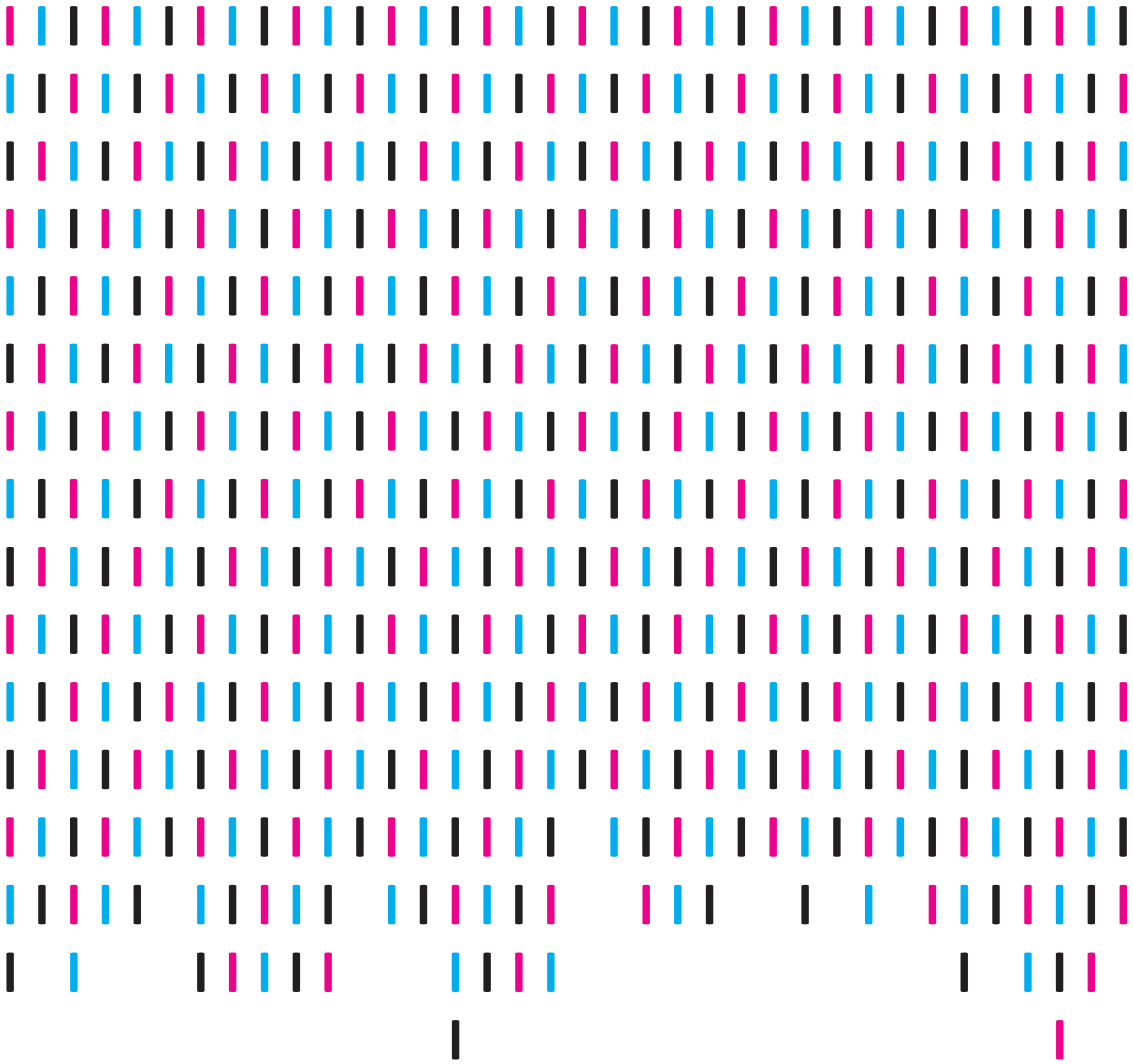

LIfBi WORKING PAPERS

Johannes Hofmann

WIE LASSEN SICH RÄUMLICHE DISTANZEN AUF STRASSENVERLAUFSEBENE IN DER SOZIALFORSCHUNG NUTZEN?

LIfBi Working Paper No. 72
Bamberg, Mai 2018

LIfBi
LEIBNIZ-INSTITUT FÜR BILDUNGSVERLÄUFE e.V.

Working Papers of the Leibniz Institute for Educational Trajectories (LIfBi)
at the University of Bamberg

The LIfBi Working Papers publish articles, expertises, and findings related to data collected and studies conducted at the Leibniz Institute for Educational Trajectories, first and foremost to the German National Educational Panel Study (NEPS).

The LIfBi Working Papers are edited by a board of researchers representing the wide range of disciplines covered by NEPS and other LIfBi studies. The series started in 2011 under the name "NEPS working papers" and was renamed in 2017 to broaden the range of studies which may be published there.

Papers appear in this series as work in progress and may also appear elsewhere. They often represent preliminary studies and are circulated to encourage discussion. Citation of such a paper should account for its provisional character.

Any opinions expressed in this series are those of the author(s) and not those of the LIfBi management or the NEPS Consortium.

The LIfBi Working Papers are available at www.lifbi.de (see section "Institute > Publications"). LIfBi Working Papers based on NEPS data are also available at www.neps-data.de (see section "Data Center > Publications").

Contact: German National Educational Panel Study (NEPS) – Leibniz Institute for Educational Trajectories – Wilhelmsplatz 3 – 96047 Bamberg – Germany – contact@lifbi.de

Wie lassen sich räumliche Distanzen auf Straßenverlaufsebene in der Sozialforschung nutzen?

Johannes Hofmann, Leibniz-Institut für Bildungsverläufe

Erkenntnisse aus dem Projekt „BildungsLandschaft Oberfranken (BiLO)“

E-Mail-Adresse des Autors:

johannes.hofmann@lifbi.de

Bibliographische Angaben bei deutschsprachigen Papers:

Hofmann, J. (2018). *Wie lassen sich räumliche Distanzen auf Straßenverlaufsebene in der Sozialforschung nutzen?* (LIfBi Working Paper Nr. 72). Bamberg: Leibniz-Institut für Bildungsverläufe, Projekt „BildungsLandschaft Oberfranken (BiLO)“.

Wie lassen sich räumliche Distanzen auf Straßenverlaufsebene in der Sozialforschung nutzen?

