammenhänge zwischen dem Wohnort bzw. dem Versicherungsstatus eines AOK-Versicherten und seinen Erkrankungen, verschriebenen Medikamenten und seinen anfallenden Kosten erkennen. Dazu wurden Patientendaten von rund 600.000 Versicherten der AOK Sachsen-Anhalts mit dem Statistikprogramm `R` ausgewertet.
Für einen Großteil der Untersuchungen wurde das Bundesland Sachsen-Anhalt entsprechend seiner Postleitzahlen in kleine regionale Bereiche aufgeteilt, während sich der geringere Teil auf die vierzehn Landkreise/kreisfreien Städte bezog.

Die Frage, ob regionale Schwerpunkte bezüglich der Häufigkeit einzelner Erkrankungen existieren, hat bei der Einzeluntersuchung bestimmter Diagnoseschlüssel einige Auffälligkeiten ergeben. Durch mehrere Analysen, sowie unter Einbeziehung der Standardabweichung oder Heatmap-Visualisierung konnte nachgewiesen werden, dass sich mehrere der ICD-10-Codes lokal stark auf ein Gebiet konzentrieren. Bei der Zusammenfassung der Codes entsprechend ihrer 22 Kapitel, wurden ebenfalls in einigen Bereichen des Landes leichte Abweichungen im nördlichen und östlichen Teil festgestellt. Diese Gebiete sind sehr ländlich bzw. industriell/städtisch geprägt, mit dementsprechend unterschiedlichen Umweltbedingungen, die auf die Unregelmäßigkeiten sicherlich einen Einfluss ausüben. Ferner konnte die Zusammenfassung kleinerer ICD-10-Gruppen, die ähnliche Diagnosen kennzeichnen, keiner speziellen Region zugewiesen werden.

In Verbindung mit den Diagnoseschlüsseln ist in der Arbeit auch der Versuch unternommen worden, zu klären, inwiefern verschriebene Arzneien und festgestellte Krankheitsbilder in Relation stehen, bzw. welches Verhältnis zwischen ICD-10-Codes und ATC-Codes besteht. Dieser Teil der Arbeit wurde eingestellt, da das zur Verfügung gestellte Datenmaterial diesbezüglich keine Verbindung zuließ. Somit war es nicht möglich, den ICD-10-Codes ihre speziellen Wirkstoffe zuzuordnen, die im Regelfall verschrieben werden.

Um Zusammenhänge zwischen dem Versicherungsstatus und einzelnen Erkrankungen zu erforschen, wurden die ersten vierzig ICD-10-Codes mit der größten Standardabweichung und einem Vorkommen von mindestens 100 Fällen in Sachsen-Anhalt betrachtet. Dabei war die Gruppe der Rentner mit einem Anteil von ca. 70% vom Gesamtaufkommen bei knapp über 50% der untersuchten Diagnoseschlüssel, vertreten. Eventuelle Gründe dafür sind einerseits der hohe Anteil an Rentnern unter den AOK-Versicherten und andererseits das Medianalter von 62 Jahren.
Bei den restlichen ICD-10-Codes sind bis auf wenige Ausnahmen, bei denen die Familienangehörigen stärker hervortreten, die Gruppe der Arbeitslosen, Beschäftigten und Familienangehörigen relativ gleichmäßig vertreten. Zwar zeigte sich bei zwei ICD-10-Codes eine größere regionale Häufung, aber in Verbindung mit dem Versicherungsstatus konnte nur in einem Fall ein Bezug erkannt werden.

Eine weitere Analyse beschäftigte sich mit den Arzneimitteln, die sich in der sog. *PRISCUS*-Liste befinden. Es handelt sich hierbei um Medikamente, die entsprechend vorangegangener Untersuchungsergebnisse für Menschen ab 65 Jahren nicht geeignet sind und sich zusätzlich negativ auf den Gesundheitszustand auswirken können. Dieser Teil beschränkte sich lediglich auf die Ermittlung der absoluten Anzahl Betroffener pro Postleitzahlgebiet ohne weiterführende Fragestellungen. Die Auswertung der Daten ergab, dass ca. 24% der über 65-jährigen AOK-Versicherten diese Medikamente erhielten und damit einen großen Anteil an der Gruppe der gefährdeten Versicherungsnehmer einnehmen.

Der Frage, ob bestimmte wirkstoffgleiche Arzneimittel in einzelnen Regionen vermehrt verschrieben werden, konnte leider aufgrund der fehlenden Zuordnung von ATC-Codes zu den PZN-Codes und dem Umstand, dass der PZN-Code keine rückwärtige Aufschlüsselung erlaubt, nicht nachgegangen werden.

Bei der Kostenverteilung wurden die Patienten in Altersgruppen eingeteilt und für jede Gruppe gesondert nach Frauen und Männern die Kosten ermittelt und diese in Beziehung zu den Postleitzahlenbereichen von Sachsen-Anhalt gebracht. In fünf der sechs Gruppen lag der Gesamtdurchschnitt der Männer stets höher als der Durchschnitt bei den Frauen. Die Erwartung, dass in der Altersgruppe 21-35 Jahre die Frauen mehr Kosten verursachen als die Männer, durch die zusätzlich in Anspruch genommenen Leistungen für Schwangerschaft und Entbindung, konnte nicht bestätigt werden. Auffällig war der Sprung zwischen der Gruppe 21-35 und 36-50 Jahre, der sich durch eine annähernde Verdoppelung der Kosten bemerkbar gemacht hat. Die Durchschnittskostenanalyse ohne eine spezifische Gruppeneinteilung, hat für den Osten des Bundeslandes die geringsten und für die kreisfreie Stadt Halle (Saale) und deren Umgebung die höchsten Ausgaben ermittelt.

Bei der Verarbeitung des Datenmaterials hat sich gezeigt, dass einige Untersuchungen durch die vorliegende Datenstruktur erschwert wurden. Beispielsweise würde eine Veränderung der Stammdaten durch die Mitaufnahme der personenspezifischen Daten wie Geburtsdatum und Geschlecht, zu einer vereinfachten Bearbeitung führen. Zudem könnte die Redundanz in den weiteren Datensätzen aufgehoben und somit die Konsistenz gesteigert werden. Ausgenommen minimaler widersprüchlicher Daten, wie in dem Kapitel *Voranalyse* erläutert, konnte das gesamte Datenmaterial gut analysiert werden.

Abschließend betrachtet wurden in dieser Arbeit, die sich mehr als eine Breiten- denn Tiefenanalyse versteht, eine Vielzahl an interessanten Ergebnissen vorgestellt, auf deren Basis weitergehende Analysen durchgeführt werden könnten. Beispielsweise wäre eine Verknüpfung der ICD-10-Codes mit den Ärzten denkbar. Dadurch könnte bei regional gehäuften Krankheitsbildern die Anzahl der diagnostizierenden Ärzte ermittelt und eine

genauere Aussage über die tatsächliche Konzentration einer Diagnose getroffen werden. Die hohe Anzahl an gefährdeten Versicherungsnehmern betreffend der *PRISCUS*-Liste sollte zu weiteren Untersuchungen und daraus resultierenden Entscheidungen führen, um sie erheblich zu reduzieren. Zum einen dient dies dem Wohl der Patienten und zum anderen sind eventuelle Kosteneinsparungen möglich. Der Sprung, der zur Verdoppelung der Kosten bei beiden Altersgruppen führt, könnte ebenfalls näher untersucht werden, um festzustellen, welche Erkrankungen bzw. Medikationen diese Steigerung verursachen.

Insgesamt gesehen, bildet vorliegende Bachelorarbeit eine solide Grundlage, um innerhalb des Forschungsprojektes daran anzuknüpfen, mit dem Ziel, dass sie ihren Teil dazu beiträgt, Verbesserungen für die Versicherten zu schaffen und gleichzeitig den Versicherungsträgern Möglichkeiten aufzeigt, zukünftig Kosten einzusparen.

7. Literaturverzeichnis

[1] BIVAND, R.: *Package 'maptools'*, http://cran.r-project.org/web/packages/maptools/maptools.pdf, 2015

[2] BIVAND, R.; PEBESMA, E.: *Package 'rgeos'*, http://cran.r-project.org/web/packages/rgeos/rgeos.pdf, 2015

[3] BUNDESMINISTERIUM FÜR GESUNDHEIT: *Aktionsplan 2013 - 2015 zur Verbesserung der Arzneimitteltherapiesicherheit in Deutschland*, http://www.akdae.de/AMTS/Aktionsplan/Aktionsplan-AMTS-2013-2015.pdf, 2013

[4] DALGAARD, P.: *Introductory Statistics with R*, Second Edition, Springer Science+Business Media, 2008

[5] DEUTSCHES INSTITUT FÜR MED. DOKUMENTATION UND INFORMATIONEN: *ATC-Klassifikation mit definierten Tagesdosen DDD*, http://www.dimdi.de/static/de/amg/atcddd/index.htm, 2013

[6] DEUTSCHES INSTITUT FÜR MED. DOKUMENTATION UND INFORMATIONEN: *ICD-10-GM*, http://www.dimdi.de/static/de/klassi/icd-10-gm/index.htm, 2015

[7] DEUTSCHES INSTITUT FÜR MED. DOKUMENTATION UND INFORMATIONEN: *ICD-10-GM - Kapitelstruktur*, https://www.dimdi.de/static/de/klassi/icd-10-gm/systematik/systematik.htm, 2015

[8] DEUTSCHES INSTITUT FÜR MED. DOKUMENTATION UND INFORMATIONEN: *ICD-10-GM - Wie viele Schlüsselnummern gibt es in der*

ICD-9 und in der ICD-10, https://www.dimdi.de/static/de/klassi/faq/icd-10/allgemein/faq_0008.htm_319159480.htm, 2014

[9] DOLIC, D.: *Statistik mit R: Einführung für Wirtschafts- und Sozialwissenschaftler*, Oldenbourg Wissenschaftsverlag GmbH, 2004

[10] ECKSTEIN, Prof. P.: *Repetitorium Statistik*, Auflage 8, Springer Gabler, 2014

[11] ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE, INC.: *ESRI Shapefile Technical Description*, http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf, 1998

[12] GROß, J.: *Grundlegende Statistik mit R*, 1. Auflage, Vieweg+Tuebner Verlag, 2010

[13] HOLT, S.; SCHMIEDL, S.; THÜRMANN, P. A.: *PRISCUS-Liste potenziell inadäquater Medikation für ältere Menschen*, http://priscus.net/download/PRISCUS-Liste_PRISCUS-TP3_2011.pdf, 2011

[14] HOLT, S.; SCHMIEDL, S.; THÜRMANN, P. A.: *PRISCUS-Liste potenziell inadäquater Medikation für ältere Menschen*, http://priscus.net/content.php?menuid=43&pos=7, 2011

[15] INFORMATIONSSTELLE FÜR ARZNEISPEZIALITÄTEN: *Funktion und Aufbau der Pharmazentralnummer*, http://www.ifaffm.de/mandanten/1/documents/02_ifa_anbieter/informationen/IFA-Info_Funktion_und_Aufbau_PZN.pdf, 2015

[16] INFORMATIONSSTELLE FÜR ARZNEISPEZIALITÄTEN: *Pharmazentralnummer (PZN)*, http://www.ifaffm.de/de/ifa-gmbh/pharmazentralnummer.html, 2015

[17] MENGES, G.: *Die Statistik - Zwölf Stationen des statistischen Arbeitens*, Auflage 1, Gabler GmbH, 1982

[18] PEBESMA, E.; BIVAND, R.: *Package 'sp'*, http://cran.r-project.org/web/packages/sp/sp.pdf, 2015

[19] PETERSOHN, H.: *Data Mining - Verfahren, Prozesse, Anwendungsarchitektur*, Oldenbourg Wissenschaftsverlag GmbH, 2005

[20] PROJEKTVERBUND PRISCUS: *Die Ziele von "priscus"*, http://priscus.net/content.php?menuid=35&pos=1

[21] RUNKLER, T.: *Data Mining: Methoden und Algorithmen intelligenter Datenanalyse*, 1. Auflage, Vieweg+Tuebner Verlag, 2010

[22] STATISTISCHES BUNDESAMT: *Lebenserwartung in Deutschland*, https://www.destatis.de/DE/ZahlenFakten/GesellschaftStaat/Bevoelkerung/Sterbefaelle/Tabellen/LebenserwartungDeutschland.html, 2015

[23] STATISTISCHES LANDESAMT SACHSEN-ANHALT: *Statistischer Bericht*, https://www.statistik.sachsen-anhalt.de/download/stat_berichte/6A102_hj_2012_01.pdf, 2012

[24] THE R FOUNDATION: *What is R?*, http://www.r-project.org/about.html, 2015

[25] THE R FOUNDATION: *Contributed Packages*, http://cran.r-project.org/web/packages, 2015

[26] TOUTENBURG, H.; HEUMANN, C.: *Deskriptive Statistik - Eine Einführung in Methoden und Anwendungen mit R und SPSS*, Springer-Verlag Berlin Heidelberg, 6.Auflage, 2008

[27] WOHLFARTH, Dr. K.: *Forschungskooperation (Versorgungsforschung) Kritische Analyse der Arzneimittelausgaben der AOK Sachsen-Anhalt, Arzneimitteltherapiesicherheit im ambulanten Sektor, Pharmakovigilanz*, 2014

[28] WOLLSCHLÄGER, D.: *Grundlagen der Datenanalyse mit R - Eine anwendungsorientierte Einführung*, 3. Auflage, Springer Verlag, 2014

[29] WORLD HEALTH ORGANIZATION COLLABORATING CENTRE FOR DRUG STATISTICS METHODOLOGY: *ATC Structure and principles*, http://www.whocc.no/atc/structure_and_principles, 2011

A. Quelltexte

Quelltext A.1: R-Quelltext initial.R

```
### Initial ###

# Postleitzahlen für Landkreise
plz_magdeburg         <- c("39104-39130")
plz_halle             <- c("06108-06132")
plz_dessau_rosslau    <- c("06842-06844","06846-06853","06855-06862")
plz_altmarkkreis_salzwedel <- c("29410-29416","38486-38489","39619","39624","
    39638","39649")
plz_stendal           <- c("14715","39517-39615","39629")
plz_boerde            <- c("39164-39171","39179","39326-39397","39646")
plz_jerichower_land   <- c("39175","39245","39279","39288-39319")
plz_harz              <- c("06458","06484-06495","06497-06502","38820-38880","
    38883-38899")
plz_salzlandkreis     <- c("06406-06430","06433-06457","06459-06469","
    39217-39240","39249","39418-39448")
plz_anhalt_bitterfeld <- c("06366-06388","06749-06753","06755-06758","
    06760-06766","06768-06771","06774-06780","06792-06809","39261-39264")
plz_wittenberg        <- c("06772-06773","06785","06868-06870","06872-06895","
    06897-06917","06919-06920","06923-06925")
plz_mansfeld_suedharz <- c("06295-06299","06301-06320","06322-06329","
    06331-06347","06526-06543")
plz_saalekreis        <- c("06179-06202","06204-06253","06255-06279")
plz_burgenlandkreis   <- c("06618-06629","06631-06650","06654-06693","
    06695-06723","06728-06729")

# String-Vektor mit den Landkreisnamen
landkreise <- c("magdeburg","halle","dessau_rosslau","altmarkkreis_salzwedel","
    stendal","boerde","jerichower_land","harz","salzlandkreis","anhalt_
    bitterfeld","wittenberg","mansfeld_suedharz","saalekreis","burgenlandkreis"
    )

# Farben für Balkendiagramme (Blau)
farben <- c("#0a519c","#3182bd","#6baed5","#b8d5e7","#e4ebfa")
```

```
25 # Farben für Balkendiagramme (Regenbogen)
26 farben_2 <- c("#A50026","#D73027","#F46D43","#FDAE61","#FEE090","#FFFFBF","#
       BFF0B9","#74C476","#37924F","#E0F3F8","#74ADD1","#4575B4","#373ca1","#080
       e72","black")

28 # Farben für die Shape-Datei bzw. Heatmap
29 farben_shape <- c("#F2F2F2","#fff5f0","#fee0d2","#fcbba1","#fdd0a2","#fdae6b","
       #fc9272","#fd8d3c","#fb6a4a","#ef3b2c","#cb181d","#a50f15","#7f2704","
       #67000d")

31 # Funktion um aus einer Tabelle einen bestimmten Bereich zu extrahieren
32 # tabelle -> übergebene Tabelle (Typ: data.frame)
33 # gruppenvektor -> Vektor nach dem gruppiert wird (Typ: vector)
34 # typ -> "c"=character und "n"=numeric
35 func_bereich <- function(tabelle,spalte,gruppenvektor,typ) {
36   df <- NULL
37   splitvektor <- NULL
38   i <- NULL
39   for (i in gruppenvektor) {
40     if (grepl("-",i) == TRUE) {
41       splitvektor <- unlist(strsplit(i,"-"))
42     }
43     else {
44       splitvektor[1] <- i
45       splitvektor[2] <- i
46     }
47     # Falls gruppenvektor numerisch (z.B. PLZ), dann umwandeln
48     if (typ == "n") {
49       von <- as.numeric(splitvektor[1])
50       bis <- as.numeric(splitvektor[2])
51     }
52     else {
53       von <- as.character(splitvektor[1])
54       bis <- as.character(splitvektor[2])
55     }
56     if (von == bis) {
57       df <- rbind(df,subset(tabelle,tabelle[,spalte] == von))
58     }
59     else {
60       df <- rbind(df,subset(tabelle,tabelle[,spalte] >= von & tabelle[,spalte]
           <= bis))
61     }
62   }
```

```
63    return(df)
64  }
65
66  # unkorrigierte Standardabweichung (wenn keine Stichprobe aus einer Gesamtheit
        getestet wird)
67  sd2 <- function(x) {sqrt(sum((x - mean(x))^2) / length(x))}
68
69  ### Einlesen der Wohlfarth_Stammdaten.txt in R (wird automatisch ein data.frame
        )
70    W_Sd <- read.table(file="/Users/Florian/Documents/Bachelorarbeit/AOK/
          Wohlfarth_Stammdaten.txt",stringsAsFactors=FALSE,header=TRUE,sep="\t",
          fileEncoding="latin1")
71
72    # Den Spalten den korrekten Datentyp zuweisen
73    W_Sd$VERSICHERUNGSART <- as.character(W_Sd$VERSICHERUNGSART)
74    W_Sd$PLZ <- as.numeric(as.character(W_Sd$PLZ))
75
76    # Doppelte Pseudonyme ermitteln
77    doppel_pseudo <- W_Sd[which(duplicated (W_Sd[,1])),1]
78
79    # Doppelte Pseudonyme entfernen (alle 82 stück)
80    stammdaten <- W_Sd[!W_Sd$PSEUDONYM %in% doppel_pseudo,]
81
82    # Aufteilen der Stammdaten nach Landkreisen
83    # Die Spalten "PSEUDONYM", "PLZ", "VERSICHERUNGSART" werden extrahiert
84    stammdaten_landkreise <- stammdaten[c(1,3,5)]
85
86    # Landkreise in Sachsen-Anhalt definieren
87    halle <- func_bereich(stammdaten_landkreise,2,plz_halle,"n")
88    saalekreis <- func_bereich(stammdaten_landkreise,2,plz_saalekreis,"n")
89    mansfeld_suedharz <- func_bereich(stammdaten_landkreise,2,plz_mansfeld_
          suedharz,"n")
90    harz <- func_bereich(stammdaten_landkreise,2,plz_harz,"n")
91    anhalt_bitterfeld <- func_bereich(stammdaten_landkreise,2,plz_anhalt_
          bitterfeld,"n")
92    salzlandkreis <- func_bereich(stammdaten_landkreise,2,plz_salzlandkreis,"n")
93    burgenlandkreis <- func_bereich(stammdaten_landkreise,2,plz_burgenlandkreis,"
          n")
94    wittenberg <- func_bereich(stammdaten_landkreise,2,plz_wittenberg,"n")
95    dessau_rosslau <- func_bereich(stammdaten_landkreise,2,plz_dessau_rosslau,"n"
          )
96    stendal <- func_bereich(stammdaten_landkreise,2,plz_stendal,"n")
97    altmarkkreis_salzwedel <- func_bereich(stammdaten_landkreise,2,plz_
          altmarkkreis_salzwedel,"n")
```

```
98    magdeburg <- func_bereich(stammdaten_landkreise,2,plz_magdeburg,"n")
99    boerde <- func_bereich(stammdaten_landkreise,2,plz_boerde,"n")
100   jerichower_land <- func_bereich(stammdaten_landkreise,2,plz_jerichower_land,"
          n")

102 # Alle Landkreise in Sachsen-Anhalt zusammenfassen
103 alle_kreise <- rbind (magdeburg,halle,dessau_rosslau,altmarkkreis_salzwedel,
104             stendal,boerde,jerichower_land,harz,salzlandkreis,
105             anhalt_bitterfeld,wittenberg,mansfeld_suedharz,
106             saalekreis,burgenlandkreis)

108 # Ermittlung der NICHT in Sachsen-Anhalt gemeldeten Versicherten
109 alle_anderen <- stammdaten_landkreise[!stammdaten_landkreise$PSEUDONYM %in%
        alle_kreise$PSEUDONYM,]

111 ### Einlesen der Wohlfarth_ambuDaten.txt in R
112 ambudaten_laden <- function() {
113   W_Ambu <- read.table(file="/Users/Florian/Documents/Bachelorarbeit/AOK/
          Wohlfahrt_ambuDaten.txt",stringsAsFactors=FALSE,header=TRUE,sep="\t",dec=
          ",")

115   # Die Spalten PSEUDONYM, AGE, GENDER werden extrahiert
116   tmp_ambu <- W_Ambu[c(1,6,7)]

118   # Divergierendes Alter und Geschlecht für ein Pseudonym ermitteln
119   tmp_doppel1 <- data.frame(table("PSEUDONYM"=unique(tmp_ambu)[,1]))
120   tmp_doppel2 <- subset(tmp_doppel1,tmp_doppel1$Freq > 1)

122   # Löschen der doppeldeutigen Pseudonym-Nummern
123   ambudaten <- W_Ambu[!W_Ambu$PSEUDONYM %in% tmp_doppel2$PSEUDONYM,]

125   # ICD Codes müssen 5-stellig sein
126   ambudaten$ICD <- substr(x=paste(ambudaten$ICD,"0000",sep=""),start=1,stop=5)

128   # Rückgabe der bereinigten AmbuDaten
129   return(ambudaten)
130 }

132 ### Einlesen der Wohlfarth_Arzneimittel.txt in R
133 arzneimittel_laden <- function() {
134   arznei <- read.table(file="/Users/Florian/Documents/Bachelorarbeit/AOK/
          Wohlfarth_Arzneimittel.txt",stringsAsFactors=FALSE,header=TRUE,sep="\t",
          dec=",")
135   return(arznei)
```

```
136 }
138 ### Einlesen der shape-Datei in R
139 shape_laden <- function(karte) {
140   # Benötigte Module/Plugins laden
141   library(sp)
142   library(rgeos)
143   library(stringr)
144   library(maptools)
145   # Falls "plz", wird Sachsen-Anhalt unterteilt nach PLZ-Gebieten geladen
146   if (karte=="plz") {
147     shapefile <- readShapeSpatial(fn="/Users/Florian/Documents/Bachelorarbeit/
          shapefiles/sachsen_anhalt/sa_plz.shp")
148   }
149   # Falls "lk", wird Sachsen-Anhalt unterteilt nach Landkreisen geladen
150   if (karte=="lk") {
151     shapefile <- readShapeSpatial(fn="/Users/Florian/Documents/Bachelorarbeit/
          shapefiles/sachsen_anhalt/sa_lk.shp")
152   }
153   # Rückgabe der Shape-Datei
154   return(shapefile)
155 }
157 ### Einlesen der Priscus-Liste_Joerg.csv in R
158 priscus_laden <- function() {
159   Priscus_CSV <- read.table(file="/Users/Florian/Documents/Bachelorarbeit/AOK/
        Priscus-Liste_Joerg.csv",stringsAsFactors=FALSE,header=TRUE,sep=";")
160   # Spalte ATC Codes auswählen
161   priscus_liste <- data.frame("ATC"=Priscus_CSV$ATC)
162   # Rückgabe der ATC-Codes der Priscus-Liste
163   return(priscus_liste)
164 }
166 ### Einlesen der pzn-kosten-tabelle.csv in R
167 pznkosten_laden <- function() {
168   PZN_Preise <- read.table(file="/Users/Florian/Documents/Bachelorarbeit/AOK/
        pzn-kosten-tabelle.csv",stringsAsFactors=FALSE,header=TRUE,sep=";",dec=",
        ")
169   # Spaltenbeschriftung hinzufügen
170   colnames(PZN_Preise) <- c("PZN","Preis")
171   # Rückgabe der PZN_Preise
172   return(PZN_Preise)
173 }
```

Quelltext A.2: R-Quelltext 1.1-Anzahl-Versicherte-Absolut-Heatmap.R

```
### Heatmap - Absolute Versicherten-Verteilung in SA ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")

# Data.frame für die Darstellung (maptools) erzeugen
anzahl_lk <- data.frame(shapes$plz)
anzahl_lk <- cbind(anzahl_lk,0)
# Spaltenüberschriften vergeben
colnames(anzahl_lk) <- c("PLZ","Freq")

# Häufigkeitstabelle über alle vorkommenden PLZ der Landkreise
plz_lk <- data.frame(table(alle_kreise$PLZ))
# Spaltenüberschriften vergeben
colnames(plz_lk) <- c("PLZ", "Freq")
# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
plz_lk$PLZ <- str_pad(string=plz_lk$PLZ,width=5,pad="0")

# Anzahl der vorkommenden PLZ der Landkreise in den data.frame "anzahl_lk"
    schreiben
for (i in 1:length(anzahl_lk$PLZ)) {tmp <- subset(x=plz_lk$Freq,subset=(plz_lk$
    PLZ == anzahl_lk[i,1]))
  if (length(tmp) == 0) {
    anzahl_lk[i,2] <- 0
  }
  else {
    anzahl_lk[i,2] <- tmp
  }
}

# Einteilung der Schritte für die Heatmap
break_points <- c
    (0,250,500,1000,2000,3000,4000,5000,6000,7000,8000,9000,10000,11000,12000)
class <- cut(anzahl_lk$Freq,breaks=break_points)
levels(class) <- break_points[2:length(break_points)]

### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(3,2,2,10))
```

```
41 # Plotten
42 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="Anzahl der
       Versicherten in Sachsen-Anhalt")

44 # Gesamtanzahl an Versicherten anzeigen
45 mtext(text=paste("Gesamtanzahl AOK-Versicherte: ",sum(plz_lk$Freq),sep=""),side
       =1,cex=0.7,line=1)

47 # Legende anzeigen
48 legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
       black",cex=1.0,title="Legende:",bty="n")
```