Abstract
Distanzberechnungen sind nicht nur in der Geografie wichtig, sondern werden auch in den Sozialwissenschaften immer häufiger aufgegriffen. Bisher hatte man sich auf aggregierte Informationen in administrativ abgegrenzten Regionen, Luftliniendistanzen, datenschutzrechtlich nicht hinreichend geklärte Distanzberechnungen durch Onlinedienste (z.B. Google Maps) oder kostenintensive Ermittlungen von Distanzen durch externe Dienstleister beschränken müssen. Mit dem Befehl `osrmtime` ermöglichen es Huber und Rust (2016) jedem Anwender (bei vorliegenden geographischen Koordinaten), mit Stata räumliche Distanzen auf Straßenverlaufsebene selbst zu berechnen. Dadurch können tatsächliche Fahrtzeiten berücksichtigt werden.

Somit stellt dieses Vorgehen eine kostengünstige und (daten-) sichere Alternative zu Online-Routing-Diensten dar, lässt sich auch über die Sozialforschung hinaus anwenden und leistet damit einen Beitrag zur fachübergreifenden, praktischen, alltäglichen Anwendung unter Berücksichtigung von internetsensiblen Datenschutzkonzepten.

Dieser Beitrag soll die Hintergründe und die Entwicklung von Distanzberechnung im Projekt „BildungsLandschaft Oberfranken (BiLO)“ (angesiedelt am Leibniz-Institut für Bildungsverläufe) aufzeigen sowie über einen Beispieldatensatz Anwendungsmöglichkeiten und Anknüpfungspunkte liefern, welche über dem Beitrag von Hubert und Rust hinausgehen.

Keywords
Distanzberechnung, Koordinaten, Georeferenzierung, Straßenverlauf, Stata, osrmtime, OpenStreetMap, Google Maps, QGIS, Datenschutz, Adressen

1 Einleitung

Distanzberechnungen sind nicht nur in der Geographie wichtig. Sie werden immer häufiger von anderen Sozialwissenschaften aufgegriffen (vgl. Abernathy (2016), Hintze und Lakes (2009)). Es ist weit verbreitet, sich auf aggregierte Informationen in administrativ abgegrenzten Regionen zu beschränken (vgl. Madelin, Grasland, Mathian, Sanders und Vincent (2009)). Doch in der Realität sind Grenzen für Gemeinden oder Landkreise nicht immer relevant für Personen, die in diesem Gebiet leben, gerade wenn sie an dessen Grenzen wohnen.

Eine erste Möglichkeit, um ohne großen Aufwand Entfernungen zwischen zwei Punkten zu generieren, sind Luftliniendistanzen. Doch auch hier können die reale Entfernungen von den errechneten Daten abweichen. Gerade in ländlichen Gebieten ohne stark ausgebautes Straßennetz (mit mehreren Alternativrouten) können diese Abweichungen zunehmen. Natürlich lassen sich Luftlinien über Faktoren annäherungsweise zu Fahrtdistanzen umrechnen, jedoch würden für ländliche und städtische Gebiete unterschiedliche Faktoren gelten, was zur Fehleranfälligkeit beitragen würde.

Eine Methode für exakte Distanzberechnungen mit geringen Hürden bieten Online-Dienste wie zum Beispiel Google Maps. Es werden keine, bzw. nur geringe Vorkenntnisse benötigt, lediglich Adressen bzw. Start- und Zielkoordinaten. Allerdings liegen auch hier Restriktionen vor, die Projekte mit hohem Anspruch an den Datenschutz vor Probleme stellen. Es werden oft nur eine begrenze Anzahl an Anfragen pro Tag zugelassen. Bei Google Maps liegt diese Obergrenze bei 2500. Für mehr Anfragen wird ein kostenpflichtiger Premium-Account benötigt (vgl. Google Developers (2018)). Doch ein weitaus größeres Problem ist die Ungewissheit, wo und wie die Daten gespeichert werden, wenn mit ihnen Distanzen berechnet werden. Da Koordinaten gleichzusetzen sind mit Adressen, ist es für viele wissenschaftliche Projekte aus Datenschutzgründen nicht möglich, diese online auslesen zu lassen (Potenzial zur Reidentifikation der Personen und Unklarheit, wo die Daten abgelegt und wie diese womöglich wiederverwendet werden).

Um dem Datenschutz gerecht zu werden, können auch Ermittlungen von Distanzen durch externe Dienstleister in Betracht gezogen werden. Diese Methode kann mit den datenschutzrechtliche Anforderungen konform gehen, dies erfordert allerdings das explizierte Einverständnis der Teilnehmenden, was wiederum Auswirkungen auf die Teilnahmequote haben kann. Außerdem stehen viele Forschungsprojekte unter strikten finanziellen Restriktionen und externe Ermittlungen können hohe Kosten nach sich ziehen.

Vor diesen Herausforderungen stand auch das Projekt „BildungsLandschaft Oberfranken (BiLO)“ welches von der Oberfrankenstiftung gefördert wird und am LIfBi angesiedelt ist. Es startete 2014 mit einer Laufzeit von fünf Jahren. Am Beispiel Oberfrankens bearbeitet es die Frage nach dem Zusammenspiel von Bildung und Raum. Dabei wird insbesondere der Zusammenhang von individuellen Bildungsentscheidungen und (wahrgenommenen) Bildungsangeboten, unter Berücksichtigung der sozialen Herkunft untersucht. Entsprechend ist die Verknüpfung von Daten zum Angebot von Bildungseinrichtungen mit Individualdaten zu Bildungsentscheidungen sowie die physische Verortung von beiden in Relation zueinander von zentraler Bedeutung. Um diese Daten zu generieren führte das Projekt BiLO insgesamt zwei Bevölkerungsbefragungen durch.

Um die Ziele des Projektes BiLO umzusetzen, war es von hoher Wichtigkeit, reale Fahrdistanzen zu ermitteln um möglichst genau die Entfernung zwischen Wohn- und Bildungsort zu beschreiben und bei Einzugsgebieten von Einrichtungen nicht auf politische Grenzen wie Landkreise angewiesen zu sein. Diese Anforderungen kombiniert mit dem hohen Schutz der Privatsphäre von Einzelpersonen und einem begrenzten finanziellen Budget schloss von Anfang an viele Möglichkeiten der Distanzberechnung aus. Eine Möglichkeit, die all diesen Anforderungen gerecht wird, ist das Stata-Ado `osrmtime` von Stephan Huber und Christoph Rust (vgl. Huber und Rust (2016)).

Der Beitrag beschreibt das breite Anwendungsspektrum des Ados von Huber und Rust am Beispiel des Projekts BiLO und zeigt auf, welche Vorbereitungen notwendig waren. Es hält auch Beispielhaft verschiedene stata commands bereit, die als Vorlage zur Berechnung von Distanzmaßen, Umkreisen und Einzugsgebieten in anderen Projekten verwendet werden können.

2 Die Distanzberechnung im Projekt BiLO

2.1 Datenaufbereitung und -ablage

Die erste Bevölkerungsbefragung des Projektes „BildungsLandschaft Oberfranken (BiLO)“ wurde vom Erhebungsinstitut infas durchgeführt. Dazu wurde bei den Einwohnermeldeämter über infas eine zufällige Stichprobe gezogen. Name, Adressen und Merkmale wie Alter bzw. Name der Eltern bei Kindern unter 14 Jahren wurden an infas übermittelt. Durch infas fand auch die Georeferenzierung der Adressen statt, die daran anschließend dem Projekt BiLO für die Distanzberechnungen zu den Bildungsangeboten übermittelt wurden. Um eine hohe Datensicherheit zu gewährleisten, wurden die Daten nach der Lieferung auf einem LIfBi-internen Server abgelegt, auf welchen nur Projektmitarbeiter Zugriff haben und der nicht mit dem Internet in Verbindung steht, sodass ein unbefugter Zugriff durch Dritte ausgeschlossen wird. Die Kontakt- und Adressdaten der Bildungsanbieter wurden – mit Ausnahme der Kinderpflegepersonen – aus öffentlich zugänglichen Quellen recherchiert bzw. eingekauft (vgl. Leibniz-Institut für Bildungsverläufe e.V. (2018)). Für diese wurde die Georeferenzierung durch das Projekt BiLO selbst durchgeführt, da hier die Anschriften bereits öffentlich verfügbar waren und eine Georeferenzierung über Google Maps zu keiner Gefährdung des Datenschutzes führen konnte.

Von Anfang an musste im Projekt BiLO eine einheitliche Projektion und Schreibweise der Koordinaten festgelegt werden, um keine internen Inkonsistenzen zu generieren. Das Projekt verwendet die Projektion des World Geodetic System 1984 (WGS 84), da dieses auch von Kartendiensten wie Google Maps oder OpenStreetMap unterstützt wird und stellt diese Koordinaten in Dezimalschreibweise dar, da diese von `osrmtime` benötigt wird. Natürlich wäre eine Datenablage auch in anderen Projektionen (z.B. UTM - Koordinatensystem) und Schreibweisen (z.B. Grad, Minuten und Sekunden) möglich und denkbar, jedoch sollte die Datenaufbereitung möglichst wenige Arbeitsschritte beinhalten um so die Fehleranfälligkeit auf einem geringen Niveau zu halten.

In diesem Sinne wird im Projekt auch kein aufwendiges Datenbankmanagementsystem (wie MySQL oder Microsoft Access) verwendet, sondern Microsoft Excel eingesetzt. Die gesamte Adressdatenbank unterteilt sich in zwei Dateien: eine nur für Koordinaten von Personen und eine mit Adressen und Koordinaten von Anbieter.

Sowohl Personen als auch Anbieter verfügen über eine eindeutige, fortlaufende sechsstellige ID. Für Städte, Gemeinden, Landkreise etc. wird als ID der offizielle Amtliche Gemeindeschlüssel (AGS) verwendet.

Die Datei für die Personen beinhaltet lediglich deren ID und die Koordinaten (von infas ermittelt).

Die Datei für die Anbieter stellt eine weitaus größere Informationsquelle dar. Neben Name, Straße, Hausnummer, Postleitzahl und Ort wurden über die Georeferenzierung zusätzlich Koordinaten (getrennt nach geographischer Breite und Länge), Amtlicher Gemeindeschlüssel (AGS), Gemeinde, Landkreis, Regierungsbezirk und Bundesland ermittelt. Darüber hinaus wurden zusätzliche Informationen wie Telefon- und Faxnummer, Homepage, Name der Leitung und E-Mail-Adresse mit abgelegt (soweit diese zu ermitteln waren). Die Überschriften der Tabelle sind möglichst kurz gehalten, haben aber eindeutige Bezeichnungen, da diese Namen nach dem Import zu Stata auch als Variablennamen genutzt werden können.

Grundsätzlich muss darauf gedacht werden, dass Koordinaten in Excel als Text formatiert sind, da die Dezimalschreibweise mit Punkt von Excel nicht als Zahl erkannt wird, und der Punkt im Deutschen automatisch gelöscht wird. Ein ähnliches Problem stellte der AGS dar, dieser wurde jedoch ohne führende Null als Zahl abgelegt, da eine eindeutige Identifizierung dennoch gegeben ist.

Für eine Georeferenzierung über mmqgis (Minn (2018)) in QGIS (QGIS Development Team (2018)) (siehe nächstes Kapitel) wird der Straßenname und Hausnummer innerhalb einer Zelle benötigt. Dennoch kann es von Vorteil sein, von Beginn an, die beiden Informationen getrennt zu erfassen, da der Zeitaufwand für eine (temporäre) Zusammenführung wesentlich geringer ist, als eine nachträgliche Trennung. Über den Excel-Befehl `[=A1&""&B1]` werden zwei Zellen mit zusätzlichen Leerzeichen dazwischen zusammengefügt.

Sollte ein Import der Daten in ein anderes Programm von Nöten sein (so kann QGIS keine XLSX-Datei erkennen, Stata allerdings schon), empfiehlt sich statt Excel das Datenformat CSV (Trennzeichen getrennt) zu verwenden, da dieses von den meisten Programmen gelesen wird. Oft ist die nachträgliche Codierung nach UTF-8 nötig. Im Projekt BiLO wird dies durch das Programm „Notepad++“ sichergestellt.

2.2 QGIS

QGIS ist ein geographisches Informationssystem (GIS), welches seit seiner Veröffentlichung 2002 unter der GNU Public License (GPL) herausgegeben wird. Diese Lizenz gewährleistet es, dass die Software nicht nur kostenlos nutzbar, sondern auch für erfahrene Nutzer frei veränderbar ist (QGIS Development Team (2018)). QGIS bildet durch seine vielfältigen Anwendungsmöglichkeiten eine Alternative zu dem Softwarepaket ArcGIS, welches zwar häufig in der Forschung genutzt wird, allerdings durch einen hohen Anschaffungspreis für wissenschaftliche Projekte mit eingeschränktem finanziellen Rahmen nur bedingt attraktiv ist. QGIS ist auch über den Kostenaspekt hinaus durch die sehr detaillierte Dokumentation und einen hohen Grad an Erweiterungsmöglichkeiten durch Add-ons auch für zunächst unerfahrenere, jedoch interes-

sierte Nutzer schnell nutzbar.