Quelltext A.3: R-Quelltext 1.2-Anzahl-Versicherte-Prozentual-Heatmap.R

```
1  ### Heatmap - Prozentualer Anteil der AOK an der Bevölkerung in SA ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Laden der Shape-Datei: Sachsen-Anhalt (nach Landkreisen)
7  shape_lk <- shape_laden(karte="lk")

9  # Data.frame für die Darstellung (maptools) erzeugen
10 bevoelkerung <- data.frame(shape_lk$name)

12 # Spalte hinzufügen: Gesamtanzahl der Bevölkerung pro Landkreis
13 bevoelkerung$Anzahl_LK <- c(173138,190545,176699,85329,233107,228030,94776,
       232203,147036,194180,205672,88055,119470,134622)

15 # Spalte hinzufügen: Anzahl der AOK-Versicherten pro Landkreis
16 bevoelkerung$Anzahl_AOK <- c(length(anhalt_bitterfeld$PSEUDONYM),length(
       burgenlandkreis$PSEUDONYM),length(boerde$PSEUDONYM),length(dessau_rosslau$
       PSEUDONYM),length(halle$PSEUDONYM),length(harz$PSEUDONYM),length(jerichower
       _land$PSEUDONYM),length(magdeburg$PSEUDONYM),length(mansfeld_suedharz$
       PSEUDONYM),length(saalekreis$PSEUDONYM),length(salzlandkreis$PSEUDONYM),
       length(altmarkkreis_salzwedel$PSEUDONYM),length(stendal$PSEUDONYM),length(
       wittenberg$PSEUDONYM))

18 # Prozentualen Anteil berechnen
19 bevoelkerung$Proz <- (bevoelkerung[,3] / bevoelkerung[,2])*100

21 ### PLOT BEGINNT ###
22 # Anzeigen der Landkarte bzw. Plotten
23 plot(x=shape_lk,border="black")
```

```
25 # Berechneten proz. Anteil auf der Landkarte ausgeben
26 text(x=getSpPPolygonsLabptSlots(shape_lk),labels=paste(round(x=bevoelkerung$
       Proz,digits=1),"%",sep=""),cex=0.55,col=farben[1])
```

Quelltext A.4: R-Quelltext 2.1-Versicherungsarten-Absolut-Diagramm.R

```
1  ### Barplot der Versicherungsarten (absolute Häufigkeit) ###
2
3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
5
6  # Funktion um aus 9 nur 5 Versicherungsarten zu erzeugen
7  func_v_arten <- function(frequenz,vers_arten) {
8    # Variablen definieren
9    grpvektor <- NULL; arten <- NULL
10   # 5 Versicherungsarten
11   art1 <- "arbeitslos"
12   art2 <- "beschäftigt"
13   art3 <- "Familienangehörige"
14   art4 <- "Rentner"
15   art10 <- "Sonstige"
16   arten <- c(art1,art2,art3,art4,art10)
17   # Arten 1-4 werden übernommen
18   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art1)))
19   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art2)))
20   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art3)))
21   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art4)))
22   # Arten 5-9 werden zu art10=Sonstiges zusammengefasst
23   grpvektor = rbind(grpvektor,sum(subset(x=frequenz,subset=(vers_arten != art1
         & vers_arten != art2 & vers_arten != art3 & vers_arten != art4))))
24   # Rückgabe
25   return(data.frame(VERSICHERUNGSART=arten,Freq=grpvektor))
26 }
27
28 # Häufigkeitstabelle für jede Region erstellen
29 sum_halle <- data.frame(table("VERSICHERUNGSART"=halle$VERSICHERUNGSART))
30 sum_saalekreis <- data.frame(table("VERSICHERUNGSART"=saalekreis$
       VERSICHERUNGSART))
31 sum_mansfeld_suedharz <- data.frame(table("VERSICHERUNGSART"=mansfeld_suedharz$
       VERSICHERUNGSART))
32 sum_harz <- data.frame(table("VERSICHERUNGSART"=harz$VERSICHERUNGSART))
33 sum_anhalt_bitterfeld <- data.frame(table("VERSICHERUNGSART"=anhalt_bitterfeld$
       VERSICHERUNGSART))
```

```
34 sum_salzlandkreis <- data.frame(table("VERSICHERUNGSART"=salzlandkreis$
       VERSICHERUNGSART))
35 sum_burgenlandkreis <- data.frame(table("VERSICHERUNGSART"=burgenlandkreis$
       VERSICHERUNGSART))
36 sum_wittenberg <- data.frame(table("VERSICHERUNGSART"=wittenberg$
       VERSICHERUNGSART))
37 sum_dessau_rosslau <- data.frame(table("VERSICHERUNGSART"=dessau_rosslau$
       VERSICHERUNGSART))
38 sum_stendal <- data.frame(table("VERSICHERUNGSART"=stendal$VERSICHERUNGSART))
39 sum_altmarkkreis_salzwedel <- data.frame(table("VERSICHERUNGSART"=altmarkkreis_
       salzwedel$VERSICHERUNGSART))
40 sum_magdeburg <- data.frame(table("VERSICHERUNGSART"=magdeburg$VERSICHERUNGSART
       ))
41 sum_boerde <- data.frame(table("VERSICHERUNGSART"=boerde$VERSICHERUNGSART))
42 sum_jerichower_land <- data.frame(table("VERSICHERUNGSART"=jerichower_land$
       VERSICHERUNGSART))
43 sum_alle_anderen <- data.frame(table("VERSICHERUNGSART"=alle_anderen$
       VERSICHERUNGSART))

45 # Funktionsaufruf um Versicherungsarten zusammenzufassen
46 gruppe_sum_halle <- func_v_arten(sum_halle$Freq,sum_halle$VERSICHERUNGSART)
47 gruppe_sum_saalekreis <- func_v_arten(sum_saalekreis$Freq,sum_saalekreis$
       VERSICHERUNGSART)
48 gruppe_sum_mansfeld_suedharz <- func_v_arten(sum_mansfeld_suedharz$Freq,sum_
       mansfeld_suedharz$VERSICHERUNGSART)
49 gruppe_sum_harz <- func_v_arten(sum_harz$Freq,sum_harz$VERSICHERUNGSART)
50 gruppe_sum_anhalt_bitterfeld <- func_v_arten(sum_anhalt_bitterfeld$Freq,sum_
       anhalt_bitterfeld$VERSICHERUNGSART)
51 gruppe_sum_salzlandkreis <- func_v_arten(sum_salzlandkreis$Freq,sum_
       salzlandkreis$VERSICHERUNGSART)
52 gruppe_sum_burgenlandkreis <- func_v_arten(sum_burgenlandkreis$Freq,sum_
       burgenlandkreis$VERSICHERUNGSART)
53 gruppe_sum_wittenberg <- func_v_arten(sum_wittenberg$Freq,sum_wittenberg$
       VERSICHERUNGSART)
54 gruppe_sum_dessau_rosslau <- func_v_arten(sum_dessau_rosslau$Freq,sum_dessau_
       rosslau$VERSICHERUNGSART)
55 gruppe_sum_stendal <- func_v_arten(sum_stendal$Freq,sum_stendal$
       VERSICHERUNGSART)
56 gruppe_sum_altmarkkreis_salzwedel <- func_v_arten(sum_altmarkkreis_salzwedel$
       Freq,sum_altmarkkreis_salzwedel$VERSICHERUNGSART)
57 gruppe_sum_magdeburg <- func_v_arten(sum_magdeburg$Freq,sum_magdeburg$
       VERSICHERUNGSART)
58 gruppe_sum_boerde <- func_v_arten(sum_boerde$Freq,sum_boerde$VERSICHERUNGSART)
59 gruppe_sum_jerichower_land <- func_v_arten(sum_jerichower_land$Freq,sum_
```

```
    jerichower_land$VERSICHERUNGSART)
gruppe_sum_alle_anderen <- func_v_arten(sum_alle_anderen$Freq,sum_alle_anderen$
    VERSICHERUNGSART)

# Ergebnismatrix der abs. Häufigkeiten aller Regionen
v_art_plot <- cbind(
  gruppe_sum_magdeburg$Freq,
  gruppe_sum_halle$Freq,
  gruppe_sum_dessau_rosslau$Freq,
  gruppe_sum_altmarkkreis_salzwedel$Freq,
  gruppe_sum_stendal$Freq,
  gruppe_sum_boerde$Freq,
  gruppe_sum_jerichower_land$Freq,
  gruppe_sum_harz$Freq,
  gruppe_sum_salzlandkreis$Freq,
  gruppe_sum_anhalt_bitterfeld$Freq,
  gruppe_sum_wittenberg$Freq,
  gruppe_sum_mansfeld_suedharz$Freq,
  gruppe_sum_saalekreis$Freq,
  gruppe_sum_burgenlandkreis$Freq,
  gruppe_sum_alle_anderen$Freq
)
# Überschriften setzen
rownames(v_art_plot) <- c("Arbeitslose","Beschäftigte","Familienangehörige","
    Rentner","Sonstige")
colnames(v_art_plot) <- c(landkreise,"au ß erhalb_Sachsen_Anhalt")

### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(9,6,4,2))

# Y-Achse Markierung
ybereich <- pretty(x=c(0,signif(1.1*max(colSums(v_art_plot)),digits=1)),n=15)

# Plotten
balken <- barplot(height=v_art_plot,main="Verteilung der Versicherungsarten
    nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,axisnames=FALSE,col=
    farben,ylim=c(0,1.1*max(colSums(v_art_plot))))

# Legende anzeigen
legend('topright',legend=rownames(v_art_plot),fill=farben,cex=0.8)

# X-Achse anzeigen
text(x=balken+0.1,y=-2500,srt=35,adj=1,xpd=TRUE,labels=colnames(v_art_plot),cex
```

```
    =0.9)

100 # X-Achsen-Beschriftung
101 mtext(text="Regionen",side=1,line=6.5)

103 # Y-Achse anzeigen
104 axis(side=2,at=ybereich,labels=format(ybereich,big.mark=".",scientific=FALSE),
    las=2,cex.axis=0.9,pos=-0.1)

106 # Y-Achsen-Beschriftung
107 mtext(text="Anzahl Personen",side=2,line=3.5)

109 # Absolute Anzahl pro Region über Balken anzeigen
110 text(x=balken,y=colSums(v_art_plot)+1500,labels=colSums(v_art_plot),cex=0.8)


113 ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit der
    Versicherungsarten pro Landkreis)
114 write.table(x=cbind(Landkreise=rownames(v_art_plot),v_art_plot),file="/Users/
    Florian/Documents/Bachelorarbeit/CSV/versicherungsarten_pro_lk_absolut.csv"
    ,sep=";",row.names=FALSE,col.names=TRUE,fileEncoding="latin1")
```

Quelltext A.5: R-Quelltext 2.2-Versicherungsarten-Prozentual-Diagramm.R

```
1  ### Barplot der Versicherungsarten (relative Häufigkeit) ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Funktion um aus 9 nur 5 Versicherungsarten zu erzeugen
7  func_v_arten <- function(frequenz,vers_arten) {
8    # Variablen definieren
9    grpvektor <- NULL; arten <- NULL
10   # 5 Versicherungsarten
11   art1 <- "arbeitslos"
12   art2 <- "beschäftigt"
13   art3 <- "Familienangehörige"
14   art4 <- "Rentner"
15   art10 <- "Sonstige"
16   arten <- c(art1,art2,art3,art4,art10)
17   # Arten 1-4 werden übernommen
18   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art1)))
19   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art2)))
20   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art3)))
```

```
21    grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art4)))
22    # Arten 5-9 werden zu art10=Sonstiges zusammengefasst
23    grpvektor = rbind(grpvektor,sum(subset(x=frequenz,subset=(vers_arten != art1
          & vers_arten != art2 & vers_arten != art3 & vers_arten != art4))))
24    # Rückgabe
25    return(data.frame(VERSICHERUNGSART=arten,Freq=grpvektor))
26  }

28  # Häufigkeitstabelle für jede Region erstellen
29  sum_halle <- data.frame(table("VERSICHERUNGSART"=halle$VERSICHERUNGSART))
30  sum_saalekreis <- data.frame(table("VERSICHERUNGSART"=saalekreis$
        VERSICHERUNGSART))
31  sum_mansfeld_suedharz <- data.frame(table("VERSICHERUNGSART"=mansfeld_suedharz$
        VERSICHERUNGSART))
32  sum_harz <- data.frame(table("VERSICHERUNGSART"=harz$VERSICHERUNGSART))
33  sum_anhalt_bitterfeld <- data.frame(table("VERSICHERUNGSART"=anhalt_bitterfeld$
        VERSICHERUNGSART))
34  sum_salzlandkreis <- data.frame(table("VERSICHERUNGSART"=salzlandkreis$
        VERSICHERUNGSART))
35  sum_burgenlandkreis <- data.frame(table("VERSICHERUNGSART"=burgenlandkreis$
        VERSICHERUNGSART))
36  sum_wittenberg <- data.frame(table("VERSICHERUNGSART"=wittenberg$
        VERSICHERUNGSART))
37  sum_dessau_rosslau <- data.frame(table("VERSICHERUNGSART"=dessau_rosslau$
        VERSICHERUNGSART))
38  sum_stendal <- data.frame(table("VERSICHERUNGSART"=stendal$VERSICHERUNGSART))
39  sum_altmarkkreis_salzwedel <- data.frame(table("VERSICHERUNGSART"=altmarkkreis_
        salzwedel$VERSICHERUNGSART))
40  sum_magdeburg <- data.frame(table("VERSICHERUNGSART"=magdeburg$VERSICHERUNGSART
        ))
41  sum_boerde <- data.frame(table("VERSICHERUNGSART"=boerde$VERSICHERUNGSART))
42  sum_jerichower_land <- data.frame(table("VERSICHERUNGSART"=jerichower_land$
        VERSICHERUNGSART))
43  sum_alle_anderen <- data.frame(table("VERSICHERUNGSART"=alle_anderen$
        VERSICHERUNGSART))

45  # Funktionsaufruf um Versicherungsarten zusammenzufassen
46  gruppe_sum_halle <- func_v_arten(sum_halle$Freq,sum_halle$VERSICHERUNGSART)
47  gruppe_sum_saalekreis <- func_v_arten(sum_saalekreis$Freq,sum_saalekreis$
        VERSICHERUNGSART)
48  gruppe_sum_mansfeld_suedharz <- func_v_arten(sum_mansfeld_suedharz$Freq,sum_
        mansfeld_suedharz$VERSICHERUNGSART)
49  gruppe_sum_harz <- func_v_arten(sum_harz$Freq,sum_harz$VERSICHERUNGSART)
50  gruppe_sum_anhalt_bitterfeld <- func_v_arten(sum_anhalt_bitterfeld$Freq,sum_
```

```
    anhalt_bitterfeld$VERSICHERUNGSART)
gruppe_sum_salzlandkreis <- func_v_arten(sum_salzlandkreis$Freq,sum_
    salzlandkreis$VERSICHERUNGSART)
gruppe_sum_burgenlandkreis <- func_v_arten(sum_burgenlandkreis$Freq,sum_
    burgenlandkreis$VERSICHERUNGSART)
gruppe_sum_wittenberg <- func_v_arten(sum_wittenberg$Freq,sum_wittenberg$
    VERSICHERUNGSART)
gruppe_sum_dessau_rosslau <- func_v_arten(sum_dessau_rosslau$Freq,sum_dessau_
    rosslau$VERSICHERUNGSART)
gruppe_sum_stendal <- func_v_arten(sum_stendal$Freq,sum_stendal$
    VERSICHERUNGSART)
gruppe_sum_altmarkkreis_salzwedel <- func_v_arten(sum_altmarkkreis_salzwedel$
    Freq,sum_altmarkkreis_salzwedel$VERSICHERUNGSART)
gruppe_sum_magdeburg <- func_v_arten(sum_magdeburg$Freq,sum_magdeburg$
    VERSICHERUNGSART)
gruppe_sum_boerde <- func_v_arten(sum_boerde$Freq,sum_boerde$VERSICHERUNGSART)
gruppe_sum_jerichower_land <- func_v_arten(sum_jerichower_land$Freq,sum_
    jerichower_land$VERSICHERUNGSART)
gruppe_sum_alle_anderen <- func_v_arten(sum_alle_anderen$Freq,sum_alle_anderen$
    VERSICHERUNGSART)

# Ergebnismatrix der abs. Häufigkeiten aller Regionen
v_art_plot <- cbind(
  gruppe_sum_magdeburg$Freq,
  gruppe_sum_halle$Freq,
  gruppe_sum_dessau_rosslau$Freq,
  gruppe_sum_altmarkkreis_salzwedel$Freq,
  gruppe_sum_stendal$Freq,
  gruppe_sum_boerde$Freq,
  gruppe_sum_jerichower_land$Freq,
  gruppe_sum_harz$Freq,
  gruppe_sum_salzlandkreis$Freq,
  gruppe_sum_anhalt_bitterfeld$Freq,
  gruppe_sum_wittenberg$Freq,
  gruppe_sum_mansfeld_suedharz$Freq,
  gruppe_sum_saalekreis$Freq,
  gruppe_sum_burgenlandkreis$Freq,
  gruppe_sum_alle_anderen$Freq
  )
# Überschriften setzen
rownames(v_art_plot) <- c("Arbeitslose","Beschäftigte","Familienangehörige","
    Rentner","Sonstige")
colnames(v_art_plot) <- c(landkreise,"au ß erhalb_Sachsen_Anhalt")
```

```
84  # Prozentuale Ergebnismatrix erstellen
85  v_art_plot_proz <- round(x=prop.table(x=v_art_plot,margin=2)*100,digits=2)

87  ### PLOT BEGINNT ###
88  # Seitenabstand
89  par(mar=c(9,5,5,10.3))

91  # Legende zusammensetzen
92  leg_name <- rownames(v_art_plot)
93  leg_proz <- c(
94        round(sum(v_art_plot[1,]) * 100 / sum(v_art_plot),digits=2),
95        round(sum(v_art_plot[2,]) * 100 / sum(v_art_plot),digits=2),
96        round(sum(v_art_plot[3,]) * 100 / sum(v_art_plot),digits=2),
97        round(sum(v_art_plot[4,]) * 100 / sum(v_art_plot),digits=2),
98        round(sum(v_art_plot[5,]) * 100 / sum(v_art_plot),digits=2)
99  )
100 legende <- paste(leg_name," (",leg_proz,"%)",sep="")

102 # Y-Achse Markierung
103 ybereich <- pretty(x=c(0,100),n=10)

105 # Plotten
106 balken2 <- barplot(height=v_art_plot_proz,main="Verteilung der
        Versicherungsarten nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,
        axisnames=FALSE,col=farben,space=0.2)

108 # Legende anzeigen
109 legend(x=18.5,y=100,legend=legende,fill=farben,cex=0.8,xpd=TRUE)

111 # X-Achse anzeigen
112 text(x=balken2+0.1,y=-3,srt=35,adj=1,xpd=TRUE,labels=colnames(v_art_plot_proz),
        cex=0.9)

114 # X-Achsen-Beschriftung
115 mtext(text="Regionen",side=1,line=6.5)

117 # Y-Achse anzeigen
118 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos=-0.1)

120 # Y-Achsen-Beschriftung
121 mtext(text="Prozent",side=2,line=2)

123 # Prozente anzeigen für Arbeitslose,Beschäftigte,Familienangehörige,Rentner,
        Sonstige
```

```
124 text(x=balken2+0.33,y=v_art_plot_proz[1,]-4,adj= 1,xpd=TRUE,labels=paste(v_art_
        plot_proz[1,],"%"),cex=0.75,col="white")
125 text(x=balken2+0.33,y=22,adj=1,xpd=TRUE,labels=paste(v_art_plot_proz[2,],"%"),
        cex=0.75,col="white")
126 text(x=balken2+0.33,y=colSums(v_art_plot_proz[1:3,])-8,adj=1,xpd=TRUE,labels=
        paste(v_art_plot_proz[3,],"%"),cex=0.75,col="black")
127 text(x=balken2+0.33,y=78,adj=1,xpd=TRUE,labels=paste(v_art_plot_proz[4,],"%"),
        cex=0.75,col="black")
128 text(x=balken2+0.33,y=colSums(v_art_plot_proz[1:5,])-1,adj=1,xpd=TRUE,labels=
        paste(v_art_plot_proz[5,],"%"),cex=0.75,col="black")
```