Für das Projekt „BildungsLandschaft Oberfranken (BiLO)“ war gerade die Erweiterung mmqgis von Michael Minn (Minn (2018)) von großer Hilfe, da damit die Georeferenzierung der Bildungsanbieter durchgeführt wurde. Die Entscheidung, ob mmqgis die Daten von Google Maps oder OpenStreetMap als Referenzierungsgrundlage verwenden soll wurde über einen Test zur Fehlerquote getroffen. Dieser Test ergab eine Fehleranfälligkeit (nicht oder falsch gefundene Adressen) bei Google Maps von 2,5%. Bei OpenStreetMap lag diese Fehleranfälligkeit bei über 5% und damit mehr als doppelt so hoch. Diese Fehler wurden mithilfe der Postleitzahlen erkannt (s.u.) und anschließend per Hand nachcodiert. Die Nutzung von Google und dem Internet ging, wie bereits geschrieben, mit dem Datenschutz konform, da es sich hierbei um öffentlich zugängliche, selbstrecherchierte Adressen von Bildungsanbietern handelte. Die täglich zugelassenen 2500 Anfragen wurden im Projekt (außer für anfängliche Tests) nie überschritten. Die Zeit, welche zur Georeferenzierung benötigt wird, hängt von der Rechenleistung und der Internetverbindung ab, wird aber selbst bei 2500 Abfragen nicht mehr als 90 Minuten in Anspruch nehmen. Während dieser Zeit sollte der Rechner nur eingeschränkt genutzt werden, um die Stabilität von QGIS nicht zu gefährden. Kommt es zu einem Abbruch, kann es passieren, dass alle bis zu diesem Zeitpunkt errechnetet Koordinaten gelöscht sind - allerdings zählen diese Abfragen in das 2500-Limit mit hinein.

Zum Import benötigen QGIS und mmqgis (wie bereits im Abschnitt ”Datenaufbereitung und -ablage”beschrieben) eine schlichte CSV-Datei in der mindestens die Adresse (Straßenname und Hausnummer) und der Ort vorliegt. Zusätzliche Daten wie Posteilzahl (PLZ), (Bundes)Land oder Regierungsbezirk können von Vorteil sein. Außerdem muss darauf geachtet werden, dass die CSV-Datei eine Codierung von UTF-8 aufweist. Dies gilt auch für den Import von bereits referenzierten Daten. Hierzu sind lediglich die Breiten- und Längengrade von Bedeutung. QGIS erkennt verschiedene Schreibweisen und versucht die Projektion selbst zu ermitteln.

Um evtl. Fehler in der erfolgten Georeferenzierung aufzudecken, wurde eine Prüfung über Postleitzahlen implementiert: In QGIS wurde die Funktion „Spacial Join“ verwendet um die Postleitzahl zu ermitteln und in Excel wurde diese ermittelte Angabe mit der Postleitzahl der original Adresse verglichen. Stimmte diese nicht überein, wurden die Koordinaten gelöscht und die Adresse zunächst als ”nicht referenziert”behandelt (siehe nächster Abschnitt). Diese Funktion bietet dem Projekt BiLO allerdings noch eine weitere Anwendungsmöglichkeit. Allgemein gesprochen, werden einem Punkt aufgrund seiner Verortung im Raum zusätzliche Informationen angespielt. So kann für eine referenzierte Adresse der zugehöre allgemeine Gemeindeschlüssel (AGS) ermittelt werden. Dieser ist nötig, um die Angaben einer Person oder Einrichtung mit strukturellen Daten (z.B. Bevölkerungszahl der entsprechenden Gemeinde) anzureichern.

Wird eine Adresse nicht von QGIS erkannt, bzw. nicht referenziert, so kann dies verschiedene Gründe haben. Es könnte ein Tipp-Fehler vorliegen, die Adresse könnte in einem Straßenzug liegen, welcher (noch) nicht von Google erfasst wurde (z.B. Neubaugebiet) oder Teile der Adresse stimmen nicht überein (z.B. Ort liegt in einem anderen Bundesland als angegeben). Sollten keine oder nur sehr wenige Adressen referenziert werden, kann der Fehler auch den ganzen Datensatz betreffen oder die Internetverbindung beeinträchtigt sein. Liegt die Fehleranfälligkeit im einstelligen Prozentbereich, ist eine manuelle Nachcodierung per Hand möglich. Im Internet finden sich einige Seiten, deren Ziel es ist, Adressen in Koordinaten zu konvertieren. Bei

vielen Angeboten wird das Ergebnis auch sofort auf einer Karte angezeigt und kann somit gut direkt kontrolliert werden. Dieser Prozess ist zwar zeitaufwändig, trägt jedoch zu einer hohen und verlässlichen Informationsdichte und Datenqualität bei.

QGIS kann über die Ermittlung der Koordinaten hinaus auch für eine visuelle Aufbereitung der Daten genutzt werden. Politische Grenzen und topographische Karten (als Hintergrund) können von OpenStreetMap generiert werden. So können nicht nur referenzierte Adressen mit zusätzlichen Informationen (durch Farbgebung, Form oder Größe) angezeigt werden, sondern auch Gemeinden oder Landkreise zum Beispiel nach Einwohnerdichte eingefärbt werden. Diese zusätzlichen Informationen können entweder durch die eigenen Befragungen generiert oder durch frei zugängliche Informationen der Landesämter (Bayerische Landesamt für Statistik (2016)) angereichert werden. Die Verknüpfung dieser Angaben findet entweder über die ID, den AGS oder durch ßpacial Joinßtatt. Auch lassen sich so Bezüge zwischen den Adressen gut visuell darstellen - so werden Einzugsgebiete von Bildungsanbieter oder Nutzungen von Einrichtungen durch einzelne Personen sichtbar.

Über die Ermittlung der Koordinaten hinaus wird QGIS auch für eine visuelle Aufbereitung der Daten genutzt. Jegliche Kartendarstellung, welche durch BiLO veröffentlicht wurde, konnte mit QGIS erstellt werden. Bis auf die Georeferenzierung der Daten wurde aus rechtlichen Gründen auf die Nutzung von Google Maps verzichtet.

Erfordert die Georeferenzierung eine anschließende Anonymisierung der Koordinaten (z.B. für eine Veröffentlichung oder Weitergabe an Dritte) so gibt es in QGIS die Möglichkeit die Punkte mit anderen Koordinaten zu ersetzen. Das Projekt BiLO verwendet hierzu geographische Mittelpunkte aller Gemeindeteile. Sollte dieser Grad der Anonymisierung noch zu genau sein, lassen sich natürlich auch Gemeinde- oder Landkreismittelpunkte oder ein von QGIS automatisch generiertes Punktgitter verwenden. QGIS errechnet nun den am nächsten gelegenen Anonymisierungspunkt, ordnet diesen der entsprechenden realen Adresse zu und ermöglicht somit, dass die originalen Koordinaten gelöscht werden können.

2.3 OpenStreetMap

Gegründet im Jahre 2004 wurde OpenStreetMap (OSM) noch vor Google Maps (2005) veröffentlicht. Allerdings muss gesagt werden, dass die anfänglich zugrunde liegenden Daten noch sehr spärlich waren und erst mit der Zeit gewachsen sind. Anders als Google Maps werden die Daten von freiwilligen Helfern gesammelt und in das System eingespeist. Ähnlich wie QGIS ist OpenStreetMap Quelloffen und läuft unter der Open Data Commons Open Database Lizenz (ODbL) (OpenStreetMap Foundation (2018b); OpenStreetMap Foundation (2018a)). Auf diesem Weg der Datengewinnung sind zwar Fehler möglich, jedoch wird das Angebot von einer großen Community gestützt (laut eigenen Angaben vom 8. November 2017: 4,3 Millionen registrierte Nutzer (OpenStreetMap Foundation (2018c))) und Ungenauigkeiten können von anderen Personen ausgebessert werden. Es ist davon auszugehen, dass die Datendichte auch in Zukunft weiter wachsen wird. Der Informationsgehalt ist zwar eher auf urbanerer Gebiete konzentriert, jedoch sind zumindest Straßen auch für kleine Ortschaften eingezeichnet. Da dies für die Distanzberechnung in Projekt BiLO ausreicht, konnte die Datengrundlage von OSM ohne große Bedenken für Distanzberechnungen genutzt werden (anders als für die Georeferenzierung, da hierfür nicht ausreichend Adressdaten vorhanden sind).

Die Kartendaten liegen, ähnlich wie bei Google Maps mit der Projektion WGS 84 und in Dezimalschreibweise vor, was bei der Nutzung mit beachtet werden muss, wenn in der eigenen Arbeit eine andere Projektion genutzt werden soll.

Ein großer Vorteil von OpenStreetMap gegenüber Google Maps der vielen Datenschutzbestimmungen entgegenkommen kann, ist die Möglichkeit, Daten von OSM auch offline nutzen zu können (somit ist sichergestellt, dass personenbezogene Daten, wie z.B. Adressen nicht ins Internet und auf unkontrollierte Server gelangen). Es existieren Lösungen (z.B. von http://www.geofabrik.de/), um Kartendaten herunterzuladen, lokal zu speichern und damit zu rechnen oder sie beliebig zu ändern. Auch QGIS verfügt über verschiedene Schnittstellen, um Daten von OpenStreetMap offline zu nutzen (z.B. über die Erweiterung Quick OSM, welche sich der Overpass API von OSM bedient).

Obwohl OSM teilweise sogar über einen höheren Informationsgehalt als Google Maps verfügt (z.B. eingezeichnete Hydranten) war die Nutzung durch das Projekt BiLO nur auf sehr grundlegenden Daten beschränkt.

Bei der Nutzung von OSM fallen allerdings auch Nachteile auf: so ist die Anzahl der Abfragen (z.B. bei der Georeferenzierung), ähnlich wie bei Google Maps, beschränkt und OSM verfügt (noch) nicht über eine eigene Routing-Implementierung wie Google Maps. Jedoch nutzen andere Anbieter wie die Open Source Routing Machine die Datengrundlage von OSM und ergänzen diese durch eine eigene Berechnungslösung.

2.4 Open Source Routing Machine

Wie bereits beschrieben, ist es dem Nutzer von Open Street Map nicht möglich, sich direkt eine Verbindung zwischen zwei Punkten berechnen zu lassen. Jedoch ist die Datengrundlage valide genug, um eine quelloffene Alternative zu Google Maps darzustellen. Aus diesem Grund nutzen die Autoren der Open Source Rouring Machine (OSRM) die Daten von Open Street Map, um Mittels einer selbstentwickelten Software Fahrtdistanzen und -dauer zu berechnen. Auch diese Lösung ist Quelloffen und läuft unter der (vereinfachten) 2-Klausel-BSD Lizenz Luxen (2018b).

Das Angebot ist über die URL http://project-osrm.org/ zu erreichen und hat eine geringe Anforderung an die Softwarekenntnis seiner Nutzer.

Die Entwickler greifen auf die Kartendaten von OSM zurück und können so nicht nur Einbahnstraßen berücksichtigen, sondern haben auch eine Hierarchie der Straßentypen implementiert, um zu verhindern, dass schwer zugängliche Straßen (z.B. Waldwege) oder Straßen mit geringen Geschwindigkeitslimit (z.B. 30er-Zonen) aufgrund einer möglicherweise kürzeren Strecke (Weglänge) gut zu befahrenen Straßen (z.B. Bundesstraßen) vorgezogen werden. Die Fahrtdauer berechnet sich mit Durchschnittsgeschwindigkeiten, die in der Stadt oder im Überlandverkehr erreicht werden. So wird in der Stadt nicht mit exakt 50 km/h sondern mit 40 km/h gerechnet, um erhöhte Verkehrsdichte oder Ampelphasen annähernd miteinzubeziehen Luxen (2018a).

Des Weiteren bietet das Angebot verschiedene Profile für unterschiedliche Verkehrsmittel zur Wegberechnung an; so können für Fahrzeuge andere Wege genutzt werden, als für Fahrrad-

fahrer oder für Personen, welche zu Fuß unterwegs sind. Diese Profile wirken sich nicht nur auf die Hierarchie der Straßentypen aus, sondern natürlich auch auf die Durchschnittsgeschwindigkeiten und die damit verbundene Wegdauer.

Es wurden im Projekt BiLO Vergleiche zwischen der Open Source Routing Machine und Google Maps zur Zuverlässigkeit der Distanzberechnung durchgeführt. Selbst bei größeren Distanzen von bis zu 1000 Kilometern unterscheiden sich die Ergebnisse nur geringfügig. Ein Vorteil, den Google Maps hat, ist die Einbeziehung der aktuellen Verkehrssituation in die Berechnung. So werden nicht nur Alternativrouten vorgeschlagen, sondern auch die Fahrtdauer angepasst. Diese Informationen umfassen nicht nur aktuelle Staumeldungen oder kurzfristige Sperrungen sondern bezieht auch eine höhere Verkehrsdichte zu Stoßzeiten wie dem Berufsverkehr mit ein. Doch die Vorteile der Quelloffenheit und die Möglichkeit der internetunabhängigen Nutzung überwiegen diese Nachteile gegenüber Google Maps deutlich.