Quelltext A.6: R-Quelltext 2.3-Versicherungsarten-Prozentual-Frau-Mann-Diagramm.R

```
1  ### Barplot der Versicherungsarten (relative Häufigkeit) nach Mann und Frau ###
2
3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
5
6  # Funktion um aus 9 nur 5 Versicherungsarten zu erzeugen
7  func_v_arten <- function(frequenz,vers_arten) {
8
9    # Variablen definieren
10   grpvektor <- NULL; arten <- NULL
11   # 5 Versicherungsarten
12   art1 <- "arbeitslos"
13   art2 <- "beschäftigt"
14   art3 <- "Familienangehörige"
15   art4 <- "Rentner"
16   art10 <- "Sonstige"
17   arten <- c(art1,art2,art3,art4,art10)
18
19   # Arten 1-4 werden übernommen
20   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art1)))
21   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art2)))
22   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art3)))
23   grpvektor = rbind(grpvektor,subset(x=frequenz,subset=(vers_arten == art4)))
24   # Arten 5-9 werden zu art10=Sonstiges zusammengefasst
25   grpvektor = rbind(grpvektor,sum(subset(x=frequenz,subset=(vers_arten != art1
         & vers_arten != art2 & vers_arten != art3 & vers_arten != art4))))
26
27   # Rückgabe
28   return(data.frame(VERSICHERUNGSART=arten,Freq=grpvektor))
29 }
```

```
31 # Ambudaten einlesen und bereinigen
32 ambudaten <- ambudaten_laden()
33
34 # Die Spalten "PSEUDONYM", "GESCHLECHT" werden extrahiert
35 ambu_gender <- ambudaten[c(1,7)]
36
37 # Verbinden der Stammdaten mit den Ambudaten für jede Region
38 halle_gender <- unique(merge(halle,ambu_gender,by="PSEUDONYM"))
39 saalekreis_gender <- unique(merge(saalekreis,ambu_gender,by="PSEUDONYM"))
40 mansfeld_suedharz_gender <- unique(merge(mansfeld_suedharz,ambu_gender,by="
       PSEUDONYM"))
41 harz_gender <- unique(merge(harz,ambu_gender,by="PSEUDONYM"))
42 anhalt_bitterfeld_gender <- unique(merge(anhalt_bitterfeld,ambu_gender,by="
       PSEUDONYM"))
43 salzlandkreis_gender <- unique(merge(salzlandkreis,ambu_gender,by="PSEUDONYM"))
44 burgenlandkreis_gender <- unique(merge(burgenlandkreis,ambu_gender,by="
       PSEUDONYM"))
45 wittenberg_gender <- unique(merge(wittenberg,ambu_gender,by="PSEUDONYM"))
46 dessau_rosslau_gender <- unique(merge(dessau_rosslau,ambu_gender,by="PSEUDONYM"
       ))
47 stendal_gender <- unique(merge(stendal,ambu_gender,by="PSEUDONYM"))
48 altmarkkreis_salzwedel_gender <- unique(merge(altmarkkreis_salzwedel,ambu_
       gender,by="PSEUDONYM"))
49 magdeburg_gender <- unique(merge(magdeburg,ambu_gender,by="PSEUDONYM"))
50 boerde_gender <- unique(merge(boerde,ambu_gender,by="PSEUDONYM"))
51 jerichower_land_gender <- unique(merge(jerichower_land,ambu_gender,by="
       PSEUDONYM"))
52 alle_anderen_gender <- unique(merge(alle_anderen,ambu_gender,by="PSEUDONYM"))
53
54 # Aggregieren der Versicherungsarten nach Mann und Frau für jede Region
55 halle_gender_freq <- aggregate(x=halle_gender$GENDER,by=list(VERSICHERUNGSART=
       halle_gender$VERSICHERUNGSART),FUN=table)
56 saalekreis_gender_freq <- aggregate(x=saalekreis_gender$GENDER,by=list(
       VERSICHERUNGSART=saalekreis_gender$VERSICHERUNGSART),FUN=table)
57 mansfeld_suedharz_gender_freq <- aggregate(x=mansfeld_suedharz_gender$GENDER,by
       =list(VERSICHERUNGSART=mansfeld_suedharz_gender$VERSICHERUNGSART),FUN=table
       )
58 harz_gender_freq <- aggregate(x=harz_gender$GENDER,by=list(VERSICHERUNGSART=
       harz_gender$VERSICHERUNGSART),FUN=table)
59 anhalt_bitterfeld_gender_freq <- aggregate(x=anhalt_bitterfeld_gender$GENDER,by
       =list(VERSICHERUNGSART=anhalt_bitterfeld_gender$VERSICHERUNGSART),FUN=table
       )
60 salzlandkreis_gender_freq <- aggregate(x=salzlandkreis_gender$GENDER,by=list(
       VERSICHERUNGSART=salzlandkreis_gender$VERSICHERUNGSART),FUN=table)
```

```
61 burgenlandkreis_gender_freq <- aggregate(x=burgenlandkreis_gender$GENDER,by=
       list(VERSICHERUNGSART=burgenlandkreis_gender$VERSICHERUNGSART),FUN=table)
62 wittenberg_gender_freq <- aggregate(x=wittenberg_gender$GENDER,by=list(
       VERSICHERUNGSART=wittenberg_gender$VERSICHERUNGSART),FUN=table)
63 dessau_rosslau_gender_freq <- aggregate(x=dessau_rosslau_gender$GENDER,by=list(
       VERSICHERUNGSART=dessau_rosslau_gender$VERSICHERUNGSART),FUN=table)
64 stendal_gender_freq <- aggregate(x=stendal_gender$GENDER,by=list(
       VERSICHERUNGSART=stendal_gender$VERSICHERUNGSART),FUN=table)
65 altmarkkreis_salzwedel_gender_freq <- aggregate(x=altmarkkreis_salzwedel_gender
       $GENDER,by=list(VERSICHERUNGSART=altmarkkreis_salzwedel_gender$
       VERSICHERUNGSART),FUN=table)
66 magdeburg_gender_freq <- aggregate(x=magdeburg_gender$GENDER,by=list(
       VERSICHERUNGSART=magdeburg_gender$VERSICHERUNGSART),FUN=table)
67 boerde_gender_freq <- aggregate(x=boerde_gender$GENDER,by=list(VERSICHERUNGSART
       =boerde_gender$VERSICHERUNGSART),FUN=table)
68 jerichower_land_gender_freq <- aggregate(x=jerichower_land_gender$GENDER,by=
       list(VERSICHERUNGSART=jerichower_land_gender$VERSICHERUNGSART),FUN=table)
69 alle_anderen_gender_freq <- aggregate(x=alle_anderen_gender$GENDER,by=list(
       VERSICHERUNGSART=alle_anderen_gender$VERSICHERUNGSART),FUN=table)

71 # Funktionsaufruf um Versicherungsarten zusammenzufassen (Frauen)
72 gruppe_sum_halle_f <- func_v_arten(halle_gender_freq$x[,1],halle_gender_freq$
       VERSICHERUNGSART)
73 gruppe_sum_saalekreis_f <- func_v_arten(saalekreis_gender_freq$x[,1],saalekreis
       _gender_freq$VERSICHERUNGSART)
74 gruppe_sum_mansfeld_suedharz_f <- func_v_arten(mansfeld_suedharz_gender_freq$x
       [,1],mansfeld_suedharz_gender_freq$VERSICHERUNGSART)
75 gruppe_sum_harz_f <- func_v_arten(harz_gender_freq$x[,1],harz_gender_freq$
       VERSICHERUNGSART)
76 gruppe_sum_anhalt_bitterfeld_f <- func_v_arten(anhalt_bitterfeld_gender_freq$x
       [,1],anhalt_bitterfeld_gender_freq$VERSICHERUNGSART)
77 gruppe_sum_salzlandkreis_f <- func_v_arten(salzlandkreis_gender_freq$x[,1],
       salzlandkreis_gender_freq$VERSICHERUNGSART)
78 gruppe_sum_burgenlandkreis_f <- func_v_arten(burgenlandkreis_gender_freq$x[,1],
       burgenlandkreis_gender_freq$VERSICHERUNGSART)
79 gruppe_sum_wittenberg_f <- func_v_arten(wittenberg_gender_freq$x[,1],wittenberg
       _gender_freq$VERSICHERUNGSART)
80 gruppe_sum_dessau_rosslau_f <- func_v_arten(dessau_rosslau_gender_freq$x[,1],
       dessau_rosslau_gender_freq$VERSICHERUNGSART)
81 gruppe_sum_stendal_f <- func_v_arten(stendal_gender_freq$x[,1],stendal_gender_
       freq$VERSICHERUNGSART)
82 gruppe_sum_altmarkkreis_salzwedel_f <- func_v_arten(altmarkkreis_salzwedel_
       gender_freq$x[,1],altmarkkreis_salzwedel_gender_freq$VERSICHERUNGSART)
83 gruppe_sum_magdeburg_f <- func_v_arten(magdeburg_gender_freq$x[,1],magdeburg_
```

```
    gender_freq$VERSICHERUNGSART)
gruppe_sum_boerde_f <- func_v_arten(boerde_gender_freq$x[,1],boerde_gender_freq
    $VERSICHERUNGSART)
gruppe_sum_jerichower_land_f <- func_v_arten(jerichower_land_gender_freq$x[,1],
    jerichower_land_gender_freq$VERSICHERUNGSART)
gruppe_sum_alle_anderen_f <- func_v_arten(alle_anderen_gender_freq$x[,1],alle_
    anderen_gender_freq$VERSICHERUNGSART)

# Funktionsaufruf um Versicherungsarten zusammenzufassen (Männer)
gruppe_sum_halle_m <- func_v_arten(halle_gender_freq$x[,2],halle_gender_freq$
    VERSICHERUNGSART)
gruppe_sum_saalekreis_m <- func_v_arten(saalekreis_gender_freq$x[,2],saalekreis
    _gender_freq$VERSICHERUNGSART)
gruppe_sum_mansfeld_suedharz_m <- func_v_arten(mansfeld_suedharz_gender_freq$x
    [,2],mansfeld_suedharz_gender_freq$VERSICHERUNGSART)
gruppe_sum_harz_m <- func_v_arten(harz_gender_freq$x[,2],harz_gender_freq$
    VERSICHERUNGSART)
gruppe_sum_anhalt_bitterfeld_m <- func_v_arten(anhalt_bitterfeld_gender_freq$x
    [,2],anhalt_bitterfeld_gender_freq$VERSICHERUNGSART)
gruppe_sum_salzlandkreis_m <- func_v_arten(salzlandkreis_gender_freq$x[,2],
    salzlandkreis_gender_freq$VERSICHERUNGSART)
gruppe_sum_burgenlandkreis_m <- func_v_arten(burgenlandkreis_gender_freq$x[,2],
    burgenlandkreis_gender_freq$VERSICHERUNGSART)
gruppe_sum_wittenberg_m <- func_v_arten(wittenberg_gender_freq$x[,2],wittenberg
    _gender_freq$VERSICHERUNGSART)
gruppe_sum_dessau_rosslau_m <- func_v_arten(dessau_rosslau_gender_freq$x[,2],
    dessau_rosslau_gender_freq$VERSICHERUNGSART)
gruppe_sum_stendal_m <- func_v_arten(stendal_gender_freq$x[,2],stendal_gender_
    freq$VERSICHERUNGSART)
gruppe_sum_altmarkkreis_salzwedel_m <- func_v_arten(altmarkkreis_salzwedel_
    gender_freq$x[,2],altmarkkreis_salzwedel_gender_freq$VERSICHERUNGSART)
gruppe_sum_magdeburg_m <- func_v_arten(magdeburg_gender_freq$x[,2],magdeburg_
    gender_freq$VERSICHERUNGSART)
gruppe_sum_boerde_m <- func_v_arten(boerde_gender_freq$x[,2],boerde_gender_freq
    $VERSICHERUNGSART)
gruppe_sum_jerichower_land_m <- func_v_arten(jerichower_land_gender_freq$x[,2],
    jerichower_land_gender_freq$VERSICHERUNGSART)
gruppe_sum_alle_anderen_m <- func_v_arten(alle_anderen_gender_freq$x[,2],alle_
    anderen_gender_freq$VERSICHERUNGSART)

# Ergebnismatrix der abs. Häufigkeiten aller Regionen für Mann und Frau
v_art_plot <- cbind(
  gruppe_sum_magdeburg_f$Freq,
  gruppe_sum_magdeburg_m$Freq,
```

```
109   gruppe_sum_halle_f$Freq,
110   gruppe_sum_halle_m$Freq,
111   gruppe_sum_dessau_rosslau_f$Freq,
112   gruppe_sum_dessau_rosslau_m$Freq,
113   gruppe_sum_altmarkkreis_salzwedel_f$Freq,
114   gruppe_sum_altmarkkreis_salzwedel_m$Freq,
115   gruppe_sum_stendal_f$Freq,
116   gruppe_sum_stendal_m$Freq,
117   gruppe_sum_boerde_f$Freq,
118   gruppe_sum_boerde_m$Freq,
119   gruppe_sum_jerichower_land_f$Freq,
120   gruppe_sum_jerichower_land_m$Freq,
121   gruppe_sum_harz_f$Freq,
122   gruppe_sum_harz_m$Freq,
123   gruppe_sum_salzlandkreis_f$Freq,
124   gruppe_sum_salzlandkreis_m$Freq,
125   gruppe_sum_anhalt_bitterfeld_f$Freq,
126   gruppe_sum_anhalt_bitterfeld_m$Freq,
127   gruppe_sum_wittenberg_f$Freq,
128   gruppe_sum_wittenberg_m$Freq,
129   gruppe_sum_mansfeld_suedharz_f$Freq,
130   gruppe_sum_mansfeld_suedharz_m$Freq,
131   gruppe_sum_saalekreis_f$Freq,
132   gruppe_sum_saalekreis_m$Freq,
133   gruppe_sum_burgenlandkreis_f$Freq,
134   gruppe_sum_burgenlandkreis_m$Freq,
135   gruppe_sum_alle_anderen_f$Freq,
136   gruppe_sum_alle_anderen_m$Freq
137   )
138 # Überschriften setzen
139 rownames(v_art_plot) <- c("Arbeitslose","Beschäftigte","Familienangehörige","
        Rentner","Sonstige")
140 colnames(v_art_plot) <- c(paste(rep(x=landkreise,times=1,each=2),rep(x=c("F","M
        "),times=14),sep="_"),"au ß erhalb_sachsen_anhalt_F","au ß erhalb_sachsen_
        anhalt_M")

142 # Prozentuale Ergebnismatrix erstellen
143 v_art_plot_proz <- round(x=prop.table(x=v_art_plot,margin=2)*100,digits=2)

145 ### PLOT BEGINNT ###
146 # Seitenabstand
147 par(mar=c(5,4,5,6.5))

149 # Y-Achse Markierung
```

```
150 ybereich <- pretty(x=c(0,100),n=20)

152 # Abstände der Barplots definieren
153 a1 <- c(0.1)
154 a2 <- c(0.8)
155 abstand <- c(a1,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,
        a2,a1,a2,a1,a2,a1,a2,a1)

157 # Plotten
158 balken3 <- barplot(height=v_art_plot_proz,main="Verteilung der
        Versicherungsarten nach den Landkreisen in Sachsen-Anhalt",axes=FALSE,
        axisnames=FALSE,col=farben,border=NA,space=abstand)

160 # Legende anzeigen
161 legend(x=44.5,y=100,legend=rownames(v_art_plot),fill=farben,cex=0.6,xpd=TRUE)

163 # Geschlecht anzeigen
164 text(x=balken3,y=-2.2,labels=rep(x=c("F","M"),n=15),xpd=TRUE,cex=0.7)

166 # Regionen anzeigen
167 mtext(text=c(landkreise,"au ß erhalb"),side=1,at=balken3[seq(from=1,to=length(
        balken3),by=2)]+0.7,line=0.8,cex=0.7)

169 # X-Achsen-Beschriftung
170 mtext(text="Regionen",side=1,line=3.3,cex=0.8)

172 # Y-Achse anzeigen
173 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.7,pos=-1.2)

175 # Y-Achsen-Beschriftung
176 mtext(text="Prozent",side=2,line=2.5,cex=0.8)

178 # Prozente anzeigen für Arbeitslose,Beschäftigte,Familienangehörige,Rentner,
        Sonstige
179 text(x=balken3+0.35,y=5,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz[1,],
        digits=1),"%",sep=""),cex=0.6,col="white")
180 text(x=balken3+0.35,y=20,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz[2,],
        digits=1),"%",sep=""),cex=0.6,col="white")
181 text(x=balken3+0.35,y=colSums(v_art_plot_proz[1:3,])-8,adj=1,xpd=TRUE,labels=
        paste(round(v_art_plot_proz[3,],digits=1),"%",sep=""),cex=0.6,col="black")
182 text(x=balken3+0.35,y=75,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz[4,],
        digits=1),"%",sep=""),cex=0.6,col="black")
183 text(x=balken3+0.35,y=99,adj=1,xpd=TRUE,labels=paste(round(v_art_plot_proz[5,],
        digits=1),"%",sep=""),cex=0.6,col="black")
```

Quelltext A.7: R-Quelltext 3.1-Geschlechterverteilung-Diagramm.R

```
### Barplot der Geschlechterverteilung (relative Häufigkeit) ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "GESCHLECHT" werden extrahiert
ambu_gender <- ambudaten[c(1,7)]

# Funktion um den prozentualen Anteil der Frauen und Männer für jeden Landkreis zu erhalten
gender_verteilung <- function(landkreis,ambu_gender) {

  # Verbinden der Stammdaten des Landkreises mit Alter und Geschlecht der Ambudaten
  tmp_landkreis <- unique(merge(landkreis,ambu_gender,by="PSEUDONYM"))

  # Prozentanteil für Frauen und Männer berechnen
  landkreis_data <- rbind(
     round(x=length(subset(tmp_landkreis,tmp_landkreis$GENDER=="M")[,1]) / length(tmp_landkreis$PSEUDONYM)*100,digits=1),
     round(x=length(subset(tmp_landkreis,tmp_landkreis$GENDER=="F")[,1]) / length(tmp_landkreis$PSEUDONYM)*100,digits=1)
  )

  # Rückgabe
  return(landkreis_data)
}

# Ergebnismatrix für den Plot erstellen
gender_plot <- cbind(
  gender_verteilung(magdeburg,ambu_gender),
  gender_verteilung(halle,ambu_gender),
  gender_verteilung(dessau_rosslau,ambu_gender),
  gender_verteilung(altmarkkreis_salzwedel,ambu_gender),
  gender_verteilung(stendal,ambu_gender),
  gender_verteilung(boerde,ambu_gender),
  gender_verteilung(jerichower_land,ambu_gender),
  gender_verteilung(harz,ambu_gender),
  gender_verteilung(salzlandkreis,ambu_gender),
  gender_verteilung(anhalt_bitterfeld,ambu_gender),
```

```
40    gender_verteilung(wittenberg,ambu_gender),
41    gender_verteilung(mansfeld_suedharz,ambu_gender),
42    gender_verteilung(saalekreis,ambu_gender),
43    gender_verteilung(burgenlandkreis,ambu_gender)
44    )
45  # Überschriften setzen
46  rownames(gender_plot) <- c("proz_male","proz_female")
47  colnames(gender_plot) <- landkreise

49  ### PLOT BEGINNT ###
50  # Seitenabstand
51  par(mar=c(9,5,7,7))

53  # Plotten
54  balken <- barplot(height=gender_plot,beside=FALSE,main="Geschlechterverteilung
        in den Landkreisen (Sachsen-Anhalt)",col=c(farben[3],farben[4]),axes=FALSE,
        axisnames=FALSE)

56  # Legende anzeigen
57  legend(x=17.5,y=100,legend=c("Frauen","Männer"),fill=c(farben[4],farben[3]),cex
        =0.8,xpd=TRUE)

59  # Prozente der Männer anzeigen
60  text(x=balken+0.35,y=25,labels=paste(gender_plot[1,],"%"),cex=0.8,xpd=TRUE,adj
        =1)

62  # Prozente der Frauen anzeigen
63  text(x=balken+0.35,y=70,labels=paste(gender_plot[2,],"%"),cex=0.8,xpd=TRUE,adj
        =1)

65  # Y-Achsenbereich festlegen und Schritte definieren
66  ybereich <- pretty(x=c(0,100),n=5)

68  # Y-Achse anzeigen
69  axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos
        =-0.35)

71  # Y-Achsen-Beschriftung
72  mtext(text="Prozent",side=2,line=3)

74  # X-Achsen-Beschriftung
75  text(x=balken+0.2,y=-4,labels=landkreise,cex=0.9,srt=35,xpd=TRUE,adj=1)
76  mtext(text="Regionen",side=1,line=6)
```

Quelltext A.8: R-Quelltext 4.1-Altersverteilung-Diagramm.R

```
### Barplot der Altersverteilung ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "ALTER", "GESCHLECHT" werden extrahiert
ambu_age <- ambudaten[c(1,6,7)]

# Funktion um das Durchschnitts- und Maximalalter der Männer und Frauen für
    jeden Landkreis zu erhalten
age_verteilung <- function(landkreis,ambu_age) {

  # Verbinden der Stammdaten des Landkreises mit Alter und Geschlecht der
      Ambudaten
  tmp_landkreis <- unique(merge(landkreis,ambu_age,by="PSEUDONYM"))

  # Prozentanteil für die Männer berechnen
  subset_m <- subset(x=tmp_landkreis$AGE,subset=(tmp_landkreis$GENDER=="M"))
  landkreis_data_m <- rbind(
    round(mean(subset_m)),
    max(subset_m) - round(mean(subset_m))
  )
  # Prozentanteil für die Frauen berechnen
  subset_f <- subset(x=tmp_landkreis$AGE,subset=(tmp_landkreis$GENDER=="F"))
  landkreis_data_f <- rbind(
    round(mean(subset_f)),
    max(subset_f) - round(mean(subset_f))
  )
  # Ergebnis für Männer und Frauen verbinden
  landkreis_data <- cbind(landkreis_data_m,landkreis_data_f)
  # Rückgabe
  return(landkreis_data)
}

# Ergebnismatrix für den Plot erstellen
age_plot <- cbind(
  age_verteilung(magdeburg,ambu_age),
  age_verteilung(halle,ambu_age),
  age_verteilung(dessau_rosslau,ambu_age),
  age_verteilung(altmarkkreis_salzwedel,ambu_age),
```

```
42    age_verteilung(stendal,ambu_age),
43    age_verteilung(boerde,ambu_age),
44    age_verteilung(jerichower_land,ambu_age),
45    age_verteilung(harz,ambu_age),
46    age_verteilung(salzlandkreis,ambu_age),
47    age_verteilung(anhalt_bitterfeld,ambu_age),
48    age_verteilung(wittenberg,ambu_age),
49    age_verteilung(mansfeld_suedharz,ambu_age),
50    age_verteilung(saalekreis,ambu_age),
51    age_verteilung(burgenlandkreis,ambu_age)
52    )
53  # Überschriften setzen
54  rownames(age_plot) <- c("mean", "max-mean")

56  ### PLOT BEGINNT ###
57  # Seitenabstand
58  par(mar=c(8,5,8,3))

60  # Abstände der Barplots
61  a1 <- c(0.3)
62  a2 <- c(1.3)
63  abstand <- c(a1,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,a2,a1,
        a2,a1,a2,a1,a2,a1)

65  # Plotten
66  balken<-barplot(height=age_plot,beside=FALSE,main="Altersverteilung in den
        Landkreisen (Sachsen-Anhalt)",col=farben[4],axes=FALSE,axisnames=FALSE,
        space=abstand,ylim=c(0,1.1*max(colSums(age_plot))))

68  # Legende anzeigen
69  legend(x=46.5,y=140,legend=c("maximales Alter","mittleres Alter"),fill=c(farben
        _2[2],farben_2[13]),cex=0.8,xpd=TRUE)

71  # Mittleres Alter anzeigen
72  text(x=balken,y=age_plot[1,]-3,labels=age_plot[1,],col=farben_2[13])

74  # Maximales Alter anzeigen
75  max_alter <- rbind(colSums(age_plot))
76  text(x=balken,y=max_alter+3,labels=max_alter,col=farben_2[2])

78  # Y-Achsenbereich festlegen und Schritte definieren
79  ybereich <- pretty(x=c(0,max(colSums(age_plot))),n=12)

81  # Y-Achse anzeigen
```

```
82 axis(side=2,at=ybereich,labels=ybereich,las=2,cex.axis=0.9,pos=-0.85)

84 # Y-Achsen-Beschriftung
85 mtext(text="Alter",side=2,line=2)

87 # X-Achsen-Beschriftung
88 mtext(text="Regionen",side=1,line=5.5)

90 # Geschlecht anzeigen
91 text(x=balken,y=-4,labels=rep(x=c("M","F"),times=14),xpd=TRUE,cex=0.8)

93 # Regionen anzeigen
94 mtext(text=landkreise,side=1,at=balken[seq(from=1,to=length(balken),by=2)]+0.7,
       line=2,cex=0.65)
```

Quelltext A.9: R-Quelltext 4.2-Altersverteilung-Tabelle.R

```
1  ### Absolute und prozentuale Tabelle der Altersverteilung ###

3  # Variablen setzen (Altersgruppen)
4  alter_unterteilung_breaks <- c(0,20,40,60,80,100,120)
5  alter_unterteilung_labels <- c("0-20 J","21-40 J","41-60 J","61-80 J","81-100 J
       ","101-120 J")

7  # initial laden und ausführen
8  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

10 # Ambudaten einlesen und bereinigen
11 ambudaten <- ambudaten_laden()

13 # Die Spalten "PSEUDONYM", "ALTER", "GESCHLECHT" werden extrahiert
14 ambu_age <- ambudaten[c(1,6,7)]

16 # Verbinden der einzelnen Landkreise mit den Ambudaten
17 halle_age <- unique(merge(halle,ambu_age,by="PSEUDONYM"))
18 saalekreis_age <- unique(merge(saalekreis,ambu_age,by="PSEUDONYM"))
19 mansfeld_suedharz_age <- unique(merge(mansfeld_suedharz,ambu_age,by="PSEUDONYM"
       ))
20 harz_age <- unique(merge(harz,ambu_age,by="PSEUDONYM"))
21 anhalt_bitterfeld_age <- unique(merge(anhalt_bitterfeld,ambu_age,by="PSEUDONYM"
       ))
22 salzlandkreis_age <- unique(merge(salzlandkreis,ambu_age,by="PSEUDONYM"))
23 burgenlandkreis_age <- unique(merge(burgenlandkreis,ambu_age,by="PSEUDONYM"))
24 wittenberg_age <- unique(merge(wittenberg,ambu_age,by="PSEUDONYM"))
```

```
25 dessau_rosslau_age <- unique(merge(dessau_rosslau,ambu_age,by="PSEUDONYM"))
26 stendal_age <- unique(merge(stendal,ambu_age,by="PSEUDONYM"))
27 altmarkkreis_salzwedel_age <- unique(merge(altmarkkreis_salzwedel,ambu_age,by="
       PSEUDONYM"))
28 magdeburg_age <- unique(merge(magdeburg,ambu_age,by="PSEUDONYM"))
29 boerde_age <- unique(merge(boerde,ambu_age,by="PSEUDONYM"))
30 jerichower_land_age <- unique(merge(jerichower_land,ambu_age,by="PSEUDONYM"))

32 # Gruppierung jedes Pseudonyms eines Landkreises in die Altersgruppen
33 halle_tmp2 <- cut(x=halle_age$AGE,breaks=alter_unterteilung_breaks,labels=alter
       _unterteilung_labels)
34 saalekreis_tmp2 <- cut(x=saalekreis_age$AGE,breaks=alter_unterteilung_breaks,
       labels=alter_unterteilung_labels)
35 mansfeld_suedharz_tmp2 <- cut(x=mansfeld_suedharz_age$AGE,breaks=alter_
       unterteilung_breaks,labels=alter_unterteilung_labels)
36 harz_tmp2 <- cut(x=harz_age$AGE,breaks=alter_unterteilung_breaks,labels=alter_
       unterteilung_labels)
37 anhalt_bitterfeld_tmp2 <- cut(x=anhalt_bitterfeld_age$AGE,breaks=alter_
       unterteilung_breaks,labels=alter_unterteilung_labels)
38 salzlandkreis_tmp2 <- cut(x=salzlandkreis_age$AGE,breaks=alter_unterteilung_
       breaks,labels=alter_unterteilung_labels)
39 burgenlandkreis_tmp2 <- cut(x=burgenlandkreis_age$AGE,breaks=alter_unterteilung
       _breaks,labels=alter_unterteilung_labels)
40 wittenberg_tmp2 <- cut(x=wittenberg_age$AGE,breaks=alter_unterteilung_breaks,
       labels=alter_unterteilung_labels)
41 dessau_rosslau_tmp2 <- cut(x=dessau_rosslau_age$AGE,breaks=alter_unterteilung_
       breaks,labels=alter_unterteilung_labels)
42 stendal_tmp2 <- cut(x=stendal_age$AGE,breaks=alter_unterteilung_breaks,labels=
       alter_unterteilung_labels)
43 altmarkkreis_salzwedel_tmp2 <- cut(x=altmarkkreis_salzwedel_age$AGE,breaks=
       alter_unterteilung_breaks,labels=alter_unterteilung_labels)
44 magdeburg_tmp2 <- cut(x=magdeburg_age$AGE,breaks=alter_unterteilung_breaks,
       labels=alter_unterteilung_labels)
45 boerde_tmp2 <- cut(x=boerde_age$AGE,breaks=alter_unterteilung_breaks,labels=
       alter_unterteilung_labels)
46 jerichower_land_tmp2 <- cut(x=jerichower_land_age$AGE,breaks=alter_unterteilung
       _breaks,labels=alter_unterteilung_labels)

48 # Ergebnismatrix erzeugen
49 final_landkreise <- cbind(
50   t(table(magdeburg_age$GENDER,magdeburg_tmp2)),
51   t(table(halle_age$GENDER,halle_tmp2)),
52   t(table(dessau_rosslau_age$GENDER,dessau_rosslau_tmp2)),
53   t(table(altmarkkreis_salzwedel_age$GENDER,altmarkkreis_salzwedel_tmp2)),
```

```
54    t(table(stendal_age$GENDER,stendal_tmp2)),
55    t(table(boerde_age$GENDER,boerde_tmp2)),
56    t(table(jerichower_land_age$GENDER,jerichower_land_tmp2)),
57    t(table(harz_age$GENDER,harz_tmp2)),
58    t(table(salzlandkreis_age$GENDER,salzlandkreis_tmp2)),
59    t(table(anhalt_bitterfeld_age$GENDER,anhalt_bitterfeld_tmp2)),
60    t(table(wittenberg_age$GENDER,wittenberg_tmp2)),
61    t(table(mansfeld_suedharz_age$GENDER,mansfeld_suedharz_tmp2)),
62    t(table(saalekreis_age$GENDER,saalekreis_tmp2)),
63    t(table(burgenlandkreis_age$GENDER,burgenlandkreis_tmp2))
64  )
65  # Überschriften setzen
66  colnames(final_landkreise) <- paste(rep(x=landkreise,times=1,each=2),rep(x=c("F
        ","M"),times=14),sep="_")

68  # Summieren der Männer und Frauen für jeweils einen Landkreis für die
        prozentuale Betrachtung
69  summe <- c(
70    sum(final_landkreise[,1],final_landkreise[,2]),
71    sum(final_landkreise[,3],final_landkreise[,4]),
72    sum(final_landkreise[,5],final_landkreise[,6]),
73    sum(final_landkreise[,7],final_landkreise[,8]),
74    sum(final_landkreise[,9],final_landkreise[,10]),
75    sum(final_landkreise[,11],final_landkreise[,12]),
76    sum(final_landkreise[,13],final_landkreise[,14]),
77    sum(final_landkreise[,15],final_landkreise[,16]),
78    sum(final_landkreise[,17],final_landkreise[,18]),
79    sum(final_landkreise[,19],final_landkreise[,20]),
80    sum(final_landkreise[,21],final_landkreise[,22]),
81    sum(final_landkreise[,23],final_landkreise[,24]),
82    sum(final_landkreise[,25],final_landkreise[,26]),
83    sum(final_landkreise[,27],final_landkreise[,28])
84  )

86  # Prozentualen Anteil jeder Altersgruppe berechnen (bezogen auf ihren Landkreis
        )
87  final_landkreise_proz <- NULL
88  for (i in 1:length(final_landkreise[1,])) {final_landkreise_proz <- cbind(final
        _landkreise_proz,final_landkreise[,i] * 100 / rep(x=summe,each=2)[i])}
89  # Überschriften setzen
90  colnames(final_landkreise_proz) <- paste(rep(x=landkreise,times=1,each=2),rep(x
        =c("F","M"),times=14),sep="_")
91  # Ergebnis auf 2 Nachkommastellen runden
92  final_landkreise_proz <- round(x=final_landkreise_proz,digits=3)
```

```
94  ### CSV Tabellen erzeugen ###
95  # Prozentuale Tabelle
96  tabelle <- cbind("Geschlecht"=rep(x=c("F","M"),times=14))
97  tabelle <- cbind(tabelle,round(x=t(final_landkreise_proz),digits=2))
98  tabelle[,2:7] <- paste(tabelle[,2:7],"%",sep="")
99  # Überschriften setzen
100 rownames(tabelle) <- c("Magdeburg","","Halle","","Dessau-Rosslau","","
        Altmarkkreis-Salzwedel","","Stendal","","Börde","","Jerichower Land","","
        Harz","","Salzlandkreis","","Anhalt-Bitterfeld","","Wittenberg","","
        Mansfeld-Südharz","","Saalekreis","","Burgenlandkreis","")
101 ### Tabelle als CSV auf Festplatte speichern (prozentuale Tabelle)
102 write.table(x=cbind(Landkreise=rownames(tabelle),tabelle),file="/Users/Florian/
        Documents/Bachelorarbeit/CSV/altersverteilung_pro_lk_prozentual.csv",sep=";
        ",row.names=FALSE,col.names=TRUE,fileEncoding="latin1")

104 # Absolute Tabelle
105 tabelle <- cbind("Geschlecht"=rep(x=c("F","M"),times=14))
106 tabelle <- cbind(tabelle,t(final_landkreise))
107 # Überschriften setzen
108 rownames(tabelle) <- c("Magdeburg","","Halle","","Dessau-Rosslau","","
        Altmarkkreis-Salzwedel","","Stendal","","Börde","","Jerichower Land","","
        Harz","","Salzlandkreis","","Anhalt-Bitterfeld","","Wittenberg","","
        Mansfeld-Südharz","","Saalekreis","","Burgenlandkreis","")
109 ### Tabelle als CSV auf Festplatte speichern (absolute Tabelle)
110 write.table(x=cbind(Landkreise=rownames(tabelle),tabelle),file="/Users/Florian/
        Documents/Bachelorarbeit/CSV/altersverteilung_pro_lk_absolut.csv",sep=";",
        row.names=FALSE,col.names=TRUE,fileEncoding="latin1")
```