Die Entwickler der Open Source Routing Machine bieten die Möglichkeit an, eine Instanz auf dem eigenen Rechner zu implementieren um somit die gesamt Software auch offline nutzen zu können. Dies hätte den Vorteil, dass eigene Änderungen und Anpassungen an die Software, wie zum Beispiel die Änderung der Durchschnittsgeschwindigkeit, jederzeit möglich sind. Jedoch gestalten sich Einrichtung und Bedienung wenig benutzerfreundlich und erfordern ein erhöhtes Verständnis an Softwareentwicklung mit textbasierter alltäglicher Nutzung viele Codezeilen. Somit wurde diese Möglichkeit im Projekt BiLO verworfen und statt dessen das Stata-Ado `osrmtime` von Huber und Rust vorgezogen, welches die Software der OSRM nach einer zwar anspruchsvollen, jedoch gut dokumentierten, einmaligen Installation ohne großes Vorwissen in Stata über ein do-File nutzen lässt.

2.5 osrmtime

Das Stata-Ado `osrmtime` wurde von Stephan Huber und Christoph Rust an der Uni Regensburg entwickelt und 2016 im Stata Journal dokumentiert (Huber und Rust (2016)). Der Artikel gibt nicht nur einen guten Überblick über die Funktionsweisen und Vorteile des Ados, sondern ist zeitgleich eine Anleitung zur Einrichtung und Nutzung. Dadurch ist es weitaus benutzerfreundlicher, als eine eigene Instanz der Open Source Routing Machine lokal einzurichten, zumal sich die Dokumentation von Huber und Rust an Nutzer mit geringerem Vorwissen richtet.

`Osrmtime` nutzt die Kartengrundlage von OpenStreetMap (bezogen über `Geofabrik.de` (Geofabrik GmbH Karlsruhe und OpenStreetMap Contributors (2017))), aufbereitet durch den Befehl `osrmprepare`, nutzt die Softwareumgebung der Open Source Routing Machine und implementiert diese in einem Programm, welches von Stata über einen Befehl ausgeführt werden kann. Dies bringt viele Vorteile über die bereits angesprochen Benutzerfreundlichkeit hinaus mit sich. So ist eine Berechnung unabhängig vom Internet möglich, es existieren keine Restriktionen der Anzahl der Abfragen mehr und die Rechenzeit ist nicht mehr von der Netzwerkumgebung abhängig, sondern lediglich von der Leistung des Rechners selbst. Bei den handelsüblichen Laptops im Projekt BiLO werden ca. 45 Berechnungen pro Sekunde erreicht. Mit leistungsstärkeren Rechnern kann diese Zahl allerdings leicht übertroffen werden.

Im Vorfeld zur Berechnung muss sich darüber Gedanken gemacht werden, in welchem Gebiet die Distanzberechnung stattfinden soll. Je kleiner das Gebiet, desto geringer die Datengröße

der Karte und desto kürzer die Vorbereitungszeit. Befinden sich alle Punkte innerhalb einer Stadt wird auch nur eine kleine Karte benötigt. Im Projekt BiLO wird eine Deutschland-Karte verwendet, welche eine Datengröße von ca. fünf Gigabyte besitzt (nach Aufbereitung durch `osrmprepare`). Es ist allerdings auch eine größere Kartengrundlage möglich, gleichwohl in dem Fall mit einer wesentlich größerer Datenmenge und Aufbereitungszeit gerechnet werden muss. Ist die Aufbereitung erst abgeschlossen, lässt sich die Karte auf Dauer ohne Einschränkung nutzen. Allerdings muss gesagt sein, dass sich diese Datengrundlage natürlich nicht automatisch selbst aktualisiert. Ist eine aktuellere Karte gewünscht, muss der Aufbereitungsprozess mit einer neueren Karte abermals durchgeführt werden. Darüber hinaus muss schon bei der Kartenaufbereitung festgelegt werden, ob eine Distanzbrechung für Fahrzeuge, Fahrradfahrer oder Fußgänger stattfinden soll. Wenn unterschiedliche Fortbewegungsmittel in Betracht gezogen werden, empfiehlt es sich die selbe Karte auf drei unterschiedliche Weisen (jeweils mit dem bestimmten Fortbewegungsmittel im Blick) aufzubereiten - `osrmprepare` bietet dazu eine entsprechende Funktion an.

Wie bereits angesprochen, benötigt `osrmtime` die Koordinaten der Start- und Zielpunkte mit der Projektion WGS 84 und in Dezimalschreibweise. Dies liegt in der Tatsache begründet, da die Karten von OpenStreetMap in eben dieser Konfiguration vorliegen und von Geofabrik genauso bereitgestellt werden. Eine Konvertierung der Start- und Zielkoordinaten kann allerdings ohne großen Aufwand im Vorfeld über QGIS stattfinden.

2.6 Distanzberechnung

Die vergangenen Kapitel hatten das Ziel, verschiedene Alternativen darzustellen und letztendlich zu erläutern, warum im Projekt „BildungsLandschaft Oberfranken (BiLO)“ für eine Distanzberechnung das Ados `osrmtime` von Huber und Rust gewählt wurde. Es ermöglicht ein Verfahren um die Entfernungen zwischen zwei Punkten auf Straßenverlaufsebene zu errechnen und gleichzeitig die hohen Ansprüche des Datenschutzes gerecht zu werden. Die Nachteile, keine aktuellen Staumeldungen oder hohes Verkehrsaufkommen mit einbeziehen zu können werden durch die Vorteile, unabhängig von Internet und einer restriktiven Anzahl an Abfragen zu sein, ausgeglichen.

Bei der Aufbereitung der Straßenkarte für Deutschland durch `osrmprepare` wurde der Cache von 5 auf 50 Gigabyte erweitert, um die Größe der Karte zu berücksichtigen. Bei einem 5, 10, 20 und 25 Gigabyte großen Cache kam die Aufbereitung der Straßenkarte auch nach mehreren Stunden Rechendauer jeweils nicht zu Stande.

Die aufbereitete Karte schließt insgesamt 14 Dateien mit ein (gleicher Dateiname, unterschiedliche Dateiendung). Diese 14 Dateien wurden in einen, für alle Mitarbeiter des Projektes zugänglichen Ordner auf dem Institut-internen Server gelegt, sodass alle mit der gleichen Kartengrundlage arbeiten können und nicht jeder Rechner mit dem Kartenpaket ausgestattet werden muss. Außerdem verfügt dieser Server nicht über einen Zugang zum Internet und kann somit nicht von außerhalb des Gebäudes genutzt werden.

Auf den Befehl `osrmtime` soll an dieser Stelle nicht weiter eingegangen werden. Die Dokumentation im Stata Journal ist dazu mehr als ausreichend. Im Projekt BiLO sind bei der Umsetzung allerdings zwei Herausforderungen aufgetaucht, die kurz erläutert werden sollen.

Zum einen kann es passieren, dass `osrmtime` nicht ausgeführt wird, obwohl der Befehl richtig in der do-File hinterlegt wurde und Stata daraufhin erfolglos auf eine Reaktion bzw. Antwort des Ados wartet. Eine Lösung für dieses Vorkommnis besteht darin, die 14 Kartendaten zuerst in einen Unterordner zu verschieben und direkt danach an den ursprünglichen Ablageort zurück zu kopieren. Dabei ist es wichtig, dass die Daten im ersten Schritt verschoben und im zweiten Schritt kopiert werden. Danach lässt sich das do-File problemlos ausführen.

Zum anderen kann es ab einer hohen Zahl an Berechnungen passieren, dass der Cache des Rechners überlastet wird. Die Höhe der Anzahl ist von der Leistung des Rechners abhängig. Bei den handelsüblichen Laptops des Projektes BiLO passiert dies jenseits der zehn Millionen Abfragen (pro do-File). Wenn ein solches Problem auftritt, empfiehlt es sich, den Ursprungsdatensatz zu teilen, die Berechnungen einzeln durchzuführen und anschließend zu einem Datensatz mit `append` zusammenzuführen.

3 Anwendungsbeispiele

Während im ersten Teil dieses Working Papers über den Weg hin zur Distanzberechnung im Projekt „BildungsLandschaft Oberfranken (BiLO)“ berichtet wurde, geht dieser folgende Abschnitt auf die möglichen Anwendungsmöglichkeiten dieses Verfahrens ein. Um die Verständlichkeit der Daten zu erleichtern, befinden sich im Anhang zwei Dokumente. Zum einen ein Beispieldatensatz mit fiktiven Personen- und Einrichtungsdaten und zum anderen ein do-File mit den hier aufgeführten Befehlen, um eine reale Anwendung anschaulich präsentieren zu können.

Die hier aufgeführten Anwendungsbeispiele setzten voraus, dass bereits eine Georeferenzierung der Start- sowie der Zielpunkte erfolgte, das Ado `osrmtime` eingerichtet und eine entsprechende Straßenkarte aufbereitet wurde (Siehe Kapitel „QGIS“ und „osrmtime“).

Im Projekt „BildungsLandschaft Oberfranken (BiLO)“ liegt das Hauptaugenmerk auf die Nutzung und Wahrnehmung von Bildungseinrichtungen meist in Relation zum Wohnort der Teilnehmenden, für das im Folgenden einige Beispiele für Distanzmaße aufgeführt werden. Jedoch schließt dies natürlich andere Anwendungen nicht aus. Sobald geographische Koordinaten für Start- und Zielpunkte vorliegen, lassen sich diese Beispiele auch auf Distanzberechnungen in anderen Kontexte übertragen.

3.1 Erläuterungen zum Beispieldatensatz

Alle hier dargestellten Koordinaten sind speziell für dieses Working Paper mit QGIS frei erstellt worden. Sie befinden sich alle in Wohngebieten im Regierungsbezirk Oberfranken, sind jedoch nicht Teil der Erfassung und Befragung im Zuge des Projektes „BildungsLandschaft Oberfranken (BiLO)“. Jegliche Übereinstimmungen mit realen Adressen sind daher rein zufällig.

Der Beispieldatensatz (`Beispieldatensatz.dta`) stellt einen sehr rudimentären Befragungsdatensatz dar und beinhaltet eine Personen-ID für jeden Fall und einer Variable, die wiederum die ID einer Einrichtung enthält, die besucht bzw. genutzt wurde. Darüber hinaus gibt es einen Koordinaten-Datensatz (`Koordinaten_bev.dta`), der zu jeder Personen-ID die zugehörigen Koordinaten des Wohnorts enthält und zu einen zweiten (`Koordinaten_an.dta`), der zu jeder

Einrichtungs-ID deren Koordinaten enthält sowie eine (fiktive) Variable zum Einrichtungstyp.

Der hier folgende Abschnitt ist in fünf Themenbereiche gegliedert, welche anhand des Beispieldatensatzes bearbeitet werden können. Jeder Themenbereich wiederum ist unterteilt in die Fragestellungen „Welchen Vorteil bieten diese Informationen?“ und „Wie werden diese Informationen generiert?“. Die erste Fragestellung soll den Hintergrund des Themenbereiches erläutern, während die zweite Fragestellung die praktische Herangehensweise mit Stata darlegt.

3.2 Distanz zur angegebenen Einrichtung

Welchen Vorteil bieten diese Informationen? Die Angabe, wie lange eine Person zu ihrer angegebenen Einrichtung fährt, kann bereits im Fragebogen abgefragt werden. Dies bildet eine subjektive Einschätzung der Distanz ab. Um sich allerdings bei den weiteren Analysen zusätzlich auf eine objektive Distanzgrundlage (auf Straßenverlaufsebene) stützen zu können, kann `osrmtime` verwendet werden.

Wie werden diese Informationen generiert? Zur Berechnung der Distanz zur angegebenen Einrichtung bilden die Koordinaten der befragten Person die Startpunkte und die Koordinaten der angegebenen Einrichtungen die Zielpunkte.

Ausgangspunkt ist der Befragungsdatensatz. Es empfiehlt sich, diesen zu duplizieren und auf die wichtigsten Variablen (ID der Person und der angegebenen Einrichtung) zu reduzieren. Später können die hier generierten Informationen an den Befragungsdatensatz angespielt werden. Dies ist keine zwingende Voraussetzung, dient jedoch der Sauberkeit und Übersicht im Hauptdatensatz.

Der im Befragungsdatensatz enthalten ID der besuchten Einrichtung werden nun die Koordinaten der Einrichtung aus `Koordaten_an.dta` als auch die Koordinaten aus `Koordinaten_bev` je über `merge` zugespielt.

Nun folgt die Distanzberechnung durch `osrmtime`. Daran anschließend werden die Sprungdistanzen und der Fehlercode überprüft (vgl. Huber und Rust (2016)). Nun können, der Übersicht wegen, die meisten Variablen gelöscht werden, sodass lediglich die IDs und die zugehörige Fahrtdistanz in Meter sowie die Fahrzeit in Sekunden im Datensatz verbleiben. Eine Umrechnung in Kilometer und Minuten ist zwar nicht zwingend erforderlich, wird hier jedoch mit aufgeführt. Dies stellt die einfachste Anwendung von `osrmtime` dar.

3.3 Distanz zur nächstgelegenen Einrichtung

Welchen Vorteil bieten diese Informationen? In einer Befragung kann lediglich ermittelt werden, welche Einrichtung eine Person besucht. Es wird zu geringem Erfolg führen, eine Person nach den Einrichtungen zu fragen, welche sie nicht besucht. Somit ist es von Vorteil, das Bild einer objektiv vorhandene Angebotslandschaft zu zeichnen und eventuelle Alternativen zu der besuchten Einrichtung am Computer zu ermitteln.