Quelltext A.10: R-Quelltext 5.1-ICD-Verteilung-Diagramm.R

```
1  ### Barplot der Häufigkeitstabelle aller ICD-Codes ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Ambudaten einlesen und bereinigen
7  ambudaten <- ambudaten_laden()

9  # Die Spalten "PSEUDONYM", "ICD" werden extrahiert und doppelte Zeilen entfernt
10 ambu_icd <- unique(ambudaten[c(1,3)])

12 # Absolute Häufigkeitstabelle der ATC-Codes erstellen
13 icd_abs_h <- data.frame(table("ICD"=ambu_icd$ICD))
```

```
15 # Häufigkeitstabelle absteigend sortieren
16 icd_abs_h <- data.frame(icd_abs_h[order(-icd_abs_h$Freq),])
17
18 ### PLOT BEGINNT ###
19 # Seitenabstand
20 par(mar=c(5,6,3,2))
21
22 # Plotten
23 barplot(height=icd_abs_h[,2],yaxt="n",main="Häufigkeitsverteilung der ICD-Codes
       ",xlab="ICD Codes",ylab="Anzahl",ylim=c(0,1.1*max(icd_abs_h[,2])),space=0,
       col=farben[1],border=farben[1])
24
25 # Gesamtanzahl ICD-Codes anzeigen
26 mtext(text=paste("Gesamtanzahl an Erkrankungen (2012): ",format(x=length(ambu_
       icd$ICD),big.mark=".",scientific=FALSE),sep=""),side=3,cex=0.7,line=-0.5)
27
28 # Y-Achse definieren
29 ybereich <- pretty(x=c(0,max(icd_abs_h[,2])),n=10)
30
31 # Y-Achse anzeigen
32 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=FALSE)
       ,las=2,cex.axis=0.8,pos=-110)
33
34 # X-Achse definieren
35 xbereich <- round(x=seq(from=0,to=length(icd_abs_h$ICD),by=length(icd_abs_h$ICD
       )/10),digits=0)
36
37 # X-Achse anzeigen
38 axis(side=1,at=xbereich,cex.axis=0.8,pos=-6000)
```

Quelltext A.11: R-Quelltext 5.2-ICD-Top30-Verteilung-Diagramm.R

```
1 ### Barplot häufigsten 30 ICD-Codes ###
2
3 # Variable setzen (Anzahl der häufigsten ICD-Codes)
4 # default: 30
5 anzahl_icd_codes <- 30
6
7 # initial laden und ausführen
8 source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
9
10 # Ambudaten einlesen und bereinigen
11 ambudaten <- ambudaten_laden()
```

```
13 # Die Spalten "PSEUDONYM", "ICD" werden extrahiert und doppelte Zeilen entfernt
14 ambu_icd <- unique(ambudaten[c(1,3)])

16 # Absolute Häufigkeitstabelle der ICD-Codes erstellen
17 icd_abs_h <- data.frame(table("ICD"=ambu_icd$ICD))

19 # Häufigkeitstabelle absteigend sortieren
20 icd_abs_h <- data.frame(icd_abs_h[order(-icd_abs_h$Freq),])

22 # Die häufigsten ICD-Codes extrahieren
23 icd_bereich <- icd_abs_h[1:anzahl_icd_codes,]

25 ### PLOT BEGINNT ###
26 # Seitenabstand
27 par(mar=c(4,6,5.5,2))

29 # Plotten
30 plot <- barplot(height=icd_bereich$Freq,main=paste("Die häufigsten ",anzahl_icd
       _codes," ICD-Codes",sep=""),axes=FALSE,axisnames=FALSE,col=farben[4],ylim=c
       (0,1.1*max(icd_bereich$Freq)))

32 # Tabelle (bzw. Legende) erzeugen
33   # prozentuale Häufigkeit
34   rel <- icd_bereich$Freq[1:15] * 100 / length(ambu_icd$ICD)
35   rel <- round(x=rel,digits=2)
36   rel_legende <- paste(icd_bereich[1:15,1],"  (",format(x=rel,nsmall=2),"%",")
         ",sep="")
37 legend('topright',legend=rel_legende,title="ICD-Code | rel. Häufigkeit",cex
       =0.7)

39 # Anzahl jeder Säule anzeigen
40 text(x=plot,y=(icd_bereich$Freq+16000),srt=90,labels=format(x=icd_bereich$Freq,
       big.mark=".",scientific=FALSE),cex=0.7)

42 # ICD Codes jeder Säule anzeigen
43 text(x=plot,y=-15000,labels=icd_bereich$ICD,cex=0.6,srt=90,xpd=TRUE)

45 # Y-Achse Schritte definieren
46 ybereich <- pretty(x=c(0,max(icd_bereich$Freq)),n=9)

48 # Y-Achse anzeigen
49 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=FALSE)
       ,las=2,cex.axis=0.9,pos=-0.6)
```

```
51 # Y-Achsen-Beschriftung
52 mtext(text="Anzahl",side=2,line=4,cex=0.9)

54 # X-Achsen-Beschriftung
55 mtext(text="ICD Codes",side=1,line=2.6,cex=0.9)
```

Quelltext A.12: R-Quelltext 5.3-ICD-Kapitel-5Regionen-Diagramm.R

```
1  ### Barplot der ICD-Codes nach den 22 Kapiteln (nach den 5 Regionen in SA) ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Ambudaten einlesen und bereinigen
7  ambudaten <- ambudaten_laden()

9  # Die Spalten "PSEUDONYM", "ICD" werden extrahiert
10 ambu_icd <- ambudaten[c(1,3)]

12 # Sachsen-Anhalt in 5 Gebiete einteilen
13 norden <- rbind(altmarkkreis_salzwedel,stendal)
14 mitte <- rbind(boerde,magdeburg,jerichower_land,salzlandkreis)
15 osten <- rbind(wittenberg,dessau_rosslau,anhalt_bitterfeld)
16 sueden <- rbind(burgenlandkreis,saalekreis,halle)
17 westen <- rbind(harz,mansfeld_suedharz)

19 # Alle ICD-Codes nach Kapiteln einteilen
20 icd_alle <- c("A01-B99","C00-D48","D50-D90","E00-E90",
21                "F00-F99","G00-G99","H00-H59","H60-H95",
22              "I00-I99","J00-J99","K00-K93","L00-L99",
23              "M00-M99","N00-N99","O00-O99","P00-P96",
24              "Q00-Q99","R00-R99","S00-T98","V01-Y84",
25              "Z00-Z99","U00-U99"
26 )

28 # Funktion um ICD-Codes pro Gebiet auf die ICD-Kapitel zu reduzieren
29 icd_kapitel <- function(landkreis,ambu_icd) {

31   # Verbinden der Stammdaten des Landkreises mit Ambudaten (ICD-Codes)
32   tmp_landkreis <- unique(merge(landkreis,ambu_icd,by="PSEUDONYM"))

34   # ICD-Codes auf 3 Zeichen (Kapitelname) verkürzen
35   tmp_landkreis$ICD <- substr(x=tmp_landkreis$ICD,start=1,stop=3)
```

```
37    # Mehrfach vorkommende (verkürzte) ICD-Codes entfernen
38    tmp_landkreis <- unique(tmp_landkreis)

40    # Die Spalte "ICD" als String definieren
41    tmp_landkreis$ICD <- as.character(tmp_landkreis$ICD)

43    # Rückgabe
44    return(tmp_landkreis)
45  }

47  # Funktion um die verkürzten ICD-Codes nach den Kapiteln zu summieren
48  icd_sum_kapitel <- function(tabelle,spalte,gruppenvektor) {

50    # Variable definieren
51    rueckgabe <- NULL

53    # Filterung der ICD-Codes nach den ICD-Kapiteln
54    for (i in gruppenvektor) {
55      tmp_icd <- func_bereich(tabelle,spalte,i,"c")
56      # Die Anzahl (hier length) von jedem ICD-Kapitel speichern
57      rueckgabe <- rbind(rueckgabe,length(tmp_icd[,1]))
58    }

60    # Rückgabe
61    return(rueckgabe)
62  }


65  # Norden nach den ICD-Kapiteln gruppieren
66  tmp_norden <- icd_kapitel(norden,ambu_icd)
67  norden_icd <- icd_sum_kapitel(tmp_norden,4,icd_alle)
68  # Prozentuale Spalte hinzufügen
69  norden_icd <- cbind(norden_icd,round(x=norden_icd[,1]*100/sum(norden_icd[,1]),
        digits=2))
70  # Überschriften setzen
71  rownames(norden_icd) <- icd_alle
72  colnames(norden_icd) <- c("anzahl","rel_h")

74  # Osten nach den ICD-Kapiteln gruppieren
75  tmp_osten <- icd_kapitel(osten,ambu_icd)
76  osten_icd <- icd_sum_kapitel(tmp_osten,4,icd_alle)
77  # Prozentuale Spalte hinzufügen
78  osten_icd <- cbind(osten_icd,round(x=osten_icd[,1]*100/sum(osten_icd[,1]),
```

```
    digits=2))
# Überschriften setzen
rownames(osten_icd) <- icd_alle
colnames(osten_icd) <- c("anzahl","rel_h")

# Süden nach den ICD-Kapiteln gruppieren
tmp_sueden <- icd_kapitel(sueden,ambu_icd)
sueden_icd <- icd_sum_kapitel(tmp_sueden,4,icd_alle)
# Prozentuale Spalte hinzufügen
sueden_icd <- cbind(sueden_icd,round(x=sueden_icd[,1]*100/sum(sueden_icd[,1]),
    digits=2))
# Überschriften setzen
rownames(sueden_icd) <- icd_alle
colnames(sueden_icd) <- c("anzahl","rel_h")

# Westen nach den ICD-Kapiteln gruppieren
tmp_westen <- icd_kapitel(westen,ambu_icd)
westen_icd <- icd_sum_kapitel(tmp_westen,4,icd_alle)
# Prozentuale Spalte hinzufügen
westen_icd <- cbind(westen_icd,round(x=westen_icd[,1]*100/sum(westen_icd[,1]),
    digits=2))
# Überschriften setzen
rownames(westen_icd) <- icd_alle
colnames(westen_icd) <- c("anzahl","rel_h")

# Mitte nach den ICD-Kapiteln gruppieren
tmp_mitte <- icd_kapitel(mitte,ambu_icd)
mitte_icd <- icd_sum_kapitel(tmp_mitte,4,icd_alle)
# Prozentuale Spalte hinzufügen
mitte_icd <- cbind(mitte_icd,round(x=mitte_icd[,1]*100/sum(mitte_icd[,1]),
    digits=2))
# Überschriften setzen
rownames(mitte_icd) <- icd_alle
colnames(mitte_icd) <- c("anzahl","rel_h")


# Ergebnismatrix erstellen (Prozentwerte aller 5 Gebiete zusammenfügen)
alle_regionen <- cbind(norden_icd[,2],osten_icd[,2],sueden_icd[,2],westen_icd
    [,2],mitte_icd[,2])
# Überschriften setzen
rownames(alle_regionen) <- icd_alle
colnames(alle_regionen) <- c("proz_norden","proz_osten","proz_sueden","proz_
    westen","proz_mitte")
```

```
118 ## PLOT BEGINNT ###
119 # Seitenabstand
120 par(mar=c(7,4,7,3))
121
122 # Abstände der Balken
123 space1 <- rep(x=3,times=22)
124 space2 <- c(3.5,rep(x=3,times=21))
125 space3 <- c(4.0,rep(x=3,times=21))
126 space4 <- c(4.5,rep(x=3,times=21))
127 space5 <- c(5.0,rep(x=3,times=21))
128
129 # Plotten
130 balken <- barplot(height=alle_regionen[,1],col=farben[1],space=space1,border="
        black",axes=FALSE,axisnames=FALSE,main="ICD-10 Verteilung in Sachsen-Anhalt
        ",ylim=c(0,1.1*max(alle_regionen)))
131 balken <- barplot(height=alle_regionen[,2],col=farben[2],space=space2,border="
        black",add=TRUE,axes=FALSE,axisnames=FALSE)
132 balken <- barplot(height=alle_regionen[,3],col=farben[3],space=space3,border="
        black",add=TRUE,axes=FALSE,axisnames=FALSE)
133 balken <- barplot(height=alle_regionen[,4],col=farben[4],space=space4,border="
        black",add=TRUE,axes=FALSE,axisnames=FALSE)
134 balken <- barplot(height=alle_regionen[,5],col=farben[5],space=space5,border="
        black",add=TRUE,axes=FALSE,axisnames=FALSE)
135
136 # Legende anzeigen
137 legend(x=85,y=18.5,legend=c("Norden","Osten","Süden","Westen","Mitte"),fill=
        farben,cex=0.8,xpd=TRUE)
138
139 # X-Achse anzeigen
140 text(x=balken,y=-0.5,srt=35,adj=1,xpd=TRUE,labels=rownames(alle_regionen),cex
        =0.9)
141
142 # X-Achsen-Beschriftung
143 mtext(text="ICD Codes",side=1,line=4.5)
144
145 # Y-Achse Markierung
146 ybereich <- pretty(x=c(0,max(alle_regionen)),n=15)
147
148 # Y-Achse anzeigen
149 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.9,pos=1.5)
150
151 # Y-Achsen-Beschriftung
152 mtext(text="Prozent",side=2,line=1.25)
```

Quelltext A.13: R-Quelltext 5.4-ICD-Standardabweichung-Tabelle.R

```
### Barplot der ICD-Codes nach grö ß ter Standardabweichung ###

# Variable setzen
  # Mindestanzahl von ICD's pro einzelnem Landkreis
  # Default: 10
  mindestens <- 10

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]

# Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine Hä
    ufigkeitstabelle auf Basis der ICD-Spalte erstellen
halle_icd <- data.frame(table("ICD"=subset(unique(merge(halle,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
saalekreis_icd <- data.frame(table("ICD"=subset(unique(merge(saalekreis,ambu_
    icd,by="PSEUDONYM")),select=c(ICD))))
mansfeld_suedharz_icd <- data.frame(table("ICD"=subset(unique(merge(mansfeld_
    suedharz,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
harz_icd <- data.frame(table("ICD"=subset(unique(merge(harz,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
anhalt_bitterfeld_icd <- data.frame(table("ICD"=subset(unique(merge(anhalt_
    bitterfeld,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
salzlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(salzlandkreis,
    ambu_icd,by="PSEUDONYM")),select=c(ICD))))
burgenlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(
    burgenlandkreis,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
wittenberg_icd <- data.frame(table("ICD"=subset(unique(merge(wittenberg,ambu_
    icd,by="PSEUDONYM")),select=c(ICD))))
dessau_rosslau_icd <- data.frame(table("ICD"=subset(unique(merge(dessau_rosslau
    ,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
stendal_icd <- data.frame(table("ICD"=subset(unique(merge(stendal,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
altmarkkreis_salzwedel_icd <- data.frame(table("ICD"=subset(unique(merge(
    altmarkkreis_salzwedel,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
magdeburg_icd <- data.frame(table("ICD"=subset(unique(merge(magdeburg,ambu_icd,
    by="PSEUDONYM")),select=c(ICD))))
boerde_icd <- data.frame(table("ICD"=subset(unique(merge(boerde,ambu_icd,by="
```

```
    PSEUDONYM")),select=c(ICD))))
jerichower_land_icd <- data.frame(table("ICD"=subset(unique(merge(jerichower_
    land,ambu_icd,by="PSEUDONYM")),select=c(ICD))))

# Filtern der ICD-Codes die mindestens die gesetze Anzahl pro LK haben
if (mindestens != 1) {
  halle_icd <- subset(x=halle_icd,subset=(halle_icd$Freq >= mindestens))
  saalekreis_icd <- subset(x=saalekreis_icd,subset=(saalekreis_icd$Freq >=
      mindestens))
  mansfeld_suedharz_icd <- subset(x=mansfeld_suedharz_icd,subset=(mansfeld_
      suedharz_icd$Freq >= mindestens))
  harz_icd <- subset(x=harz_icd,subset=(harz_icd$Freq >= mindestens))
  anhalt_bitterfeld_icd <- subset(x=anhalt_bitterfeld_icd,subset=(anhalt_
      bitterfeld_icd$Freq >= mindestens))
  salzlandkreis_icd <- subset(x=salzlandkreis_icd,subset=(salzlandkreis_icd$
      Freq >= mindestens))
  burgenlandkreis_icd <- subset(x=burgenlandkreis_icd,subset=(burgenlandkreis_
      icd$Freq >= mindestens))
  wittenberg_icd <- subset(x=wittenberg_icd,subset=(wittenberg_icd$Freq >=
      mindestens))
  dessau_rosslau_icd <- subset(x=dessau_rosslau_icd,subset=(dessau_rosslau_icd$
      Freq >= mindestens))
  stendal_icd <- subset(x=stendal_icd,subset=(stendal_icd$Freq >= mindestens))
  altmarkkreis_salzwedel_icd <- subset(x=altmarkkreis_salzwedel_icd,subset=(
      altmarkkreis_salzwedel_icd$Freq >= mindestens))
  magdeburg_icd <- subset(x=magdeburg_icd,subset=(magdeburg_icd$Freq >=
      mindestens))
  boerde_icd <- subset(x=boerde_icd,subset=(boerde_icd$Freq >= mindestens))
  jerichower_land_icd <- subset(x=jerichower_land_icd,subset=(jerichower_land_
      icd$Freq >= mindestens))
}

# Liste erstellen mit allen ICDs der Landkreise
alle_kreise_icd <- list (magdeburg_icd,
            halle_icd,
            dessau_rosslau_icd,
            altmarkkreis_salzwedel_icd,
            stendal_icd,
            boerde_icd,
            jerichower_land_icd,
            harz_icd,
            salzlandkreis_icd,
            anhalt_bitterfeld_icd,
            wittenberg_icd,
```

```
63              mansfeld_suedharz_icd,
64              saalekreis_icd,
65              burgenlandkreis_icd)

67  # Reduce ist eine rekursive Funktion, die merge auf alle Elemente der Liste
        anwendet
68  gemeinsame_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_
        kreise_icd)

70  # ICD's entfernen, die nicht in allen LK mindentens X-mal vorkommen
71  halle_gemeinsame_icd <- halle_icd[halle_icd$ICD %in% gemeinsame_icd$ICD,]
72  saalekreis_gemeinsame_icd <- saalekreis_icd[saalekreis_icd$ICD %in% gemeinsame_
        icd$ICD,]
73  mansfeld_suedharz_gemeinsame_icd <- mansfeld_suedharz_icd[mansfeld_suedharz_icd
        $ICD %in% gemeinsame_icd$ICD,]
74  harz_gemeinsame_icd <- harz_icd[harz_icd$ICD %in% gemeinsame_icd$ICD,]
75  anhalt_bitterfeld_gemeinsame_icd <- anhalt_bitterfeld_icd[anhalt_bitterfeld_icd
        $ICD %in% gemeinsame_icd$ICD,]
76  salzlandkreis_gemeinsame_icd <- salzlandkreis_icd[salzlandkreis_icd$ICD %in%
        gemeinsame_icd$ICD,]
77  burgenlandkreis_gemeinsame_icd <- burgenlandkreis_icd[burgenlandkreis_icd$ICD %
        in% gemeinsame_icd$ICD,]
78  wittenberg_gemeinsame_icd <- wittenberg_icd[wittenberg_icd$ICD %in% gemeinsame_
        icd$ICD,]
79  dessau_rosslau_gemeinsame_icd <- dessau_rosslau_icd[dessau_rosslau_icd$ICD %in%
         gemeinsame_icd$ICD,]
80  stendal_gemeinsame_icd <- stendal_icd[stendal_icd$ICD %in% gemeinsame_icd$ICD,]
81  altmarkkreis_salzwedel_gemeinsame_icd <- altmarkkreis_salzwedel_icd[
        altmarkkreis_salzwedel_icd$ICD %in% gemeinsame_icd$ICD,]
82  magdeburg_gemeinsame_icd <- magdeburg_icd[magdeburg_icd$ICD %in% gemeinsame_icd
        $ICD,]
83  boerde_gemeinsame_icd <- boerde_icd[boerde_icd$ICD %in% gemeinsame_icd$ICD,]
84  jerichower_land_gemeinsame_icd <- jerichower_land_icd[jerichower_land_icd$ICD %
        in% gemeinsame_icd$ICD,]

86  # Die Spalte "pro_patient" jedem LK hinzufügen
87  # "pro_patient" beschreibt: 1 Fall auf X Versicherte
88  halle_gemeinsame_icd <- cbind(halle_gemeinsame_icd,"pro_patient"=length(halle$
        PSEUDONYM) / halle_gemeinsame_icd$Freq)
89  saalekreis_gemeinsame_icd <- cbind(saalekreis_gemeinsame_icd,"pro_patient"=
        length(saalekreis$PSEUDONYM) / saalekreis_gemeinsame_icd$Freq)
90  mansfeld_suedharz_gemeinsame_icd <- cbind(mansfeld_suedharz_gemeinsame_icd,"pro
        _patient"=length(mansfeld_suedharz$PSEUDONYM) / mansfeld_suedharz_
        gemeinsame_icd$Freq)
```

```
91  harz_gemeinsame_icd <- cbind(harz_gemeinsame_icd,"pro_patient"=length(harz$
        PSEUDONYM) / harz_gemeinsame_icd$Freq)
92  anhalt_bitterfeld_gemeinsame_icd <- cbind(anhalt_bitterfeld_gemeinsame_icd,"pro
        _patient"=length(anhalt_bitterfeld$PSEUDONYM) / anhalt_bitterfeld_
        gemeinsame_icd$Freq)
93  salzlandkreis_gemeinsame_icd <- cbind(salzlandkreis_gemeinsame_icd,"pro_patient
        "=length(salzlandkreis$PSEUDONYM) / salzlandkreis_gemeinsame_icd$Freq)
94  burgenlandkreis_gemeinsame_icd <- cbind(burgenlandkreis_gemeinsame_icd,"pro_
        patient"=length(burgenlandkreis$PSEUDONYM) / burgenlandkreis_gemeinsame_icd
        $Freq)
95  wittenberg_gemeinsame_icd <- cbind(wittenberg_gemeinsame_icd,"pro_patient"=
        length(wittenberg$PSEUDONYM) / wittenberg_gemeinsame_icd$Freq)
96  dessau_rosslau_gemeinsame_icd <- cbind(dessau_rosslau_gemeinsame_icd,"pro_
        patient"=length(dessau_rosslau$PSEUDONYM) / dessau_rosslau_gemeinsame_icd$
        Freq)
97  stendal_gemeinsame_icd <- cbind(stendal_gemeinsame_icd,"pro_patient"=length(
        stendal$PSEUDONYM) / stendal_gemeinsame_icd$Freq)
98  altmarkkreis_salzwedel_gemeinsame_icd <- cbind(altmarkkreis_salzwedel_
        gemeinsame_icd,"pro_patient"=length(altmarkkreis_salzwedel$PSEUDONYM) /
        altmarkkreis_salzwedel_gemeinsame_icd$Freq)
99  magdeburg_gemeinsame_icd <- cbind(magdeburg_gemeinsame_icd,"pro_patient"=length
        (magdeburg$PSEUDONYM) / magdeburg_gemeinsame_icd$Freq)
100 boerde_gemeinsame_icd <- cbind(boerde_gemeinsame_icd,"pro_patient"=length(
        boerde$PSEUDONYM) / boerde_gemeinsame_icd$Freq)
101 jerichower_land_gemeinsame_icd <- cbind(jerichower_land_gemeinsame_icd,"pro_
        patient"=length(jerichower_land$PSEUDONYM) / jerichower_land_gemeinsame_icd
        $Freq)

103 # Ergebnismatrix erzeugen (1 Fall auf X Versicherte)
104 matrix_pro_patient <- cbind(
105   magdeburg_gemeinsame_icd$pro_patient,
106   halle_gemeinsame_icd$pro_patient,
107   dessau_rosslau_gemeinsame_icd$pro_patient,
108   altmarkkreis_salzwedel_gemeinsame_icd$pro_patient,
109   stendal_gemeinsame_icd$pro_patient,
110   boerde_gemeinsame_icd$pro_patient,
111   jerichower_land_gemeinsame_icd$pro_patient,
112   harz_gemeinsame_icd$pro_patient,
113   salzlandkreis_gemeinsame_icd$pro_patient,
114   anhalt_bitterfeld_gemeinsame_icd$pro_patient,
115   wittenberg_gemeinsame_icd$pro_patient,
116   mansfeld_suedharz_gemeinsame_icd$pro_patient,
117   saalekreis_gemeinsame_icd$pro_patient,
118   burgenlandkreis_gemeinsame_icd$pro_patient
```

```
119 )

121 # Überschriften setzen
122 rownames(matrix_pro_patient) <- magdeburg_gemeinsame_icd$ICD

124 # Die Standardabweichung pro Zeile für jeden LK berechnen und als Spalte hinzuf
        ügen
125 matrix_pro_patient <- cbind(matrix_pro_patient,apply(X=matrix_pro_patient,
        MARGIN=1,FUN=sd2))
126 colnames(matrix_pro_patient) <- c(landkreise,"st_abw")

128 # Die Ergebnismatrix absteigend der Standardabweichung sortieren
129 matrix_pro_patient <- matrix_pro_patient[order(-matrix_pro_patient[,15]),]

131 ### Tabellen als CSV auf Festplatte speichern (1Fall auf X Versicherte)
132 write.table(x=cbind("ICD-Code"=rownames(matrix_pro_patient),matrix_pro_patient)
        ,file="/Users/Florian/Documents/Bachelorarbeit/CSV/ICD-1Fall_auf_X_
        Versicherte_pro_lk.csv",sep=";",row.names=FALSE,col.names=TRUE,fileEncoding
        ="latin1")

134 # matrix_pro_patient absteigend nach den ICD's sortieren und Zahlen runden
135 matrix_pro_patient_sort <- round(matrix_pro_patient[order(rownames(matrix_pro_
        patient)),],digits=2)
136 write.table(x=cbind("ICD-Code"=rownames(matrix_pro_patient_sort),matrix_pro_
        patient_sort),file="/Users/Florian/Documents/Bachelorarbeit/CSV/ICD-1Fall_
        auf_X_Versicherte_pro_lk_nach_ICD_sortiert.csv",sep=";",row.names=FALSE,col
        .names=TRUE,fileEncoding="latin1")
```

Quelltext A.14: R-Quelltext 5.5-ICD-Standardabweichung-Diagramm.R

```
1  ### Barplot der ICD-Codes nach grö ß ter Standardabweichung ###

3  # Variablen setzen
4    # Anzahl von ICD-Codes im Balkendiagramm
5    # Default: 30
6    anzahl_icd <- 30
7    # Mindestanzahl von ICD's pro einzelnem Landkreis
8    # Default: 10
9    mindestens <- 10

11 # initial laden und ausführen
12 source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

14 # Ambudaten einlesen und bereinigen
```

```
15 ambudaten <- ambudaten_laden()

17 # Die Spalten "PSEUDONYM", "ICD" werden extrahiert
18 ambu_icd <- ambudaten[c(1,3)]

20 # Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine Hä
       ufigkeitstabelle auf Basis der ICD-Spalte erstellen
21 halle_icd <- data.frame(table("ICD"=subset(unique(merge(halle,ambu_icd,by="
       PSEUDONYM")),select=c(ICD))))
22 saalekreis_icd <- data.frame(table("ICD"=subset(unique(merge(saalekreis,ambu_
       icd,by="PSEUDONYM")),select=c(ICD))))
23 mansfeld_suedharz_icd <- data.frame(table("ICD"=subset(unique(merge(mansfeld_
       suedharz,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
24 harz_icd <- data.frame(table("ICD"=subset(unique(merge(harz,ambu_icd,by="
       PSEUDONYM")),select=c(ICD))))
25 anhalt_bitterfeld_icd <- data.frame(table("ICD"=subset(unique(merge(anhalt_
       bitterfeld,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
26 salzlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(salzlandkreis,
       ambu_icd,by="PSEUDONYM")),select=c(ICD))))
27 burgenlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(
       burgenlandkreis,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
28 wittenberg_icd <- data.frame(table("ICD"=subset(unique(merge(wittenberg,ambu_
       icd,by="PSEUDONYM")),select=c(ICD))))
29 dessau_rosslau_icd <- data.frame(table("ICD"=subset(unique(merge(dessau_rosslau
       ,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
30 stendal_icd <- data.frame(table("ICD"=subset(unique(merge(stendal,ambu_icd,by="
       PSEUDONYM")),select=c(ICD))))
31 altmarkkreis_salzwedel_icd <- data.frame(table("ICD"=subset(unique(merge(
       altmarkkreis_salzwedel,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
32 magdeburg_icd <- data.frame(table("ICD"=subset(unique(merge(magdeburg,ambu_icd,
       by="PSEUDONYM")),select=c(ICD))))
33 boerde_icd <- data.frame(table("ICD"=subset(unique(merge(boerde,ambu_icd,by="
       PSEUDONYM")),select=c(ICD))))
34 jerichower_land_icd <- data.frame(table("ICD"=subset(unique(merge(jerichower_
       land,ambu_icd,by="PSEUDONYM")),select=c(ICD))))

36 # Filtern der ICD-Codes die mindestens die gesetzte Anzahl pro LK haben
37 if (mindestens != 1) {
38   halle_icd <- subset(x=halle_icd,subset=(halle_icd$Freq >= mindestens))
39   saalekreis_icd <- subset(x=saalekreis_icd,subset=(saalekreis_icd$Freq >=
         mindestens))
40   mansfeld_suedharz_icd <- subset(x=mansfeld_suedharz_icd,subset=(mansfeld_
         suedharz_icd$Freq >= mindestens))
41   harz_icd <- subset(x=harz_icd,subset=(harz_icd$Freq >= mindestens))
```

```
42    anhalt_bitterfeld_icd <- subset(x=anhalt_bitterfeld_icd,subset=(anhalt_
          bitterfeld_icd$Freq >= mindestens))
43    salzlandkreis_icd <- subset(x=salzlandkreis_icd,subset=(salzlandkreis_icd$
          Freq >= mindestens))
44    burgenlandkreis_icd <- subset(x=burgenlandkreis_icd,subset=(burgenlandkreis_
          icd$Freq >= mindestens))
45    wittenberg_icd <- subset(x=wittenberg_icd,subset=(wittenberg_icd$Freq >=
          mindestens))
46    dessau_rosslau_icd <- subset(x=dessau_rosslau_icd,subset=(dessau_rosslau_icd$
          Freq >= mindestens))
47    stendal_icd <- subset(x=stendal_icd,subset=(stendal_icd$Freq >= mindestens))
48    altmarkkreis_salzwedel_icd <- subset(x=altmarkkreis_salzwedel_icd,subset=(
          altmarkkreis_salzwedel_icd$Freq >= mindestens))
49    magdeburg_icd <- subset(x=magdeburg_icd,subset=(magdeburg_icd$Freq >=
          mindestens))
50    boerde_icd <- subset(x=boerde_icd,subset=(boerde_icd$Freq >= mindestens))
51    jerichower_land_icd <- subset(x=jerichower_land_icd,subset=(jerichower_land_
          icd$Freq >= mindestens))
52  }

54  # Liste erstellen mit allen ICDs der Landkreise
55  alle_kreise_icd <- list (magdeburg_icd,
56                           halle_icd,
57                           dessau_rosslau_icd,
58                           altmarkkreis_salzwedel_icd,
59                           stendal_icd,
60                           boerde_icd,
61                           jerichower_land_icd,
62                           harz_icd,
63                           salzlandkreis_icd,
64                           anhalt_bitterfeld_icd,
65                           wittenberg_icd,
66                           mansfeld_suedharz_icd,
67                           saalekreis_icd,
68                           burgenlandkreis_icd)

70  # Reduce ist eine rekursive Funktion, die merge auf alle Elemente der Liste
        anwendet
71  gemeinsame_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_
        kreise_icd)

73  # Aus den LK die ICD's entfernen, die nicht in allen LK mindestens X-mal
        vorkommen
74  halle_gemeinsame_icd <- halle_icd[halle_icd$ICD %in% gemeinsame_icd$ICD,]
```

```
75 saalekreis_gemeinsame_icd <- saalekreis_icd[saalekreis_icd$ICD %in% gemeinsame_
       icd$ICD,]
76 mansfeld_suedharz_gemeinsame_icd <- mansfeld_suedharz_icd[mansfeld_suedharz_icd
       $ICD %in% gemeinsame_icd$ICD,]
77 harz_gemeinsame_icd <- harz_icd[harz_icd$ICD %in% gemeinsame_icd$ICD,]
78 anhalt_bitterfeld_gemeinsame_icd <- anhalt_bitterfeld_icd[anhalt_bitterfeld_icd
       $ICD %in% gemeinsame_icd$ICD,]
79 salzlandkreis_gemeinsame_icd <- salzlandkreis_icd[salzlandkreis_icd$ICD %in%
       gemeinsame_icd$ICD,]
80 burgenlandkreis_gemeinsame_icd <- burgenlandkreis_icd[burgenlandkreis_icd$ICD %
       in% gemeinsame_icd$ICD,]
81 wittenberg_gemeinsame_icd <- wittenberg_icd[wittenberg_icd$ICD %in% gemeinsame_
       icd$ICD,]
82 dessau_rosslau_gemeinsame_icd <- dessau_rosslau_icd[dessau_rosslau_icd$ICD %in%
        gemeinsame_icd$ICD,]
83 stendal_gemeinsame_icd <- stendal_icd[stendal_icd$ICD %in% gemeinsame_icd$ICD,]
84 altmarkkreis_salzwedel_gemeinsame_icd <- altmarkkreis_salzwedel_icd[
       altmarkkreis_salzwedel_icd$ICD %in% gemeinsame_icd$ICD,]
85 magdeburg_gemeinsame_icd <- magdeburg_icd[magdeburg_icd$ICD %in% gemeinsame_icd
       $ICD,]
86 boerde_gemeinsame_icd <- boerde_icd[boerde_icd$ICD %in% gemeinsame_icd$ICD,]
87 jerichower_land_gemeinsame_icd <- jerichower_land_icd[jerichower_land_icd$ICD %
       in% gemeinsame_icd$ICD,]

89 # Die Spalte "pro_patient" jedem LK hinzufügen
90 # "pro_patient" beschreibt: 1 Fall auf X Versicherte
91 halle_gemeinsame_icd <- cbind(halle_gemeinsame_icd,"pro_patient"=length(halle$
       PSEUDONYM) / halle_gemeinsame_icd$Freq)
92 saalekreis_gemeinsame_icd <- cbind(saalekreis_gemeinsame_icd,"pro_patient"=
       length(saalekreis$PSEUDONYM) / saalekreis_gemeinsame_icd$Freq)
93 mansfeld_suedharz_gemeinsame_icd <- cbind(mansfeld_suedharz_gemeinsame_icd,"pro
       _patient"=length(mansfeld_suedharz$PSEUDONYM) / mansfeld_suedharz_
       gemeinsame_icd$Freq)
94 harz_gemeinsame_icd <- cbind(harz_gemeinsame_icd,"pro_patient"=length(harz$
       PSEUDONYM) / harz_gemeinsame_icd$Freq)
95 anhalt_bitterfeld_gemeinsame_icd <- cbind(anhalt_bitterfeld_gemeinsame_icd,"pro
       _patient"=length(anhalt_bitterfeld$PSEUDONYM) / anhalt_bitterfeld_
       gemeinsame_icd$Freq)
96 salzlandkreis_gemeinsame_icd <- cbind(salzlandkreis_gemeinsame_icd,"pro_patient
       "=length(salzlandkreis$PSEUDONYM) / salzlandkreis_gemeinsame_icd$Freq)
97 burgenlandkreis_gemeinsame_icd <- cbind(burgenlandkreis_gemeinsame_icd,"pro_
       patient"=length(burgenlandkreis$PSEUDONYM) / burgenlandkreis_gemeinsame_icd
       $Freq)
98 wittenberg_gemeinsame_icd <- cbind(wittenberg_gemeinsame_icd,"pro_patient"=
```

```
    length(wittenberg$PSEUDONYM) / wittenberg_gemeinsame_icd$Freq)
99  dessau_rosslau_gemeinsame_icd <- cbind(dessau_rosslau_gemeinsame_icd,"pro_
    patient"=length(dessau_rosslau$PSEUDONYM) / dessau_rosslau_gemeinsame_icd$
    Freq)
100 stendal_gemeinsame_icd <- cbind(stendal_gemeinsame_icd,"pro_patient"=length(
    stendal$PSEUDONYM) / stendal_gemeinsame_icd$Freq)
101 altmarkkreis_salzwedel_gemeinsame_icd <- cbind(altmarkkreis_salzwedel_
    gemeinsame_icd,"pro_patient"=length(altmarkkreis_salzwedel$PSEUDONYM) /
    altmarkkreis_salzwedel_gemeinsame_icd$Freq)
102 magdeburg_gemeinsame_icd <- cbind(magdeburg_gemeinsame_icd,"pro_patient"=length
    (magdeburg$PSEUDONYM) / magdeburg_gemeinsame_icd$Freq)
103 boerde_gemeinsame_icd <- cbind(boerde_gemeinsame_icd,"pro_patient"=length(
    boerde$PSEUDONYM) / boerde_gemeinsame_icd$Freq)
104 jerichower_land_gemeinsame_icd <- cbind(jerichower_land_gemeinsame_icd,"pro_
    patient"=length(jerichower_land$PSEUDONYM) / jerichower_land_gemeinsame_icd
    $Freq)

106 # Ergebnismatrix erzeugen (1 Fall auf X Versicherte)
107 matrix_pro_patient <- cbind(
108   magdeburg_gemeinsame_icd$pro_patient,
109   halle_gemeinsame_icd$pro_patient,
110   dessau_rosslau_gemeinsame_icd$pro_patient,
111   altmarkkreis_salzwedel_gemeinsame_icd$pro_patient,
112   stendal_gemeinsame_icd$pro_patient,
113   boerde_gemeinsame_icd$pro_patient,
114   jerichower_land_gemeinsame_icd$pro_patient,
115   harz_gemeinsame_icd$pro_patient,
116   salzlandkreis_gemeinsame_icd$pro_patient,
117   anhalt_bitterfeld_gemeinsame_icd$pro_patient,
118   wittenberg_gemeinsame_icd$pro_patient,
119   mansfeld_suedharz_gemeinsame_icd$pro_patient,
120   saalekreis_gemeinsame_icd$pro_patient,
121   burgenlandkreis_gemeinsame_icd$pro_patient
122 )

124 # Überschriften setzen
125 rownames(matrix_pro_patient) <- magdeburg_gemeinsame_icd$ICD

127 # Die Standardabweichung pro Zeile für jeden LK berechnen und als Spalte hinzuf
    ügen
128 matrix_pro_patient <- cbind(matrix_pro_patient,apply(X=matrix_pro_patient,
    MARGIN=1,FUN=sd2))
129 colnames(matrix_pro_patient) <- c(landkreise,"st_abw")
```

```
131 # Die Ergebnismatrix absteigend der Standardabweichung sortieren
132 matrix_pro_patient <- matrix_pro_patient[order(-matrix_pro_patient[,15]),]
133
134 # Ergebnismatrix erzeugen (absolute Häufigkeit)
135 matrix_abs_h <- cbind(
136   magdeburg_gemeinsame_icd$Freq,
137   halle_gemeinsame_icd$Freq,
138   dessau_rosslau_gemeinsame_icd$Freq,
139   altmarkkreis_salzwedel_gemeinsame_icd$Freq,
140   stendal_gemeinsame_icd$Freq,
141   boerde_gemeinsame_icd$Freq,
142   jerichower_land_gemeinsame_icd$Freq,
143   harz_gemeinsame_icd$Freq,
144   salzlandkreis_gemeinsame_icd$Freq,
145   anhalt_bitterfeld_gemeinsame_icd$Freq,
146   wittenberg_gemeinsame_icd$Freq,
147   mansfeld_suedharz_gemeinsame_icd$Freq,
148   saalekreis_gemeinsame_icd$Freq,
149   burgenlandkreis_gemeinsame_icd$Freq
150 )
151
152 # Überschriften setzen
153 rownames(matrix_abs_h) <- magdeburg_gemeinsame_icd$ICD
154
155 # Aufsummierte Häufigkeit als Spalte hinzufügen
156 matrix_abs_h <- cbind(matrix_abs_h,rowSums(matrix_abs_h))
157 colnames(matrix_abs_h) <- c(landkreise,"Freq_ges")
158
159 # ICD-Codes mit der grö ß ten Standardabweichnung werden ausgewählt und
        prozentual dargestellt
160 icd_plot <- NULL
161 for (i in 1:anzahl_icd) {icd_plot <- rbind(icd_plot,sum(matrix_pro_patient[i
        ,1:14]) / matrix_pro_patient[i,1:14])}
162 rownames(icd_plot) <- rownames(matrix_pro_patient[1:anzahl_icd,])
163 # "icd_plot" transponieren um ICDs als Spalten zu erhalten
164 icd_plot <- t(icd_plot)
165
166 # Prozentuale Tabelle erstellen
167 icd_plot_proz <- prop.table(x=icd_plot,margin=2)*100
168
169 ### PLOT BEGINNT ###
170 # Seitenabstand
171 par(mar=c(5,5,7,10))
```

```
173 # Plotten
174 icd_st_abw_plot <- barplot(height=icd_plot_proz,main=paste(anzahl_icd,"ICD-
        Codes mit der größten Standardabweichung"),col=farben_2,axes=FALSE,
        axisnames=FALSE)

176 # Legende anzeigen
177 legend(x=37.2,y=100,legend=landkreise,fill=farben_2,cex=0.9,xpd=TRUE)

179 # X-Achse anzeigen
180 text(x=icd_st_abw_plot+0.1,y=-1.8,srt=35,adj=1,xpd=TRUE,labels=colnames(icd_
        plot),cex=0.65)

182 # X-Achsen-Beschriftung
183 mtext(text="ICD-Codes",side=1,line=3)

185 # Y-Achse Markierung
186 ybereich <- pretty(x=c(0,100),n=20)

188 # Y-Achse anzeigen
189 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)

191 # Y-Achsen-Beschriftung
192 mtext(text="Prozent",side=2,line=3)

194 # Absolute Häufigkeit des ICD-Codes bestimmen und anzeigen
195 icd_plot_Freq <- NULL
196 for (i in 1:anzahl_icd) {
197   tmp <- subset(x=matrix_abs_h[,15],subset=(colnames(icd_plot_proz)[i] ==
          rownames(matrix_abs_h)))
198   icd_plot_Freq <- cbind(icd_plot_Freq,tmp[[1]])
199 }

201 text(x=icd_st_abw_plot+0.3,y=103,adj=1,xpd=TRUE,labels=icd_plot_Freq,cex=0.65,
        col=farben_2[2])
202 text(x=-0.5,y=103,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.75,col=
        farben_2[2])



205 ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit)
206 write.table(x=cbind("ICD-Code"=rownames(matrix_abs_h),matrix_abs_h),file="/
        Users/Florian/Documents/Bachelorarbeit/CSV/ICD10-absolute_Haeufigkeit_pro_
        lk.csv",sep=";",row.names=FALSE,col.names=TRUE,fileEncoding="latin1")
```

Quelltext A.15: R-Quelltext 5.6-ICD-Bereich-Prozentual-Diagramm.R

```
### Barplot der ICD-Codes nach grö ß ter Standardabweichung ###

# Variablen setzen
  # ICD-Bereich auswählen, der dargestellt wird
  icd_von <- "M5190"
  icd_bis <- "M5500"
  # Mindestanzahl von ICD's pro einzelnem Landkreis
  # Default: 10
  mindestens <- 10

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]

# Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine Hä
    ufigkeitstabelle auf Basis der ICD-Spalte erstellen
halle_icd <- data.frame(table("ICD"=subset(unique(merge(halle,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
saalekreis_icd <- data.frame(table("ICD"=subset(unique(merge(saalekreis,ambu_
    icd,by="PSEUDONYM")),select=c(ICD))))
mansfeld_suedharz_icd <- data.frame(table("ICD"=subset(unique(merge(mansfeld_
    suedharz,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
harz_icd <- data.frame(table("ICD"=subset(unique(merge(harz,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
anhalt_bitterfeld_icd <- data.frame(table("ICD"=subset(unique(merge(anhalt_
    bitterfeld,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
salzlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(salzlandkreis,
    ambu_icd,by="PSEUDONYM")),select=c(ICD))))
burgenlandkreis_icd <- data.frame(table("ICD"=subset(unique(merge(
    burgenlandkreis,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
wittenberg_icd <- data.frame(table("ICD"=subset(unique(merge(wittenberg,ambu_
    icd,by="PSEUDONYM")),select=c(ICD))))
dessau_rosslau_icd <- data.frame(table("ICD"=subset(unique(merge(dessau_rosslau
    ,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
stendal_icd <- data.frame(table("ICD"=subset(unique(merge(stendal,ambu_icd,by="
    PSEUDONYM")),select=c(ICD))))
altmarkkreis_salzwedel_icd <- data.frame(table("ICD"=subset(unique(merge(
    altmarkkreis_salzwedel,ambu_icd,by="PSEUDONYM")),select=c(ICD))))
```

```
32 magdeburg_icd <- data.frame(table("ICD"=subset(unique(merge(magdeburg,ambu_icd,
       by="PSEUDONYM")),select=c(ICD))))
33 boerde_icd <- data.frame(table("ICD"=subset(unique(merge(boerde,ambu_icd,by="
       PSEUDONYM")),select=c(ICD))))
34 jerichower_land_icd <- data.frame(table("ICD"=subset(unique(merge(jerichower_
       land,ambu_icd,by="PSEUDONYM")),select=c(ICD))))

36 # Filtern der ICD-Codes die mindestens die gesetze Anzahl pro LK haben
37 if (mindestens != 1) {
38   halle_icd <- subset(x=halle_icd,subset=(halle_icd$Freq >= mindestens))
39   saalekreis_icd <- subset(x=saalekreis_icd,subset=(saalekreis_icd$Freq >=
         mindestens))
40   mansfeld_suedharz_icd <- subset(x=mansfeld_suedharz_icd,subset=(mansfeld_
         suedharz_icd$Freq >= mindestens))
41   harz_icd <- subset(x=harz_icd,subset=(harz_icd$Freq >= mindestens))
42   anhalt_bitterfeld_icd <- subset(x=anhalt_bitterfeld_icd,subset=(anhalt_
         bitterfeld_icd$Freq >= mindestens))
43   salzlandkreis_icd <- subset(x=salzlandkreis_icd,subset=(salzlandkreis_icd$
         Freq >= mindestens))
44   burgenlandkreis_icd <- subset(x=burgenlandkreis_icd,subset=(burgenlandkreis_
         icd$Freq >= mindestens))
45   wittenberg_icd <- subset(x=wittenberg_icd,subset=(wittenberg_icd$Freq >=
         mindestens))
46   dessau_rosslau_icd <- subset(x=dessau_rosslau_icd,subset=(dessau_rosslau_icd$
         Freq >= mindestens))
47   stendal_icd <- subset(x=stendal_icd,subset=(stendal_icd$Freq >= mindestens))
48   altmarkkreis_salzwedel_icd <- subset(x=altmarkkreis_salzwedel_icd,subset=(
         altmarkkreis_salzwedel_icd$Freq >= mindestens))
49   magdeburg_icd <- subset(x=magdeburg_icd,subset=(magdeburg_icd$Freq >=
         mindestens))
50   boerde_icd <- subset(x=boerde_icd,subset=(_icdboerde_icd$Freq >= mindestens))
51   jerichower_land_icd <- subset(x=jerichower_land_icd,subset=(jerichower_land_
         icd$Freq >= mindestens))
52 }

54 # Liste erstellen mit allen ICDs der Landkreise
55 alle_kreise_icd <- list (magdeburg_icd,
56             halle_icd,
57             dessau_rosslau_icd,
58             altmarkkreis_salzwedel_icd,
59             stendal_icd,
60             boerde_icd,
61             jerichower_land_icd,
62             harz_icd,
```

```
63              salzlandkreis_icd,
64              anhalt_bitterfeld_icd,
65              wittenberg_icd,
66              mansfeld_suedharz_icd,
67              saalekreis_icd,
68              burgenlandkreis_icd)

70  # Reduce ist eine rekursive Funktion, die merge auf alle Elemente der Liste
        anwendet
71  gemeinsame_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_
        kreise_icd)

73  # Aus den LK die ICD's entfernen, die nicht in allen LK mindentens X-mal
        vorkommen
74  halle_gemeinsame_icd <- halle_icd[halle_icd$ICD %in% gemeinsame_icd$ICD,]
75  saalekreis_gemeinsame_icd <- saalekreis_icd[saalekreis_icd$ICD %in% gemeinsame_
        icd$ICD,]
76  mansfeld_suedharz_gemeinsame_icd <- mansfeld_suedharz_icd[mansfeld_suedharz_icd
        $ICD %in% gemeinsame_icd$ICD,]
77  harz_gemeinsame_icd <- harz_icd[harz_icd$ICD %in% gemeinsame_icd$ICD,]
78  anhalt_bitterfeld_gemeinsame_icd <- anhalt_bitterfeld_icd[anhalt_bitterfeld_icd
        $ICD %in% gemeinsame_icd$ICD,]
79  salzlandkreis_gemeinsame_icd <- salzlandkreis_icd[salzlandkreis_icd$ICD %in%
        gemeinsame_icd$ICD,]
80  burgenlandkreis_gemeinsame_icd <- burgenlandkreis_icd[burgenlandkreis_icd$ICD %
        in% gemeinsame_icd$ICD,]
81  wittenberg_gemeinsame_icd <- wittenberg_icd[wittenberg_icd$ICD %in% gemeinsame_
        icd$ICD,]
82  dessau_rosslau_gemeinsame_icd <- dessau_rosslau_icd[dessau_rosslau_icd$ICD %in%
         gemeinsame_icd$ICD,]
83  stendal_gemeinsame_icd <- stendal_icd[stendal_icd$ICD %in% gemeinsame_icd$ICD,]
84  altmarkkreis_salzwedel_gemeinsame_icd <- altmarkkreis_salzwedel_icd[
        altmarkkreis_salzwedel_icd$ICD %in% gemeinsame_icd$ICD,]
85  magdeburg_gemeinsame_icd <- magdeburg_icd[magdeburg_icd$ICD %in% gemeinsame_icd
        $ICD,]
86  boerde_gemeinsame_icd <- boerde_icd[boerde_icd$ICD %in% gemeinsame_icd$ICD,]
87  jerichower_land_gemeinsame_icd <- jerichower_land_icd[jerichower_land_icd$ICD %
        in% gemeinsame_icd$ICD,]

89  # Die Spalte "pro_patient" jedem LK hinzufügen
90  # "pro_patient" beschreibt: 1 Fall auf X Versicherte
91  halle_gemeinsame_icd <- cbind(halle_gemeinsame_icd,"pro_patient"=length(halle$
        PSEUDONYM) / halle_gemeinsame_icd$Freq)
92  saalekreis_gemeinsame_icd <- cbind(saalekreis_gemeinsame_icd,"pro_patient"=
```

```
    length(saalekreis$PSEUDONYM) / saalekreis_gemeinsame_icd$Freq)
93  mansfeld_suedharz_gemeinsame_icd <- cbind(mansfeld_suedharz_gemeinsame_icd,"pro
    _patient"=length(mansfeld_suedharz$PSEUDONYM) / mansfeld_suedharz_
    gemeinsame_icd$Freq)
94  harz_gemeinsame_icd <- cbind(harz_gemeinsame_icd,"pro_patient"=length(harz$
    PSEUDONYM) / harz_gemeinsame_icd$Freq)
95  anhalt_bitterfeld_gemeinsame_icd <- cbind(anhalt_bitterfeld_gemeinsame_icd,"pro
    _patient"=length(anhalt_bitterfeld$PSEUDONYM) / anhalt_bitterfeld_
    gemeinsame_icd$Freq)
96  salzlandkreis_gemeinsame_icd <- cbind(salzlandkreis_gemeinsame_icd,"pro_patient
    "=length(salzlandkreis$PSEUDONYM) / salzlandkreis_gemeinsame_icd$Freq)
97  burgenlandkreis_gemeinsame_icd <- cbind(burgenlandkreis_gemeinsame_icd,"pro_
    patient"=length(burgenlandkreis$PSEUDONYM) / burgenlandkreis_gemeinsame_icd
    $Freq)
98  wittenberg_gemeinsame_icd <- cbind(wittenberg_gemeinsame_icd,"pro_patient"=
    length(wittenberg$PSEUDONYM) / wittenberg_gemeinsame_icd$Freq)
99  dessau_rosslau_gemeinsame_icd <- cbind(dessau_rosslau_gemeinsame_icd,"pro_
    patient"=length(dessau_rosslau$PSEUDONYM) / dessau_rosslau_gemeinsame_icd$
    Freq)
100 stendal_gemeinsame_icd <- cbind(stendal_gemeinsame_icd,"pro_patient"=length(
    stendal$PSEUDONYM) / stendal_gemeinsame_icd$Freq)
101 altmarkkreis_salzwedel_gemeinsame_icd <- cbind(altmarkkreis_salzwedel_
    gemeinsame_icd,"pro_patient"=length(altmarkkreis_salzwedel$PSEUDONYM) /
    altmarkkreis_salzwedel_gemeinsame_icd$Freq)
102 magdeburg_gemeinsame_icd <- cbind(magdeburg_gemeinsame_icd,"pro_patient"=length
    (magdeburg$PSEUDONYM) / magdeburg_gemeinsame_icd$Freq)
103 boerde_gemeinsame_icd <- cbind(boerde_gemeinsame_icd,"pro_patient"=length(
    boerde$PSEUDONYM) / boerde_gemeinsame_icd$Freq)
104 jerichower_land_gemeinsame_icd <- cbind(jerichower_land_gemeinsame_icd,"pro_
    patient"=length(jerichower_land$PSEUDONYM) / jerichower_land_gemeinsame_icd
    $Freq)

106 # Ergebnismatrix erzeugen (1 Fall auf X Versicherte)
107 matrix_pro_patient <- cbind(
108   magdeburg_gemeinsame_icd$pro_patient,
109   halle_gemeinsame_icd$pro_patient,
110   dessau_rosslau_gemeinsame_icd$pro_patient,
111   altmarkkreis_salzwedel_gemeinsame_icd$pro_patient,
112   stendal_gemeinsame_icd$pro_patient,
113   boerde_gemeinsame_icd$pro_patient,
114   jerichower_land_gemeinsame_icd$pro_patient,
115   harz_gemeinsame_icd$pro_patient,
116   salzlandkreis_gemeinsame_icd$pro_patient,
117   anhalt_bitterfeld_gemeinsame_icd$pro_patient,
```

```
118   wittenberg_gemeinsame_icd$pro_patient,
119   mansfeld_suedharz_gemeinsame_icd$pro_patient,
120   saalekreis_gemeinsame_icd$pro_patient,
121   burgenlandkreis_gemeinsame_icd$pro_patient
122 )

124 # Überschriften setzen
125 rownames(matrix_pro_patient) <- magdeburg_gemeinsame_icd$ICD

127 # Die Standardabweichung pro Zeile für jeden LK berechnen und als Spalte hinzuf
        ügen
128 matrix_pro_patient <- cbind(matrix_pro_patient,apply(X=matrix_pro_patient,
        MARGIN=1,FUN=sd2))
129 colnames(matrix_pro_patient) <- c(landkreise,"st_abw")

131 # Die Ergebnismatrix absteigend der Standardabweichung sortieren
132 matrix_pro_patient <- matrix_pro_patient[order(-matrix_pro_patient[,15]),]

134 # Ergebnismatrix erzeugen (absolute Häufigkeit)
135 matrix_abs_h <- cbind(
136   magdeburg_gemeinsame_icd$Freq,
137   halle_gemeinsame_icd$Freq,
138   dessau_rosslau_gemeinsame_icd$Freq,
139   altmarkkreis_salzwedel_gemeinsame_icd$Freq,
140   stendal_gemeinsame_icd$Freq,
141   boerde_gemeinsame_icd$Freq,
142   jerichower_land_gemeinsame_icd$Freq,
143   harz_gemeinsame_icd$Freq,
144   salzlandkreis_gemeinsame_icd$Freq,
145   anhalt_bitterfeld_gemeinsame_icd$Freq,
146   wittenberg_gemeinsame_icd$Freq,
147   mansfeld_suedharz_gemeinsame_icd$Freq,
148   saalekreis_gemeinsame_icd$Freq,
149   burgenlandkreis_gemeinsame_icd$Freq
150 )

152 # Überschriften setzen
153 rownames(matrix_abs_h) <- magdeburg_gemeinsame_icd$ICD

155 # Aufsummierte Häufigkeit als Spalte hinzufügen
156 matrix_abs_h <- cbind(matrix_abs_h,rowSums(matrix_abs_h))
157 colnames(matrix_abs_h) <- c(landkreise,"Freq_ges")

159 ### ICD Bereich darstellen
```

```
160 # Matrix absteigend sortieren für den "von bis" Bereich
161 matrix_pro_patient_rowname_sort <- matrix_pro_patient[order(rownames(matrix_pro
        _patient)),]
162 icd_bereich <- subset(x=matrix_pro_patient_rowname_sort,subset=(rownames(matrix
        _pro_patient_rowname_sort) >= icd_von & rownames(matrix_pro_patient_rowname
        _sort) <= icd_bis))[,1:14]

164 # ICD's im definierten Bereich auswählen und prozentual darstellen
165 icd_plot <- NULL
166 for (i in 1:length(icd_bereich[,1])) {icd_plot <- rbind(icd_plot,sum(icd_
        bereich[i,]) / icd_bereich[i,]) }
167 rownames(icd_plot) <- rownames(icd_bereich)
168 # "icd_plot" transponieren um ICDs als Spalten zu bekommen
169 icd_plot <- t(icd_plot)

171 # Prozentuale Tabelle erstellen
172 icd_plot_proz <- prop.table(x=icd_plot,margin=2)*100

174 ### PLOT BEGINNT ###
175 # Seitenabstand
176 par(mar=c(5,5,7,10))

178 # Y-Achse Markierung
179 ybereich <- pretty(x=c(0,100),n=20)

181 # Plotten
182 icd_bereich_plot <- barplot(height=icd_plot_proz,main=paste("Prozentuale Hä
        ufigkeit der ICD Codes",icd_von,"-",icd_bis,"in den Landkreisen von Sachsen
        -Anhalt"),col=farben_2,axes=FALSE,axisnames=FALSE)

184 # Legende anzeigen
185 legend(x=max(icd_bereich_plot)+1,y=100,legend=landkreise,fill=farben_2,cex=0.9,
        xpd=TRUE)

187 # X-Achse anzeigen
188 text(x=icd_bereich_plot+0.1,y=-1.8,srt=35,adj=1,xpd=TRUE,labels=colnames(icd_
        plot_proz),cex=0.65)

190 # X-Achsen-Beschriftung
191 mtext(text="ICD-Codes",side=1,line=3)

193 # Y-Achse anzeigen
194 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)
```

```
196 # Y-Achsen-Beschriftung
197 mtext(text="Prozent",side=2,line=2)

199 # Absolute Häufigkeit anzeigen
200   # absolute Häufigkeit der ICD Codes in diesem Barplot bestimmen
201   icd_plot_Freq <- NULL
202   for (i in 1:length(icd_bereich[,1])) {
203     tmp <- subset(x=matrix_abs_h[,15],subset=(colnames(icd_plot_proz)[i] ==
          rownames(matrix_abs_h)))
204     icd_plot_Freq <- cbind(icd_plot_Freq,tmp[[1]])
205   }
206 text(x=icd_bereich_plot+0.35,y=103,adj=1,xpd=TRUE,labels=icd_plot_Freq,cex
      =0.65,col=farben_2[2])
207 text(x=-0.5,y=103,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.75,col=
      farben_2[2])
```

Quelltext A.16: R-Quelltext 5.7-ICD-Absolute-Haeufigkeit-Tabelle.R

```
1  ### Absolute Tabelle der ICD-Codes, absteigend nach Anzahl LK sortiert ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Ambudaten einlesen und bereinigen
7  ambudaten <- ambudaten_laden()

9  # Die Spalten "PSEUDONYM", "ICD" werden extrahiert
10 ambu_icd <- ambudaten[c(1,3)]

12 # Verbinden der einzelnen Landkreise mit den Ambudaten
13 halle_icd <- unique(merge(halle,ambu_icd,by="PSEUDONYM"))
14 saalekreis_icd <- unique(merge(saalekreis,ambu_icd,by="PSEUDONYM"))
15 mansfeld_suedharz_icd <- unique(merge(mansfeld_suedharz,ambu_icd,by="PSEUDONYM"
       ))
16 harz_icd <- unique(merge(harz,ambu_icd,by="PSEUDONYM"))
17 anhalt_bitterfeld_icd <- unique(merge(anhalt_bitterfeld,ambu_icd,by="PSEUDONYM"
       ))
18 salzlandkreis_icd <- unique(merge(salzlandkreis,ambu_icd,by="PSEUDONYM"))
19 burgenlandkreis_icd <- unique(merge(burgenlandkreis,ambu_icd,by="PSEUDONYM"))
20 wittenberg_icd <- unique(merge(wittenberg,ambu_icd,by="PSEUDONYM"))
21 dessau_rosslau_icd <- unique(merge(dessau_rosslau,ambu_icd,by="PSEUDONYM"))
22 stendal_icd <- unique(merge(stendal,ambu_icd,by="PSEUDONYM"))
23 altmarkkreis_salzwedel_icd <- unique(merge(altmarkkreis_salzwedel,ambu_icd,by="
```

```
    PSEUDONYM"))
magdeburg_icd <- unique(merge(magdeburg,ambu_icd,by="PSEUDONYM"))
boerde_icd <- unique(merge(boerde,ambu_icd,by="PSEUDONYM"))
jerichower_land_icd <- unique(merge(jerichower_land,ambu_icd,by="PSEUDONYM"))

# Verbinden aller Landkreise mit den Ambudaten
alle_kreise_icd <- rbind(halle_icd,saalekreis_icd,mansfeld_suedharz_icd,harz_
    icd,anhalt_bitterfeld_icd,salzlandkreis_icd,burgenlandkreis_icd,wittenberg_
    icd,dessau_rosslau_icd,stendal_icd,altmarkkreis_salzwedel_icd,magdeburg_icd
    ,boerde_icd,jerichower_land_icd)
# ODER: alle_kreise_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))

# Data.frame mit allen ICD-Codes in Sachsen-Anhalt erstellen
alle_icd <- data.frame("ICD"=unique(alle_kreise_icd$ICD))
# Die Spalte "Freq" hinzufügen und null setzen
alle_icd$Freq <- 0

# Absolute Häufigkeitstabelle der ICD-Codes pro Landkreis
halle_icd_freq <- data.frame(table(halle_icd$ICD))
saalekreis_icd_freq <- data.frame(table(saalekreis_icd$ICD))
mansfeld_suedharz_icd_freq <- data.frame(table(mansfeld_suedharz_icd$ICD))
harz_icd_freq <- data.frame(table(harz_icd$ICD))
anhalt_bitterfeld_icd_freq <- data.frame(table(anhalt_bitterfeld_icd$ICD))
salzlandkreis_icd_freq <- data.frame(table(salzlandkreis_icd$ICD))
burgenlandkreis_icd_freq <- data.frame(table(burgenlandkreis_icd$ICD))
wittenberg_icd_freq <- data.frame(table(wittenberg_icd$ICD))
dessau_rosslau_icd_freq <- data.frame(table(dessau_rosslau_icd$ICD))
stendal_icd_freq <- data.frame(table(stendal_icd$ICD))
altmarkkreis_salzwedel_icd_freq <- data.frame(table(altmarkkreis_salzwedel_icd$
    ICD))
magdeburg_icd_freq <- data.frame(table(magdeburg_icd$ICD))
boerde_icd_freq <- data.frame(table(boerde_icd$ICD))
jerichower_land_icd_freq <- data.frame(table(jerichower_land_icd$ICD))

# Überschriften setzen
colnames(halle_icd_freq) <- c("ICD","Freq")
colnames(saalekreis_icd_freq) <- c("ICD","Freq")
colnames(mansfeld_suedharz_icd_freq) <- c("ICD","Freq")
colnames(harz_icd_freq) <- c("ICD","Freq")
colnames(anhalt_bitterfeld_icd_freq) <- c("ICD","Freq")
colnames(salzlandkreis_icd_freq) <- c("ICD","Freq")
colnames(burgenlandkreis_icd_freq) <- c("ICD","Freq")
colnames(wittenberg_icd_freq) <- c("ICD","Freq")
colnames(dessau_rosslau_icd_freq) <- c("ICD","Freq")
```

```
63 colnames(stendal_icd_freq) <- c("ICD","Freq")
64 colnames(altmarkkreis_salzwedel_icd_freq) <- c("ICD","Freq")
65 colnames(magdeburg_icd_freq) <- c("ICD","Freq")
66 colnames(boerde_icd_freq) <- c("ICD","Freq")
67 colnames(jerichower_land_icd_freq) <- c("ICD","Freq")

69 # Ermittlung der fehlenden ICD-Codes pro Landkreis
70 # Anhängen der fehlenden Codes (mit Freq=0)
71 # Somit haben alle data.frames jedes LK gleiche Anzahl Zeilen
72 icd_temp <- alle_icd[!alle_icd$ICD %in% halle_icd_freq$ICD,]
73 halle_icd_freq <- rbind(halle_icd_freq,icd_temp)

75 icd_temp <- alle_icd[!alle_icd$ICD %in% saalekreis_icd_freq$ICD,]
76 saalekreis_icd_freq <- rbind(saalekreis_icd_freq,icd_temp)

78 icd_temp <- alle_icd[!alle_icd$ICD %in% mansfeld_suedharz_icd_freq$ICD,]
79 mansfeld_suedharz_icd_freq <- rbind(mansfeld_suedharz_icd_freq,icd_temp)

81 icd_temp <- alle_icd[!alle_icd$ICD %in% harz_icd_freq$ICD,]
82 harz_icd_freq <- rbind(harz_icd_freq,icd_temp)

84 icd_temp <- alle_icd[!alle_icd$ICD %in% anhalt_bitterfeld_icd_freq$ICD,]
85 anhalt_bitterfeld_icd_freq <- rbind(anhalt_bitterfeld_icd_freq,icd_temp)

87 icd_temp <- alle_icd[!alle_icd$ICD %in% salzlandkreis_icd_freq$ICD,]
88 salzlandkreis_icd_freq <- rbind(salzlandkreis_icd_freq,icd_temp)

90 icd_temp <- alle_icd[!alle_icd$ICD %in% burgenlandkreis_icd_freq$ICD,]
91 burgenlandkreis_icd_freq <- rbind(burgenlandkreis_icd_freq,icd_temp)

93 icd_temp <- alle_icd[!alle_icd$ICD %in% wittenberg_icd_freq$ICD,]
94 wittenberg_icd_freq <- rbind(wittenberg_icd_freq,icd_temp)

96 icd_temp <- alle_icd[!alle_icd$ICD %in% dessau_rosslau_icd_freq$ICD,]
97 dessau_rosslau_icd_freq <- rbind(dessau_rosslau_icd_freq,icd_temp)

99 icd_temp <- alle_icd[!alle_icd$ICD %in% stendal_icd_freq$ICD,]
100 stendal_icd_freq <- rbind(stendal_icd_freq,icd_temp)

102 icd_temp <- alle_icd[!alle_icd$ICD %in% altmarkkreis_salzwedel_icd_freq$ICD,]
103 altmarkkreis_salzwedel_icd_freq <- rbind(altmarkkreis_salzwedel_icd_freq,icd_
        temp)

105 icd_temp <- alle_icd[!alle_icd$ICD %in% magdeburg_icd_freq$ICD,]
```

```
106 magdeburg_icd_freq <- rbind(magdeburg_icd_freq,icd_temp)

108 icd_temp <- alle_icd[!alle_icd$ICD %in% boerde_icd_freq$ICD,]
109 boerde_icd_freq <- rbind(boerde_icd_freq,icd_temp)

111 icd_temp <- alle_icd[!alle_icd$ICD %in% jerichower_land_icd_freq$ICD,]
112 jerichower_land_icd_freq <- rbind(jerichower_land_icd_freq,icd_temp)

114 # Liste erstellen mit allen ICDs der Landkreise
115 alle_icd_list <- list ( magdeburg_icd_freq,
116             halle_icd_freq,
117             dessau_rosslau_icd_freq,
118             altmarkkreis_salzwedel_icd_freq,
119             stendal_icd_freq,
120             boerde_icd_freq,
121             jerichower_land_icd_freq,
122             harz_icd_freq,
123             salzlandkreis_icd_freq,
124             anhalt_bitterfeld_icd_freq,
125             wittenberg_icd_freq,
126             mansfeld_suedharz_icd_freq,
127             saalekreis_icd_freq,
128             burgenlandkreis_icd_freq
129 )

131 # Reduce ist eine Funktion, die merge auf alle Elemente der Liste anwendet
132 matrix_alle_icd <- Reduce(function(df1,df2) {merge(df1,df2,by="ICD")},alle_icd_
        list)

134 # Die Spalten 2 bis 15 in numeric umwandeln
135 matrix_alle_icd[,2:15] <- sapply(X=matrix_alle_icd[,2:15],FUN=as.numeric)

137 # Gesamt-Häufigkeit (Freq_ges) hinzufügen
138 matrix_alle_icd <- cbind(matrix_alle_icd,"Freq_ges"=rowSums(matrix_alle_icd
        [,2:15]))

140 # Landkreis-Häufigkeit (Freq_lk) hinzufügen
141 for (i in 1:length(matrix_alle_icd[,1])) {matrix_alle_icd[i,17] <- 14-sum(
        matrix_alle_icd[i,2:15]==0)}
142 # Überschriften setzen
143 colnames(matrix_alle_icd) <- c("ICD",landkreise,"Freq_ges","Freq_lk")

145 # Gesamt-Häufigkeit absteigend sortiert und bei gleicher Häufigkeit, die
        Landkreis-Häufigkeit aufsteigend sortiert
```

```
146 matrix_alle_icd <- matrix_alle_icd[order(-matrix_alle_icd$Freq_ges,matrix_alle_
        icd$Freq_lk),]

148 ### Tabelle als CSV auf Festplatte speichern
149 write.table(x=matrix_alle_icd,file="/Users/Florian/Documents/Bachelorarbeit/CSV
        /anzahl_icd_pro_lk.csv",sep=";",row.names=FALSE,col.names=TRUE)
```

Quelltext A.17: R-Quelltext 5.8-ICD-Code-Absolut-Heatmap.R

```
1  ### Heatmap - spezielle Krankheitsbilder / ICDs ###

3  # Variable setzen (ICD-Code)
4  icd_code = "I1390"

6  # initial laden und ausführen
7  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

9  # Ambudaten einlesen und bereinigen
10 ambudaten <- ambudaten_laden()

12 # Die Spalten "PSEUDONYM", "ICD" werden extrahiert
13 ambu_icd <- ambudaten[c(1,3)]

15 # Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebiet)
16 shapes <- shape_laden(karte="plz")

18 # Verbinden aller Landkreise mit den Ambudaten
19 all_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))

21 # Häufigkeitstabelle der ICD-Codes, gruppiert nach den PLZ
22 all_icd_freq <- data.frame(table(all_icd$ICD,all_icd$PLZ))
23 # Überschriften setzen
24 colnames(all_icd_freq) <- c("ICD","PLZ","Freq")

26 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
27 all_icd_freq$PLZ <- str_pad(string=all_icd_freq$PLZ,width=5,pad="0")

29 # Data.frame für die Darstellung (maptools) erzeugen
30 plz_shape <- data.frame(shapes$plz)
31 plz_shape <- cbind(plz_shape,0)
32 # Spaltenüberschriften vergeben
33 colnames(plz_shape) <- c("PLZ","Freq")

35 # Filterung des ausgewählten ICD-Codes
```

```
36 all_icd_code <- subset(x=all_icd_freq,subset=(ICD==icd_code))
37 # Spaltenüberschriften vergeben
38 colnames(all_icd_code) <- c("ICD","PLZ","Freq")

40 # Anzahl des ICD-Codes pro PLZ-Gebiet in den data.frame "plz_shape" schreiben
41 for (i in 1:length(plz_shape$PLZ)) {tmp <- subset(x=all_icd_code$Freq,subset=(
       all_icd_code$PLZ == plz_shape[i,1]))
42   if (length(tmp) == 0) {
43     plz_shape[i,2] <- 0
44   }
45   else {
46     plz_shape[i,2] <- tmp
47   }
48 }

50 # Dynamische Einteilung der Schritte für die Heatmap
51 if (max(plz_shape$Freq) < length(farben_shape)) {
52   break_points <- ceiling(seq(from=min(plz_shape$Freq),to=max(plz_shape$Freq),
         by=1))
53 } else {
54   break_points <- ceiling(seq(from=min(plz_shape$Freq),to=max(plz_shape$Freq),
         by=(max(plz_shape$Freq)/length(farben_shape))))
55 }

57 class <- cut(plz_shape$Freq,breaks=break_points)
58 levels(class) <- break_points[2:length(break_points)]



61 ### PLOT BEGINNT ###
62 # Plotten
63 plot(x=shapes,border="darkgrey",col=farben_shape[class],main=paste("Häufigkeit
       des ICD-Codes",icd_code,"in Sachsen-Anhalt",sep=" "))

65 # Gesamtanzahl des ICD-Codes anzeigen
66 mtext(text=paste("Gesamtanzahl des ICD-Codes: ",sum(plz_shape$Freq),sep=""),
       side=1,cex=0.7,line=1)

68 # Legende anzeigen
69 legend(x=par("usr")[2]-(1/3.95*par("usr")[2]),y=par("usr")[4],fill="white",
       legend="= 0",border="black",cex=1.2,title="Legende:",bty="n")
70 legend(x=par("usr")[2]-(1/4*par("usr")[2]),y=par("usr")[4]-(1/200*par("usr")
       [4]),fill=farben_shape,legend=paste("<=",levels(class)),border="black",cex
       =1.2,title=NA,bty="n")
```

Quelltext A.18: R-Quelltext 5.9-ICD-Versicherungsart-Diagramm.R

```
### Barplot der ICD-Codes und Versicherungsart ###

# Variable setzen
  # Anzahl von ICD Codes die dargestellt werden
  # Default: 55
  anzahl_icd <- 55
  # Mindesthäufigkeit pro ICD-Code in Sachsen-Anhalt
  # Default: 100
  mindest_haeufigkeit <- 100

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "ICD" werden extrahiert
ambu_icd <- ambudaten[c(1,3)]

# Verbinden der Stammdaten (alle Landkreise) mit den Ambudaten
vart_icd <- unique(merge(alle_kreise,ambu_icd,by="PSEUDONYM"))

### Beginn: Einteilung der ICD-Codes nach der Versicherungsart
# Aggregation der Versicherungsarten nach den ICD-Codes
agg_vart_icd <- aggregate(vart_icd$VERSICHERUNGSART,by=list(ICD=vart_icd$ICD),
    FUN=table)

# Arbeitslose, Beschäftigte, Familienang. und Rentner extrahieren
matrix_vart_icd_absolut <- cbind(
  agg_vart_icd$x[,1],
  agg_vart_icd$x[,2],
  agg_vart_icd$x[,3],
  agg_vart_icd$x[,8]
)

# Summierung der Zeilen und als Spalte hinzufügen
matrix_vart_icd_absolut <- cbind(matrix_vart_icd_absolut,rowSums(matrix_vart_
    icd_absolut))

# Überschriften setzen
rownames(matrix_vart_icd_absolut) <- agg_vart_icd$ICD
colnames(matrix_vart_icd_absolut) <- c("arbeitslos","beschäftigt","
    Familienangehörige","Rentner","row_Freq")
```

```
42 # Matrix sortieren nach der row-Frequenz
43 matrix_vart_icd_absolut <- matrix_vart_icd_absolut[order(-matrix_vart_icd_
       absolut[,5]),]

45 # Erstellung einer Matrix der Form: 1 Fall auf X Versicherte
46 matrix_vart_icd_pro_patient <- data.frame(length(alle_kreise$PSEUDONYM) /
       matrix_vart_icd_absolut[,1:4])

48 # Überschriften setzen
49 colnames(matrix_vart_icd_pro_patient) <- c("arbeitslos","beschäftigt","
       Familienangehörige","Rentner")

51 # Spalte hinzufügen: absolute Häufigkeit
52 matrix_vart_icd_pro_patient <- cbind(matrix_vart_icd_pro_patient,abs_h=matrix_
       vart_icd_absolut[,5])

54 # Filterung der ICD-Codes mit der oben gesetzten Mindesthäufigkeit
55 matrix_vart_icd_pro_patient <- subset(x=matrix_vart_icd_pro_patient,subset=(
       matrix_vart_icd_pro_patient[,5] >= mindest_haeufigkeit))

57 # Spalte hinzufügen: Standardabweichung
58 matrix_vart_icd_pro_patient <- cbind(matrix_vart_icd_pro_patient,st_abw=apply(X
       =matrix_vart_icd_pro_patient[,1:4],MARGIN=1,FUN=sd2))

60 # Sortierung der Matrix nach der grö ß ten Standardabweichung
61 matrix_vart_icd_pro_patient <- matrix_vart_icd_pro_patient[order(-matrix_vart_
       icd_pro_patient[,6]),]

63 # ICD-Codes mit der grö ß ten Standardabweichnung werden ausgewählt und
       prozentual dargestellt
64 matrix_vart_icd_pro_patient_proz <- NULL
65 for (i in 1:anzahl_icd) {matrix_vart_icd_pro_patient_proz <- rbind(matrix_vart_
       icd_pro_patient_proz,sum(matrix_vart_icd_pro_patient[i,1:4]) / matrix_vart_
       icd_pro_patient[i,1:4])}
66 matrix_vart_icd_pro_patient_proz <- as.matrix(matrix_vart_icd_pro_patient_proz)
67 matrix_vart_icd_pro_patient_proz <- prop.table(x=matrix_vart_icd_pro_patient_
       proz,margin=1)*100
68 top <- t(matrix_vart_icd_pro_patient_proz)

70 ### Beginn: Einteilung der ICD-Codes nach den Landkreisen
71 top_icd <- data.frame("ICD"=colnames(top))

73 # Filterung der gewünschten ICD-Codes aus den Ambudaten
```

```
74 ambu_icd_bereich <- unique(ambu_icd[ambu_icd$ICD %in% top_icd$ICD,])

76 # Verbinden der Stammdaten mit den Ambudaten für jeden LK und eine Hä
       ufigkeitstabelle auf Basis der ICD-Spalte erstellen
77 halle_vart_icd <- data.frame(table("ICD"=subset(merge(halle,ambu_icd_bereich,by
       ="PSEUDONYM"),select=c(ICD))))
78 saalekreis_vart_icd <- data.frame(table("ICD"=subset(merge(saalekreis,ambu_icd_
       bereich,by="PSEUDONYM"),select=c(ICD))))
79 mansfeld_suedharz_vart_icd <- data.frame(table("ICD"=subset(merge(mansfeld_
       suedharz,ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
80 harz_vart_icd <- data.frame(table("ICD"=subset(merge(harz,ambu_icd_bereich,by="
       PSEUDONYM"),select=c(ICD))))
81 anhalt_bitterfeld_vart_icd <- data.frame(table("ICD"=subset(merge(anhalt_
       bitterfeld,ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
82 salzlandkreis_vart_icd <- data.frame(table("ICD"=subset(merge(salzlandkreis,
       ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
83 burgenlandkreis_vart_icd <- data.frame(table("ICD"=subset(merge(burgenlandkreis
       ,ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
84 wittenberg_vart_icd <- data.frame(table("ICD"=subset(merge(wittenberg,ambu_icd_
       bereich,by="PSEUDONYM"),select=c(ICD))))
85 dessau_rosslau_vart_icd <- data.frame(table("ICD"=subset(merge(dessau_rosslau,
       ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
86 stendal_vart_icd <- data.frame(table("ICD"=subset(merge(stendal,ambu_icd_
       bereich,by="PSEUDONYM"),select=c(ICD))))
87 altmarkkreis_salzwedel_vart_icd <- data.frame(table("ICD"=subset(merge(
       altmarkkreis_salzwedel,ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))
88 magdeburg_vart_icd <- data.frame(table("ICD"=subset(merge(magdeburg,ambu_icd_
       bereich,by="PSEUDONYM"),select=c(ICD))))
89 boerde_vart_icd <- data.frame(table("ICD"=subset(merge(boerde,ambu_icd_bereich,
       by="PSEUDONYM"),select=c(ICD))))
90 jerichower_land_vart_icd <- data.frame(table("ICD"=subset(merge(jerichower_land
       ,ambu_icd_bereich,by="PSEUDONYM"),select=c(ICD))))

92 # Die Spalte "pro_patient" jedem LK hinzufügen
93 # "pro_patient" beschreibt: 1 Fall auf X Versicherte
94 halle_vart_icd <- cbind(halle_vart_icd,"pro_patient"=length(halle$PSEUDONYM) /
       halle_vart_icd$Freq)
95 saalekreis_vart_icd <- cbind(saalekreis_vart_icd,"pro_patient"=length(
       saalekreis$PSEUDONYM) / saalekreis_vart_icd$Freq)
96 mansfeld_suedharz_vart_icd <- cbind(mansfeld_suedharz_vart_icd,"pro_patient"=
       length(mansfeld_suedharz$PSEUDONYM) / mansfeld_suedharz_vart_icd$Freq)
97 harz_vart_icd <- cbind(harz_vart_icd,"pro_patient"=length(harz$PSEUDONYM) /
       harz_vart_icd$Freq)
98 anhalt_bitterfeld_vart_icd <- cbind(anhalt_bitterfeld_vart_icd,"pro_patient"=
```

```
    length(anhalt_bitterfeld$PSEUDONYM) / anhalt_bitterfeld_vart_icd$Freq)
salzlandkreis_vart_icd <- cbind(salzlandkreis_vart_icd,"pro_patient"=length(
    salzlandkreis$PSEUDONYM) / salzlandkreis_vart_icd$Freq)
burgenlandkreis_vart_icd <- cbind(burgenlandkreis_vart_icd,"pro_patient"=length
    (burgenlandkreis$PSEUDONYM) / burgenlandkreis_vart_icd$Freq)
wittenberg_vart_icd <- cbind(wittenberg_vart_icd,"pro_patient"=length(
    wittenberg$PSEUDONYM) / wittenberg_vart_icd$Freq)
dessau_rosslau_vart_icd <- cbind(dessau_rosslau_vart_icd,"pro_patient"=length(
    dessau_rosslau$PSEUDONYM) / dessau_rosslau_vart_icd$Freq)
stendal_vart_icd <- cbind(stendal_vart_icd,"pro_patient"=length(stendal$
    PSEUDONYM) / stendal_vart_icd$Freq)
altmarkkreis_salzwedel_vart_icd <- cbind(altmarkkreis_salzwedel_vart_icd,"pro_
    patient"=length(altmarkkreis_salzwedel$PSEUDONYM) / altmarkkreis_salzwedel_
    vart_icd$Freq)
magdeburg_vart_icd <- cbind(magdeburg_vart_icd,"pro_patient"=length(magdeburg$
    PSEUDONYM) / magdeburg_vart_icd$Freq)
boerde_vart_icd <- cbind(boerde_vart_icd,"pro_patient"=length(boerde$PSEUDONYM)
     / boerde_vart_icd$Freq)
jerichower_land_vart_icd <- cbind(jerichower_land_vart_icd,"pro_patient"=length
    (jerichower_land$PSEUDONYM) / jerichower_land_vart_icd$Freq)

# Ergebnismatrix erzeugen (1 Fall auf X Versicherte pro LK)
# Mit Nullen vorbelegen damit alle eine identische Länge besitzen
matrix_vart_icd_landkreise <- matrix(data=0,nrow=length(top_icd$ICD),ncol=
    length(landkreise))
dimnames(matrix_vart_icd_landkreise) <- list(colnames(top),landkreise)

# Liste erstellen mit allen ICD der Landkreise
lk <- list (magdeburg_vart_icd[c(1,3)],
      halle_vart_icd[c(1,3)],
      dessau_rosslau_vart_icd[c(1,3)],
      altmarkkreis_salzwedel_vart_icd[c(1,3)],
      stendal_vart_icd[c(1,3)],
      boerde_vart_icd[c(1,3)],
      jerichower_land_vart_icd[c(1,3)],
      harz_vart_icd[c(1,3)],
      salzlandkreis_vart_icd[c(1,3)],
      anhalt_bitterfeld_vart_icd[c(1,3)],
      wittenberg_vart_icd[c(1,3)],
      mansfeld_suedharz_vart_icd[c(1,3)],
      saalekreis_vart_icd[c(1,3)],
      burgenlandkreis_vart_icd[c(1,3)]
)
```

```
131 # Ergebnismatrix mit den Werten belegen
132 for (i in 1:length(matrix_vart_icd_landkreise[1,])) {
133   for (y in 1:length(matrix_vart_icd_landkreise[,1])) {
134     tmp <- subset(lk[[i]][2], lk[[i]]$ICD == rownames(matrix_vart_icd_
          landkreise)[y])
135     matrix_vart_icd_landkreise[y,i] <- tmp[1,]
136   }
137 }
138 matrix_vart_icd_landkreise[is.na(matrix_vart_icd_landkreise)] <- 0

140 # Ergebnisse prozentual darstellen
141 top_landkreise <- NULL
142 for (i in 1:length(matrix_vart_icd_landkreise[,1])) {top_landkreise <- rbind(
      top_landkreise,sum(matrix_vart_icd_landkreise[i,1:14]) / matrix_vart_icd_
      landkreise[i,1:14])}
143 rownames(top_landkreise) <- rownames(matrix_vart_icd_landkreise)
144 # "top_landkreise" transponieren um ICDs als Spalten zu bekommen
145 top_landkreise <- t(top_landkreise)
146 # Inf (Unendlich) mit Nullen ersetzen
147 top_landkreise[which(top_landkreise==Inf)] <- 0
148 # Prozentuale Tabelle erstellen
149 top_landkreise_proz <- prop.table(x=top_landkreise,margin=2)*100

151 ### PLOT BEGINNT ###
152 # Fenster in zwei gleichgro ß e Teile einteilen
153 layout(mat=matrix(c(1,1,2,2),nrow=2,ncol=2,byrow=TRUE))

155 # Seitenabstand oberes Diagramm
156 par(mar=c(2,3,5,7))

158 # Y-Achse Markierung
159 ybereich <- pretty(x=c(0,100),n=20)

161 # Plotten oben
162 top_st_abw_plot <- barplot(height=top,main="ICD-Codes nach den
      Versicherungsarten",col=farben_2,axes=FALSE,axisnames=FALSE)

164 # 1. Legende anzeigen
165 legend(x=(max(top_st_abw_plot)+1),y=100,legend=rownames(top),fill=farben_2,cex
      =0.7,xpd=TRUE,border=NA)

167 # 1. Y-Achse anzeigen
168 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
      =-0.5)
```

```
170 # 1. Y-Achsen-Beschriftung
171 mtext(text="Prozent",side=2,line=1.7,cex=0.8)

173 # Absolute Häufigkeit anzeigen
174 text(x=top_st_abw_plot+0.3,y=104,adj=1,xpd=TRUE,labels=matrix_vart_icd_pro_
        patient[1:anzahl_icd,5],cex=0.65,col=farben_2[2])
175 text(x=-0.5,y=104,adj=1,labels="absolute Häufigkeit",xpd=TRUE,cex=0.55,col=
        farben_2[2])

177 # X-Achse anzeigen
178 text(x=top_st_abw_plot+0.1,y=-2,srt=90,adj=1,xpd=TRUE,labels=colnames(top),cex
        =0.6)

180 # Seitenabstand unteres Diagramm
181 par(mar=c(2,3,0,7))

183 # Plotten unten
184 top_st_abw_plot_landkreise <- barplot(height=top_landkreise_proz,col=farben_2,
        axes=FALSE,axisnames=FALSE)

186 # 2. Legende anzeigen
187 legend(x=(max(top_st_abw_plot)+1),y=100,legend=rownames(top_landkreise),fill=
        farben_2,cex=0.7,xpd=TRUE,border=NA)

189 # 2. Y-Achse anzeigen
190 axis(side=2,at=ybereich,labels=paste(ybereich,"%"),las=2,cex.axis=0.85,pos
        =-0.5)

192 # 2. Y-Achsen-Beschriftung
193 mtext(text="Prozent",side=2,line=1.7,cex=0.8)

195 # X-Achsen-Beschriftung
196 mtext(text="ICD Codes",side=1,line=0.5,cex=0.8)
```

Quelltext A.19: R-Quelltext 6.1-ATC-Verteilung-Diagramm.R

```
1 ### Barplot aller ATC-Codes (absolute Häufigkeit) ###

3 # Variablen setzen (rote Markierungen)
4 # default: 50
5 mark_1 <- 50
6 # default: 450
7 mark_2 <- 450
```

```
9  # initial laden und ausführen
10 source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
11
12 # Arzneidaten einlesen
13 arznei <- arzneimittel_laden()
14
15 # Absolute Häufigkeitstabelle der ATC-Codes erstellen
16 arznei_absh <- data.frame(table("ATC"=arznei$ATC))
17
18 # Häufigkeitstabelle absteigend sortieren
19 arznei_absh <- data.frame(arznei_absh[order(-arznei_absh$Freq),])
20
21 ### PLOT BEGINNT ###
22 # Seitenabstand
23 par(mar=c(6,6,4,2))
24
25 # Plotten
26 barplot(height=arznei_absh[,2],yaxt="n",main="Häufigkeitsverteilung der ATC-
       Codes",xlab="ATC Codes",ylab="Anzahl",ylim=c(0,1.1*max(arznei_absh[,2])),
       space=0,col=farben[1],border=farben[1])
27
28 # Gesamtanzahl ATC-Codes anzeigen
29 mtext(text=paste("Gesamtanzahl an ATC-Codes: ",format(x=length(arznei$ATC),big.
       mark=".",scientific=FALSE),sep=""),side=3,cex=0.7,line=-1)
30
31 # Y-Achse in 50.000 Schritten definieren
32 ybereich <- pretty(x=c(0,max(arznei_absh[,2])),n=10)
33
34 # Y-Achse anzeigen
35 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=FALSE)
       ,las=2,cex.axis=0.8,pos=-17)
36
37 # X-Achse definieren
38 xbereich <- c(0,mark_1,mark_2,length(arznei_absh[,1]))
39
40 # X-Achse anzeigen
41 axis(side=1,at=xbereich,cex.axis=0.8,pos=-10000)
42
43 # Aufsummierte relative Häufigkeit berechnen
44 rel <- arznei_absh$Freq * 100 / sum(arznei_absh$Freq)
45 sum_mark_1 <- sum(rel[1:mark_1])
46 sum_mark_2 <- sum(rel[1:mark_2])
```

```
48 # Rote Striche anzeigen
49 lines(x=mark_1,y=80000,col=farben_2[2],type="h")
50 lines(x=mark_2,y=100000,col=farben_2[2],type="h")
51 text(x=mark_1+35,y=88000,labels=paste(round(x=sum_mark_1,digits=2),"%"),cex
       =0.75,xpd=TRUE,col=farben_2[2])
52 text(x=mark_2+35,y=108000,labels=paste(round(x=sum_mark_2,digits=2),"%"),cex
       =0.75,xpd=TRUE,col=farben_2[2])

54 ### Tabelle als CSV auf Festplatte speichern (abs. Häufigkeit der ATC-Codes)
55 write.table(x=arznei_absh,file="/Users/Florian/Documents/Bachelorarbeit/CSV/ATC
       -Codes_absolute_haeufigkeit.csv",sep=";",row.names=FALSE,col.names=TRUE,
       fileEncoding="latin1")
```

Quelltext A.20: R-Quelltext 6.2-ATC-Top30-Verteilung-Diagramm.R

```
1  ### Barplot der 50 häufigsten ATC-Codes (absolute Häufigkeit) ###

3  # Variable setzen (Anzahl der häufigsten ATC-Codes)
4  # default: 30
5  anzahl_atc_codes <- 30

7  # initial laden und ausführen
8  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

10 # Arzneidaten einlesen
11 arznei <- arzneimittel_laden()

13 # Absolute Häufigkeitstabelle der ATC-Codes erstellen
14 arznei_absh <- data.frame(table("ATC"=arznei$ATC))

16 # Häufigkeitstabelle absteigend sortieren
17 arznei_absh <- data.frame(arznei_absh[order(-arznei_absh$Freq),])

19 # Die häufigsten ATC-Codes extrahieren
20 atc_bereich <- arznei_absh[1:anzahl_atc_codes,]

22 ### PLOT BEGINNT ###
23 # Seitenabstand
24 par(mar=c(6,7,3,2))

26 # Plotten
27 plot <- barplot(height=atc_bereich[,2],main=paste("Die häufigsten ",anzahl_atc_
       codes," ATC-Codes",sep=""),axes=FALSE,axisnames=FALSE,col=farben[4],ylim=c
       (0,1.1*max(atc_bereich[,2])))
```

```
29 # Legende erzeugen
30   # Wirkstoffe
31   wirkstoff <- c("Metoprolol","Ramipril","Pantoprazol","Ibuprofen","Simvastatin
         ","Torasemid","Levothyroxin","Amlodipin","Metamizol","Bisoprolol","
         Metformin","Diclofenac","Omeprazol","Allopurinol","Acetylsalicylsäure")
32   tabelle_atc <- data.frame("ATC_Code"=atc_bereich[1:15,1],"Wirkstoff"=
         wirkstoff)

34   # Relative Häufigkeit berechnen
35   rel <- atc_bereich$Freq[1:15] * 100 / length(arznei$ATC)
36   rel <- round(x=rel,digits=2)
37   rel_legende <- paste(atc_bereich[1:15,1],"  (",format(x=rel,nsmall=2),"%",")
          ",sep="",wirkstoff)
38 legend('topright',legend=rel_legende,title="ATC-Code | rel. Häufigkeit |
     Wirkstoff",cex=0.7)

40 # Anzahl jeder Säule anzeigen
41 text(x=plot,y=(atc_bereich[,2]+18000),srt=90,labels=format(x=atc_bereich[,2],
     big.mark=".",scientific=FALSE),cex=0.7)

43 # ATC Codes jeder Säule anzeigen
44 text(x=plot,y=-22000,labels=atc_bereich[,1],cex=0.6,srt=90,xpd=TRUE)

46 # Y-Achse definieren
47 ybereich <- pretty(x=c(0,max(arznei_absh[,2])),n=10)

49 # Y-Achse anzeigen
50 axis(side=2,at=ybereich,labels=format(x=ybereich,big.mark=".",scientific=FALSE)
     ,las=2,cex.axis=0.9,pos=-0.6)

52 # Y-Achsen-Beschriftung
53 mtext(text="Anzahl",side=2,line=4,cex=0.9)

55 # X-Achsen-Beschriftung
56 mtext(text="ATC Codes",side=1,line=4,cex=0.9)
```

Quelltext A.21: R-Quelltext 6.3-ATC-Verteilung-Absolut-Heatmap.R

```
1 ### Heatmap - Verteilung der ATC Codes ###

3 # initial laden und ausführen
4 source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")
```

```
6  # Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
7  shapes <- shape_laden(karte="plz")
8
9  # Arzneidaten einlesen
10 arznei <- arzneimittel_laden()
11
12 # Die Spalten "PSEUDONYM", "ATC" extrahieren
13 arznei_atc <- arznei[c(1,4)]
14
15 # Verbinden aller Landkreise mit den ATC-Codes der arznei_daten
16 all_atc <- merge(alle_kreise,arznei_atc,by="PSEUDONYM")
17
18 # Absolute Häufigkeitstabelle erzeugen (nach den PLZ)
19 all_atc_freq <- data.frame(table(all_atc$PLZ))
20 colnames(all_atc_freq) <- c("PLZ","Freq")
21
22 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
23 all_atc_freq$PLZ <- str_pad(string=all_atc_freq$PLZ,width=5,pad="0")
24
25 # Data.frame für die Darstellung (maptools) erzeugen
26 anzahl_atc_lk <- data.frame(shapes$plz)
27 anzahl_atc_lk <- cbind(anzahl_atc_lk,0)
28 # Spaltenüberschriften vergeben
29 colnames(anzahl_atc_lk) <- c("PLZ","Freq")
30
31 # Anzahl der vorkommenden ATC-Codes in den data.frame "anzahl_atc_lk" schreiben
32 for (i in 1:length(anzahl_atc_lk$PLZ)) {tmp <- subset(x=all_atc_freq$Freq,
       subset=(all_atc_freq$PLZ == anzahl_atc_lk[i,1]))
33   if (length(tmp) == 0) {
34     anzahl_atc_lk[i,2] <- 0
35   }
36   else {
37     anzahl_atc_lk[i,2] <- tmp
38   }
39 }
40
41 # Einteilung der Schritte für die Heatmap
42 break_points <- seq(from=0,to=168000,by=12000)
43 class <- cut(anzahl_atc_lk$Freq,breaks=break_points)
44 levels(class) <- break_points[2:length(break_points)]
45
46 ### PLOT BEGINNT ###
47 # Plotten
48 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="Absolute
```

```
    Verteilung aller ATC Codes")

# Legende anzeigen
legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
    black",cex=1.2,title="Legende:",bty="n")

# Gesamtanzahl an ATC-Codes anzeigen
mtext(text=paste("Gesamtanzahl ATC-Codes: ",sum(all_atc_freq$Freq),sep=""),side
    =1,cex=0.7,line=1)
```

Quelltext A.22: R-Quelltext 6.4-ATC-Verteilung-ProVersicherten-Heatmap.R

```
### Heatmap - Verteilung der ATC Codes ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")

# Arzneidaten einlesen
arznei <- arzneimittel_laden()

# Die Spalten "PSEUDONYM", "ATC" extrahieren
arznei_atc <- arznei[c(1,4)]

# Verbinden aller Landkreise mit den ATC-Codes der arznei_daten
all_atc <- merge(alle_kreise,arznei_atc,by="PSEUDONYM")

# Absolute Häufigkeitstabelle ATC-Codes erzeugen (nach den PLZ)
all_atc_freq <- data.frame(table(all_atc$PLZ))
colnames(all_atc_freq) <- c("PLZ","Freq")

# Alle PLZ 5-stellig machen (mit Nullen auffüllen)
all_atc_freq$PLZ <- str_pad(string=all_atc_freq$PLZ,width=5,pad="0")

### Anzahl ATC-Codes pro PLZ ermitteln
# Data.frame für die Darstellung (maptools) erzeugen
anzahl_atc_lk <- data.frame(shapes$plz)
anzahl_atc_lk <- cbind(anzahl_atc_lk,0)
# Spaltenüberschriften vergeben
colnames(anzahl_atc_lk) <- c("PLZ","Freq")

# Anzahl der vorkommenden ATC-Codes in den data.frame "anzahl_atc_lk" schreiben
```

```
33 for (i in 1:length(anzahl_atc_lk$PLZ)) {tmp <- subset(x=all_atc_freq$Freq,
       subset=(all_atc_freq$PLZ == anzahl_atc_lk[i,1]))
34   if (length(tmp) == 0) {
35     anzahl_atc_lk[i,2] <- 0
36   }
37   else {
38     anzahl_atc_lk[i,2] <- tmp
39   }
40 }

42 ### Anzahl Einwohner pro PLZ ermitteln
43 # Data.frame für die Darstellung (maptools) erzeugen
44 anzahl_ver_lk <- data.frame(shapes$plz)
45 anzahl_ver_lk <- cbind(anzahl_ver_lk,0)
46 colnames(anzahl_ver_lk) <- c("PLZ","Freq")

48 # Absolute Häufigkeitstabelle der Versicherten erzeugen (nach den PLZ)
49 all_ver_freq <- data.frame(table(alle_kreise$PLZ))
50 colnames(all_ver_freq) <- c("PLZ","Freq")
51 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
52 all_ver_freq$PLZ <- str_pad(string=all_ver_freq$PLZ,width=5,pad="0")

54 # Anzahl der Versicherten in den data.frame "anzahl_ver_lk" schreiben
55 for (i in 1:length(anzahl_ver_lk$PLZ)) {tmp <- subset(x=all_ver_freq$Freq,
       subset=(all_ver_freq$PLZ == anzahl_ver_lk[i,1]))
56   if (length(tmp) == 0) {
57     anzahl_ver_lk[i,2] <- 0
58   }
59   else {
60     anzahl_ver_lk[i,2] <- tmp
61   }
62 }

64 # Anzahl ATC-Codes durch die Anzahl der Versicherten dividieren (1 ATC-Code pro
       Versicherten)
65 plot_pro_patient <- data.frame("PLZ"=anzahl_atc_lk$PLZ,"pro_patient"=0)
66 for (i in 1:length(anzahl_ver_lk$PLZ)) {plot_pro_patient$pro_patient[i] <-
       anzahl_atc_lk$Freq[i] / anzahl_ver_lk$Freq[i]}

68 # Einteilung der Schritte für die Heatmap
69 break_points <- c(0,12:20)
70 class <- cut(plot_pro_patient$pro_patient,breaks=break_points)
71 levels(class) <- break_points[2:length(break_points)]
```

```
73 # Nur 9 statt der 14 Farben nötig
74 farben_shape_9 <- farben_shape[6:14]

76 ### PLOT BEGINNT ###
77 # Plotten
78 plot(x=shapes,border="darkgrey",col=farben_shape_9[class],main="ATC Codes pro
       Versicherten")

80 # Legende anzeigen
81 legend('topright',fill=farben_shape_9,legend=paste("<= ",levels(class),"ATC
       Codes pro Versicherten"),border="black",cex=0.8,title="Legende:",bty="n")
```

Quelltext A.23: R-Quelltext 6.5-ATC-Priscus-Heatmap.R

```
1  ### Heatmap und Tabelle - PRISCUS-Liste ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Priscus-Liste einlesen
7  priscus_csv <- priscus_laden()

9  # Ambudaten einlesen und bereinigen
10 ambudaten <- ambudaten_laden()

12 # Arzneidaten einlesen
13 arznei <- arzneimittel_laden()

15 # Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
16 shapes <- shape_laden("plz")

18 # Data.frame aus den ATC-Codes erstellen
19 priscus_liste <- data.frame("ATC"=priscus_csv$ATC)

21 # Die Spalten "PSEUDONYM", "ALTER" werden extrahiert
22 ambu_age <- ambudaten[c(1,6)]

24 # Die Spalten "PSEUDONYM", "ATC" werden extrahiert
25 arznei_atc <- arznei[c(1,4)]

27 # In den Arzneidaten ATCs raussuchen, die in der Priscus_liste vorkommen
28 # Unique() anwenden, da ein Versicherter pro Jahr ein Medikament mehrmals
       verschrieben bekommen könnte
29 priscus_atc <- unique(arznei_atc[arznei_atc$ATC %in% priscus_liste$ATC,])
```

```
31 # In den Ambudaten die Versicherten raussuchen, die > 65 Jahre
32 # Unique() anwenden, da ein Versicherter in den Ambudaten mehrmals vorkommt
33 priscus_age <- unique(subset(x=ambu_age,subset=(ambu_age$AGE > 65)))

35 # priscus_atc  und  priscus_age  verbinden
36 age_atc <- merge(priscus_atc,priscus_age,by="PSEUDONYM")


39 ### HEATMAP BERECHNUNG BEGINNT ###
40 alle_kreise_priscus <- unique(merge(alle_kreise,age_atc,by="PSEUDONYM"))

42 # Data.frame für die Darstellung (maptools) erzeugen
43 plz_priscus <- data.frame(shapes$plz)
44 plz_priscus <- cbind(plz_priscus,0)
45 # Spaltenüberschriften vergeben
46 colnames(plz_priscus) <- c("PLZ","Freq")

48 # Häufigkeitstabelle über alle vorkommenden PLZ der Landkreise
49 plz_lk <- data.frame(table(alle_kreise_priscus$PLZ))
50 # Spaltenüberschriften vergeben
51 colnames(plz_lk) <- c("PLZ","Freq")
52 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
53 plz_lk$PLZ <- str_pad(string=plz_lk$PLZ,width=5,pad="0")

55 # Anzahl der vorkommenden PLZ der Landkreise in den data.frame "plz_priscus"
       schreiben
56 for (i in 1:length(plz_priscus$PLZ)) {tmp <- subset(x=plz_lk$Freq,subset=(plz_
       lk$PLZ == plz_priscus[i,1]))
57   if (length(tmp) == 0) {
58     plz_priscus[i,2] <- 0
59   }
60   else {
61     plz_priscus[i,2] <- tmp
62   }
63 }

65 # Einteilung der Schritte für die Heatmap
66 break_points <- c
       (0,110,220,330,440,550,660,770,880,990,1100,1210,1320,1430,1550)
67 class <- cut(plz_priscus$Freq,breaks=break_points)
68 levels(class) <- break_points[2:length(break_points)]
```

```
71 ### PLOT BEGINNT ###
72 # Plotten
73 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="PRISCUS")

75 # Gesamtanzahl an ATC-Codes anzeigen
76 mtext(text=paste("Gesamtanzahl ATC-Codes: ",format(x=length(alle_kreise_priscus
       $ATC),big.mark=".",scientific=FALSE),sep=""),side=1,cex=0.7,line=1)

78 # Legende anzeigen
79 legend('topright',fill=farben_shape,legend=paste("<=",levels(class)),border="
       black",cex=0.8,title="Legende:",bty="n")

81 ### Tabelle als CSV auf Festplatte speichern (absolute Häufigkeit der ATC-Codes
        der PRISCUS-Liste)
82 # Tabelle beinhaltet gesamten Datenbestand (alle_kreise und alle_anderen)
83 tabelle_priscus <- data.frame(table("ATC"=age_atc$ATC))
84 tabelle_priscus <- rbind(tabelle_priscus,c(NA,sum(tabelle_priscus$Freq)))
85 write.table(x=tabelle_priscus,file="/Users/Florian/Documents/Bachelorarbeit/CSV
       /priscus_atc_haeufigkeit.csv",sep=";",row.names=FALSE,col.names=TRUE,
       fileEncoding="latin1")
```

Quelltext A.24: R-Quelltext 7.1-PZN-Packungspreise-Diagramm.R

```
1  ### Barplot: Packungspreise pro PZN ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Arzneidaten einlesen
7  arznei <- arzneimittel_laden()

9  # PZN_Preise einlesen
10 pzn_preise <- pznkosten_laden()
11 colnames(pzn_preise) <- c("PZN","Preis")

13 # Absolute Häufigkeitstabelle der PZN-Codes erstellen
14 pzn_absh <- data.frame(table("PZN"=arznei$PZN))

16 # Verbinden der PZN-Codes mit den Preisen
17 pzn_absh_preise <- merge(pzn_absh,pzn_preise,by="PZN")

19 # PZN-Codes absteigend nach Häufigkeit sortieren
20 pzn_absh_preise <- pzn_absh_preise[order(-pzn_absh_preise$Freq),]
```

```
23 ## PLOT BEGINNT ###
24 # Seitenabstand
25 par(mar=c(3,7,5,7))

27 # Y-Achsen Markierungen
28 ybereich_1 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Freq)),n=20)
29 ybereich_2 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Preis)),n=20)

31 # Plotten
32 barplot(height=pzn_absh_preise$Freq,main="Packungspreise pro PZN-Code (nach Hä
       ufigkeit der PZN-Codes)",ylim=c(0,max(ybereich_1)),axes=FALSE,col=farben_
       2[12],border=farben_2[12],space=0)

34 # Y-Achse_1 anzeigen
35 axis(side=2,at=ybereich_1,labels=format(x=ybereich_1,big.mark=".",scientific=
       FALSE),col=farben_2[12],col.axis=farben_2[12],las=2,cex.axis=0.75)

37 # Y-Achse_1 Überschrift
38 mtext(text="Anzahl",side=2,line=4,col=farben_2[12],cex=0.7)

40 # Beide Y-Achsen gleich hoch
41 par(new=TRUE)

43 # Plotten
44 barplot(height=pzn_absh_preise$Preis,ylim=c(0,max(ybereich_2)),axes=FALSE,col=
       farben_2[2],border=farben_2[2],space=0)

46 # Y-Achse_2 anzeigen
47 axis(side=4,at=ybereich_2,labels=paste(format(x=ybereich_2,big.mark=".",
       scientific=FALSE),"EUR"),col=farben_2[2],col.axis=farben_2[2],las=2,cex.
       axis=0.75)

49 # Y-Achse_2 Überschrift
50 mtext(text="Preis",side=4,line=5,col=farben_2[2],cex=0.8)

52 # Legende anzeigen
53 legend(x=2000,y=29000,legend=c("Häufigkeit der PZN","Preis pro Packung (PZN)"),
       fill=c(farben_2[12],farben_2[2]),cex=0.7,xpd=TRUE,bty="n",border=NA,horiz=
       TRUE)

55 # X-Achsen Überschrift
56 mtext(text="PZN-Codes",side=1,line=1,col="black",cex=0.7)
```

Quelltext A.25: R-Quelltext 7.2-PZN-Gesamtkosten-Diagramm.R

```
### Barplot: Gesamtkosten pro PZN ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Arzneidaten einlesen
arznei <- arzneimittel_laden()

# PZN_Preise einlesen
pzn_preise <- pznkosten_laden()
colnames(pzn_preise) <- c("PZN","Preis")

# Absolute Häufigkeitstabelle der PZN-Codes erstellen
pzn_absh <- data.frame(table("PZN"=arznei$PZN))

# Verbinden der PZN-Codes mit den Preisen
pzn_absh_preise <- merge(pzn_absh,pzn_preise,by="PZN")

# Absolute Kosten pro PZN als Spalte hinzufügen
pzn_absh_preise <- cbind(pzn_absh_preise,"Kosten_pro_PZN"=pzn_absh_preise$Freq
    * pzn_absh_preise$Preis)

# PZN-Codes absteigend nach Häufigkeit sortieren
pzn_absh_preise <- pzn_absh_preise[order(-pzn_absh_preise$Freq),]

### PLOT BEGINNT ###
# Seitenabstand
par(mar=c(3,7,5,8.5))

# Y-Achsen Markierungen
ybereich_1 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Freq)),n=20)
ybereich_2 <- pretty(x=c(0,1.1*max(pzn_absh_preise$Kosten_pro_PZN)),n=20)

# Plotten
plot1 <- barplot(height=pzn_absh_preise$Kosten_pro_PZN,main="Gesamtkosten pro
    PZN-Code (nach Häufigkeit der PZN-Codes)",ylim=c(0,max(ybereich_2)),axes=
    FALSE,col=farben_2[2],border=farben_2[2],space=0)

# Y-Achse_2 anzeigen
axis(side=4,at=ybereich_2,labels=paste(format(x=ybereich_2,big.mark=".",
    scientific=FALSE),"EUR"),col=farben_2[2],col.axis=farben_2[2],las=2,cex.
    axis=0.75)
```

```
39 # Y-Achse_2 Überschrift
40 mtext(text="Preis",side=4,line=6.5,col=farben_2[2],cex=0.8)

42 # Beide Y-Achsen gleich hoch
43 par(new=TRUE)

45 # Plotten
46 barplot(height=pzn_absh_preise$Freq,ylim=c(0,max(ybereich_1)),axes=FALSE,col=
       farben_2[12],border=farben_2[12],space=0)

48 # Y-Achse_1 anzeigen
49 axis(side=2,at=ybereich_1,labels=format(x=ybereich_1,big.mark=".",scientific=
       FALSE),col=farben_2[12],col.axis=farben_2[12],las=2,cex.axis=0.75)

51 # Y-Achse_1 Überschrift
52 mtext(text="Anzahl",side=2,line=4,col=farben_2[12],cex=0.7)

54 # Legende anzeigen
55 legend(x=max(plot1)-(1/2.5*(max(plot1))),y=165000,legend=c("Häufigkeit der PZN"
       , "Kosten pro PZN"),fill=c(farben_2[12],farben_2[2]),cex=0.7,xpd=TRUE,bty="
       n",border=NA,horiz=TRUE)

57 # X-Achsen Überschrift
58 mtext(text="PZN-Codes",side=1,line=1,col="black",cex=0.7)

60 ### Tabelle als CSV auf Festplatte speichern
61 write.table(x=pzn_absh_preise,file="/Users/Florian/Documents/Bachelorarbeit/CSV
       /kosten_pzn.csv",sep=";",row.names=FALSE,col.names=TRUE)
```

Quelltext A.26: R-Quelltext 7.3-PZN-Durchschnittskosten-Heatmap.R

```
1  ### Heatmap -  Durchschnittskosten für Arzneimittel pro Versicherten (Sachsen-
       Anhalt) ###

3  # initial laden und ausführen
4  source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

6  # Arzneidaten einlesen
7  arznei <- arzneimittel_laden()

9  # Die Spalten "PSEUDONYM" und "PZN" werden extrahiert
10 arznei_pzn <- arznei[c(1,5)]

12 # Ambudaten einlesen und bereinigen
```

```
13 ambudaten <- ambudaten_laden()

15 # Die Spalten "PSEUDONYM", "AGE" und "GENDER" werden extrahiert
16 ambu_age_gender <- unique(ambudaten[c(1,6,7)])

18 # PZN_Preise einlesen
19 pzn_preise <- pznkosten_laden()
20 colnames(pzn_preise) <- c("PZN","Preis")

22 # Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
23 shapes <- shape_laden(karte="plz")

25 # Verbinden der PZN-Codes und deren Preise mit den Pseudonymen
26 arznei_pzn_preise <- merge(arznei_pzn,pzn_preise,by="PZN")

28 # Aggregation der Preise nach den Pseudonymen
29 kosten_pro_ver <- aggregate(arznei_pzn_preise[c(3)],by=list(arznei_pzn_preise$
       PSEUDONYM),FUN=sum)
30 colnames(kosten_pro_ver) <- c("PSEUDONYM","Kosten_pro_V")

32 # Verbinden aller Pseudonyme in SA (alle_kreise) mit dessen Geschlecht und
       Alter
33 ambu_alle_kreise <- merge(alle_kreise[c(1,2)],ambu_age_gender,by="PSEUDONYM")

35 # Verbinden der Ambu-Daten mit den Kosten-Pro-Versicherten
36 # INFO: Es fallen einige Pseudonyme weg, da in Ambudaten nicht vorhanden
37 sa_kosten_pro_ver <- merge(ambu_alle_kreise,kosten_pro_ver,by="PSEUDONYM")

39 # Gesamtkosten pro PLZ-Gebiet ermitteln
40 kosten_plz <- aggregate(sa_kosten_pro_ver[c(5)],by=list(sa_kosten_pro_ver$PLZ),
       FUN=sum)
41 colnames(kosten_plz) <- c("PLZ","Kosten_pro_PLZ")

43 # Anzahl Versicherte pro PLZ-Gebiet ermitteln, die Leistungen in Anspruch
       genommen haben
44 anzahl_plz <- sa_kosten_pro_ver[c(1,2)]
45 anzahl_plz <- data.frame(table(anzahl_plz$PLZ))
46 # Gesamtanzahl Versicherte pro PLZ-Gebiet ermitteln
47 anzahl_plz <- cbind(anzahl_plz,data.frame(table(alle_kreise$PLZ))[,2])
48 colnames(anzahl_plz) <- c("PLZ","Anzahl_Versicherte_Leistungen","Anzahl_
       Versicherte_Ges")

50 # Gesamtkosten und Anzahl Versicherte pro PLZ-Gebiet verbinden
51 sa_gesamt <- merge(kosten_plz,anzahl_plz,by="PLZ")
```

```
53 # Spalte hinzufügen: Durchschnittskosten pro Versicherten
54 sa_gesamt <- cbind(sa_gesamt,"Durchschnittskosten_pro_V"=sa_gesamt$Kosten_pro_
       PLZ / sa_gesamt$Anzahl_Versicherte_Ges)

56 # Data.frame für die Darstellung (maptools) erzeugen
57 shapes_plz <- data.frame(shapes$plz)
58 shapes_plz <- cbind(shapes_plz,0)
59 # Spaltenüberschriften vergeben
60 colnames(shapes_plz) <- c("PLZ","Durchschnittskosten_pro_V")

62 # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
63 sa_gesamt$PLZ <- str_pad(string=sa_gesamt$PLZ,width=5,pad="0")

65 # Durchschnittskosten pro Versicherten in den data.frame "shapes_plz" schreiben
66 for (i in 1:length(shapes_plz$PLZ)) {tmp <- subset(x=sa_gesamt$
       Durchschnittskosten_pro_V,subset=(sa_gesamt$PLZ == shapes_plz[i,1]))
67   if (length(tmp) == 0) {
68     shapes_plz[i,2] <- 0
69   }
70   else {
71     shapes_plz[i,2] <- tmp
72   }
73 }

75 # Einteilung der Schritte für die Heatmap
76 break_points <- c(0,550,585,620,655,690,725,760,795,830,865,900,935,970,1000)
77 class <- cut(shapes_plz$Durchschnittskosten_pro_V,breaks=break_points)
78 levels(class) <- break_points[2:length(break_points)]

80 ### PLOT BEGINNT ###
81 # Plotten
82 plot(x=shapes,border="darkgrey",col=farben_shape[class],main="Durchschnittliche
        Kosten für Arzneimittel pro Versicherten")

84 # Informationen unterhalb der Heatmap anzeigen
85 mtext(text=paste("Min. Durchschnitt: ",round(min(shapes_plz$Durchschnittskosten
       _pro_V),digits=0)," EUR","        ",
86         "Max. Durchschnitt: ",round(max(shapes_plz$Durchschnittskosten_pro_V),
               digits=0)," EUR","        ",
87         "Ges. Durchschnitt: ",round(sum(sa_gesamt$Kosten_pro_PLZ) / sum(sa_
               gesamt$Anzahl_Versicherte_Ges),digits=0)," EUR",
88         "\nGes. Kosten: ",round(sum(sa_gesamt$Kosten_pro_PLZ) / 1000000,digits
               =3)," Mio. EUR ","        ",
```

```
89          "\n\nAnzahl Versicherte mit Leistungen: ",sum(sa_gesamt$Anzahl_
                Versicherte_Leistungen),
90          "\nGes. Anzahl Versicherte: ",sum(sa_gesamt$Anzahl_Versicherte_Ges),
91          sep=""),side=1,cex=0.7,line=+2.5)

93  # Altersgruppe und Geschlecht als Text hinzufügen
94  mtext(text="Alle Altersgruppen  -  Geschlecht: Weiblich & Männlich",side=3,cex
        =0.7)

96  # Legende anzeigen
97  legend('topright',fill=farben_shape,legend=paste("<=",levels(class),"EUR"),cex
        =0.85,title="Legende:",xpd=TRUE,bty="n")

99  ### Tabelle als CSV auf Festplatte speichern
100 ## Liste mit den teuersten Versicherten
101 # Anzahl ICD-10-Codes ermitteln pro Versicherten
102 anzahl_icd_codes <- data.frame(table(unique(ambudaten[c(1,3)])[,1]))
103 colnames(anzahl_icd_codes) <- c("PSEUDONYM","Anzahl_ICD")
104 # Anzahl ATC-Codes ermitteln pro Versicherten
105 anzahl_atc_codes <- data.frame(table(unique(arznei[c(1,4)])[,1]))
106 colnames(anzahl_atc_codes) <- c("PSEUDONYM","Anzahl_ATC")
107 tabelle_sa_kosten_pro_ver <- merge(sa_kosten_pro_ver,anzahl_icd_codes,by="
        PSEUDONYM")
108 tabelle_sa_kosten_pro_ver <- merge(tabelle_sa_kosten_pro_ver,anzahl_atc_codes,
        by="PSEUDONYM")
109 tabelle_sa_kosten_pro_ver <- tabelle_sa_kosten_pro_ver[order(-tabelle_sa_kosten
        _pro_ver$Kosten_pro_V),]
110 tabelle_sa_kosten_pro_ver$Kosten_pro_V <- format(x=as.numeric(tabelle_sa_kosten
        _pro_ver$Kosten_pro_V),nsmall=2,justify="right")
111 write.table(x=tabelle_sa_kosten_pro_ver,file="/Users/Florian/Documents/
        Bachelorarbeit/CSV/arzneikosten_pro_versicherten.csv",sep=";",row.names=
        FALSE,col.names=TRUE,fileEncoding="latin1",dec=".")

113 # Liste der Durchschnittkosten pro Versicherten nach PLZ
114 tabelle_sa_gesamt <- sa_gesamt
115 tabelle_sa_gesamt$Durchschnittskosten_pro_V <- round(x=tabelle_sa_gesamt$
        Durchschnittskosten_pro_V,digits=2)
116 tabelle_sa_gesamt$Durchschnittskosten_pro_V <- format(x=as.numeric(tabelle_sa_
        gesamt$Durchschnittskosten_pro_V),nsmall=2,justify="right")
117 tabelle_sa_gesamt$Kosten_pro_PLZ <- format(x=as.numeric(tabelle_sa_gesamt$
        Kosten_pro_PLZ),nsmall=2,justify="right")
118 write.table(tabelle_sa_gesamt,file="/Users/Florian/Documents/Bachelorarbeit/CSV
        /durchschnitts-arzneikosten_pro_versicherten_pro_plz.csv",sep=";",row.names
        =FALSE,col.names=TRUE,fileEncoding="latin1",dec=".")
```

Quelltext A.27: R-Quelltext 7.4-PZN-Durchschnittskosten-Altersbereiche-Heatmap.R

```
### Heatmap -  Durchschnittskosten für Arzneimittel pro Versicherten (
    Altersgruppen) ###

# initial laden und ausführen
source(file="/Users/Florian/Documents/Bachelorarbeit/initial.R")

# Arzneidaten einlesen
arznei <- arzneimittel_laden()

# Die Spalten "PSEUDONYM" und "PZN" werden extrahiert
arznei_pzn <- arznei[c(1,5)]

# Ambudaten einlesen und bereinigen
ambudaten <- ambudaten_laden()

# Die Spalten "PSEUDONYM", "AGE" und "GENDER" werden extrahiert
ambu_age_gender <- unique(ambudaten[c(1,6,7)])

# PZN_Preise einlesen
pzn_preise <- pznkosten_laden()
colnames(pzn_preise) <- c("PZN","Preis")

# Laden der Shape-Datei: Sachsen-Anhalt (nach PLZ-Gebieten)
shapes <- shape_laden(karte="plz")

# Verbinden der PZN-Codes und deren Preise mit den Pseudonymen
arznei_pzn_preise <- merge(arznei_pzn,pzn_preise,by="PZN")

# Aggregation der Preise nach den Pseudonymen
kosten_pro_ver <- aggregate(arznei_pzn_preise[c(3)], by=list(arznei_pzn_preise$
    PSEUDONYM),FUN=sum)
colnames(kosten_pro_ver) <- c("PSEUDONYM","Kosten_pro_V")

# Verbinden aller Pseudonyme in SA (alle_kreise) mit dessen Geschlecht und
    Alter
ambu_alle_kreise <- merge(alle_kreise[c(1,2)],ambu_age_gender,by="PSEUDONYM")

# Verbinden der Ambu-Daten mit den Kosten-Pro-Versicherten
# INFO: Es fallen einige Pseudonyme weg, da in Ambudaten nicht vorhanden
sa_kosten_pro_ver <- merge(ambu_alle_kreise,kosten_pro_ver,by="PSEUDONYM")

# Funktion für die Berechnung und Grafikausgabe der Durchschnittskosten pro PLZ
    -Gebiet für definierte Altersgruppen und Geschlecht
```

```
40 func_pzn_kosten <- function(geschlecht,alter_von,alter_bis,break_points,
       untertitel) {

42   # Filterung der Daten nach dem Geschlecht und dem Altersbereich
43   sa_kosten_pro_ver_auswahl <- subset(x=sa_kosten_pro_ver,subset=(sa_kosten_pro
         _ver$GENDER == geschlecht & sa_kosten_pro_ver$AGE >= alter_von & sa_
         kosten_pro_ver$AGE <= alter_bis))

45   # Gesamtkosten pro PLZ-Gebiet ermitteln
46   kosten_plz_auswahl <- aggregate(sa_kosten_pro_ver_auswahl[c(5)],by=list(sa_
         kosten_pro_ver_auswahl$PLZ),FUN=sum)
47   colnames(kosten_plz_auswahl) <- c("PLZ","Kosten_pro_PLZ")

49   # Anzahl Versicherte pro PLZ-Gebiet ermitteln
50   anzahl_plz_auswahl <- sa_kosten_pro_ver_auswahl[c(1,2)]
51   anzahl_plz_auswahl <- data.frame(table(anzahl_plz_auswahl$PLZ))
52   colnames(anzahl_plz_auswahl) <- c("PLZ","Anzahl_Versicherte")

54   # Kosten und Anzahl Versicherte pro PLZ-Gebiet verbinden
55   sa_auswahl <- merge(kosten_plz_auswahl,anzahl_plz_auswahl,by="PLZ")

57   # Spalte hinzufügen: Durchschnittskosten pro Versicherten
58   sa_auswahl <- cbind(sa_auswahl,"Durchschnittskosten_pro_V"=sa_auswahl$Kosten_
         pro_PLZ / sa_auswahl$Anzahl_Versicherte)

60   # Data.frame für die Darstellung (maptools) erzeugen
61   shapes_plz_auswahl <- data.frame(shapes$plz)
62   shapes_plz_auswahl <- cbind(shapes_plz_auswahl,0)
63   # Spaltenüberschriften vergeben
64   colnames(shapes_plz_auswahl) <- c("PLZ","Durchschnittskosten_pro_V")

66   # Alle PLZ 5-stellig machen (mit Nullen auffüllen)
67   sa_auswahl$PLZ <- str_pad(string=sa_auswahl$PLZ,width=5,pad="0")

69   # Durchschnittskosten pro Versicherten in den data.frame "shapes_plz_auswahl"
         schreiben
70   for (i in 1:length(shapes_plz_auswahl$PLZ)) {tmp <- subset(x=sa_auswahl$
         Durchschnittskosten_pro_V,subset=(sa_auswahl$PLZ == shapes_plz_auswahl[i
         ,1]))
71     if (length(tmp) == 0) {
72       shapes_plz_auswahl[i,2] <- 0
73     }
74     else {
75       shapes_plz_auswahl[i,2] <- tmp
```

```
76      }
77    }

79    # Einteilung der Schritte für die Heatmap
80    class <- cut(shapes_plz_auswahl$Durchschnittskosten_pro_V,breaks=break_points
          )
81    levels(class) <- break_points[2:length(break_points)]

83    ### PLOT BEGINNT ###
84    # Plotten
85    plot(x=shapes,border="darkgrey",col=farben_shape[class],main="
          Durchschnittliche Kosten für Arzneimittel pro Versicherten")

87    # Informationen unterhalb der Heatmap anzeigen
88    mtext(text=paste("Min. Durchschnitt: ",round(min(shapes_plz_auswahl$
          Durchschnittskosten_pro_V), digits=0)," EUR","        ",
89            "Max. Durchschnitt: ",round(max(shapes_plz_auswahl$
                  Durchschnittskosten_pro_V), digits=0)," EUR","        ",
90            "Ges. Durchschnitt: ",round(sum(sa_auswahl$Kosten_pro_PLZ) / sum(sa_
                  auswahl$Anzahl_Versicherte), digits=0)," EUR",
91            "\n\nGes. Kosten: ",round(sum(sa_auswahl$Kosten_pro_PLZ) / 1000000,
                  digits=3)," Mio. EUR ","        ",
92            "Anzahl Versicherte mit erhaltenen Leistungen: ",sum(sa_auswahl$
                  Anzahl_Versicherte),
93            sep=""),side=1,cex=0.7,line=1)

95    # Altersgruppe und Geschlecht als Text hinzufügen
96    mtext(text=untertitel,side=3,cex=0.7)

98    # Legende anzeigen
99    legend('topright',fill=farben_shape,legend=paste("<=",levels(class),"EUR"),
          cex=0.85,title="Legende:",xpd=TRUE,bty="n")
100 }

102 ## Selbstdefinierte Bereiche
103 # 0-10 Jahre Mann
104 break_points <- c(0,50,100,150,200,250,300,350,400,450,500,550,600,650,2730)
105 untertitel <- "Altersgruppe: 0 - 10 Jahre  -  Geschlecht: Männlich"
106 func_pzn_kosten("M",0,10,break_points,untertitel)

108 # 0-10 Jahre Frau
109 break_points <- c(0,50,100,150,200,250,300,350,400,450,500,550,600,650,2730)
110 untertitel <- "Altersgruppe: 0 - 10 Jahre  -  Geschlecht: Weiblich"
111 func_pzn_kosten("F",0,10,break_points,untertitel)
```

```
113 # 11-20 Jahre Mann
114 break_points <- c(0,seq(from=20,to=1320,by=100))
115 untertitel <- "Altersgruppe: 11 - 20 Jahre  -  Geschlecht: Männlich"
116 func_pzn_kosten("M",11,20,break_points,untertitel)

118 # 11-20 Jahre Frau
119 break_points <- c(0,seq(from=20,to=1320,by=100))
120 untertitel <- "Altersgruppe: 11 - 20 Jahre  -  Geschlecht: Weiblich"
121 func_pzn_kosten("F",11,20,break_points,untertitel)

123 # 21-35 Jahre Mann
124 break_points <- c
        (0,100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,2000)
125 untertitel <- "Altersgruppe: 21 - 35 Jahre  -  Geschlecht: Männlich"
126 func_pzn_kosten("M",21,35,break_points,untertitel)

128 # 21-35 Jahre Frau
129 break_points <- c
        (0,100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,2000)
130 untertitel <- "Altersgruppe: 21 - 35 Jahre  -  Geschlecht: Weiblich"
131 func_pzn_kosten("F",21,35,break_points,untertitel)

133 # 36-50 Jahre Mann
134 break_points <- c
        (0,100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,5060)
135 untertitel <- "Altersgruppe: 36 - 50 Jahre  -  Geschlecht: Männlich"
136 func_pzn_kosten("M",36,50,break_points,untertitel)

138 # 36-50 Jahre Frau
139 break_points <- c
        (0,100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,5060)
140 untertitel <- "Altersgruppe: 36 - 50 Jahre  -  Geschlecht: Weiblich"
141 func_pzn_kosten("F",36,50,break_points,untertitel)

143 # 51-65 Jahre Mann
144 break_points <- c
        (0,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,2300,2780)
145 untertitel <- "Altersgruppe: 51 - 65 Jahre  -  Geschlecht: Männlich"
146 func_pzn_kosten("M",51,65,break_points,untertitel)

148 # 51-65 Jahre Frau
149 break_points <- c
        (0,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,2300,2780)
```

```
150 untertitel <- "Altersgruppe: 51 - 65 Jahre  -  Geschlecht: Weiblich"
151 func_pzn_kosten("F",51,65,break_points,untertitel)

153 # 66 - unendlich Jahre Mann
154 break_points <- c
        (0,600,710,820,930,1040,1150,1260,1370,1480,1590,1700,1810,1920,2150)
155 untertitel <- "Altersgruppe: 66 - 120 Jahre  -  Geschlecht: Männlich"
156 func_pzn_kosten("M",66,120,break_points,untertitel)

158 # 66 - unendlich Jahre Frau
159 break_points <- c
        (0,600,710,820,930,1040,1150,1260,1370,1480,1590,1700,1810,1920,2150)
160 untertitel <- "Altersgruppe: 66 - 120 Jahre  -  Geschlecht: Weiblich"
161 func_pzn_kosten("F",66,120,break_points,untertitel)
```