Wie werden diese Informationen generiert? Die Berechnung der nächstgelegenen Einrichtung baut auf den Schritten zur Berechnung der Distanz zur angegebenen Einrichtung auf. `Osrmtime` ist nicht dafür vorgesehen, dies direkt zu ermitteln, es erfordert jedoch nur geringen zusätzlichen Aufwand, um die nächste Einrichtung zu identifizieren. Dazu ist es nötig einen Datensatz zu erstellen, der jede Person jeder Einrichtung gegenüberstellt. Dazu werden zuerst die IDs der *Personen* nach folgendem Schema dupliziert (nach der Anzahl der Einrichtungen, hier 3): a,b,c,a,b,c,a,b,c. Daran anschließend werden die *Einrichtungen* nach folgendem Schema dupliziert (nach Anzahl der Personen, hier 3) a,a,a,b,b,b,c,c,c. Die Gesamtzahl der Zeilen im Datensatz ergibt sich aus [Anzahl der Personen] * [Anzahl der Einrichtungen]. `Osrmtime` ermittelt nun jede mögliche Distanz, was zu einem längeren Rechenprozess führen kann. Der Datensatz wird nun als sowohl im Long- als auch im Wide-Format gespeichert. Dies wird vorausschauend ausgeführt, da der `reshape`-Befehl bei großen Datensätzen große Zeit in Anspruch nimmt. Diese Datensätze sind nun der Ausgang für weitere Analysen.

Um das nächstgelegene Angebot zu identifizieren, wird nun der Datensatz im Long-Format geladen, und je Personen-ID nach Distanz sortiert, sodass die erste Zeile der Person die nächstgelegene Einrichtung beinhaltet, die zweite Zeile die zweitnächste, usw.. Nun werden alle Zeilen einer ID außer der ersten Zeile gelöscht. Somit enthält der Datensatz wie zu Beginn alle Personen, gegenübergestellt mit den ihnen am nächstgelegenen Einrichtungen. Die Umrechnung in Kilometer und Minuten wird hier abermals zusätzlich mit aufgeführt.

3.4 Ist das genutzte Angebot auch das nächstgelegene?

Welchen Vorteil bietet diese Informationen? Nutzt eine Person tatsächlich die nächstgelegene Einrichtung, oder wird ein zusätzlicher Weg auf sich genommen? Wie groß ist dieser zusätzliche Weg?

Wenn noch weitere Informationen aus der Befragung vorhanden sind, lässt sich auch die folgende Fragen stellen: Kannte die Person Einrichtungen, die näher lagen, nutzte diese jedoch nicht? Dies kann eine Aussage über die Informiertheit der Person geben und auch über deren Entscheidung, anderen Gründe eine höhere Priorität zu geben als dem zusätzlichen Streckenaufwand.

Wie wird diese Information generiert? An den eben generierten Datensatz mit den Distanzen zu den angegeben (besuchten) Einrichtungen wird der soeben erstellte Datensatz mit den nächstgelegenen Einrichtungen hinzugespielt.

Über eine neu gebildete Variable [`ist_nächste`] wird ermittelt, ob sich die Distanz zur besuchten Einrichtung von der Distanz zur nächstgelegene unterscheidet.

Weiter bietet es sich hier an, zu ermitteln, welche zusätzliche Distanz in Kauf genommen wurde. Dazu wird eine Variable erstellt, die die Differenz zwischen gewählter und nächstgelegener Einrichtung berechnet.

3.5 Objektiv verfügbares Angebot innerhalb eines persönlichen Umkreises

Welchen Vorteil bietet diese Informationen? Wurde ermittelt, welche Distanz eine Person bereit ist zurückzulegen, um ein Angebot zu erreichen, liegt es nahe, zu messen, welche anderen Angebote in dem gleichen Umkreis liegen.

Wie wird diese Information generiert? Ausgangspunkt für diese Fragestellung ist der eben erstellte Datensatz über alle möglichen Distanzen im Long-Format (`Distanz_bev_alle_an_long.dta`). Auch hier erfolgt zunächst die Umrechnung von Meter in Kilometer bevor die Distanz zur im Befragungsdatensatz angegebenen Einrichtung über die Einrichtungs-ID hinzugespielt werden. Nun werden für jede Person jene Einrichtungen gelöscht, die weiter entfernt liegen, als die gewählte. Wurde für alle Personen im Vorfeld ein fester Umkreis gewählt (z.B. 25 km), so fällt der `merge`-Befehl weg und es genügt hier ein [`drop if dist_alle_km>25`].

Da nun die Gesamtzahl der im persönlichen Umkreis liegenden Einrichtungen interessiert, erfolgt eine Umformung in die Wide-Formatierung.

Liegen die Daten nun im Wide-Format vor, wird die anfängliche Anzahl der Personen wiederhergestellt. Somit werden auch Teilnehmer mit eingeschlossen, die bei der Frage ein Missing hatten, oder bei denen keine Einrichtung innerhalb des Umkreises liegt, sollte eine feste Distanz als Umkreis festgelegt worden sein.

Durch den `egen`-Befehl werden alle Zellen pro Zeile gezählt, welche bei der Distanz kein Missing haben und in der neu erstellten Variable [`Anz_Gesamt`] aufsummiert. Doch nicht nur die Gesamtzahl ist hier möglich, sondern auch die Summe einzelner Informationen (hier als Beispiel der Einrichtungstyp). Es sind drei Typen vorhanden, somit erfolgen drei gleich aufgebaute Schleifen, welche pro Zeile die Angabe des Einrichtungstyps zählen und in einer jeweils neu erstellten Variable festhalten.

In einem letzten Schritt werden nur die IDs der Personen, sowie die in dem Verfahren neu erstellten Variablen im Datensatz belassen.

3.6 Objektive Einzugsgebiete von Einrichtungen

Welchen Vorteil bietet diese Informationen? Obwohl hier bisher immer aus der Perspektive der Nutzer gesprochen wurde und die gesamte Fragestellung darauf ausgelegt ist, so kann es auch von Vorteil sein, den teilnehmenden Anbieter als Incentive Ergebnisse aus ihrer Sicht anzubieten. Hierbei gilt es allerdings auf die Fallzahlen zu achten. Wie viele teilnehmende Personen nutzen ein bestimmtes Angebot, welches Einzugsgebiet besitzt die Einrichtung und welche Distanzen wurde minimal und maximal für diese Einrichtung zurückgelegt?

Wie wird diese Information generiert? Zuerst wird der Datensatz mit den bereits berechneten Distanzen zwischen Person und genutzter Einrichtung geladen. Die Variable [`var1`] wird zu der bekannten Variablenbezeichnung [`id_an`] umbenannt. Nun wird über den `egen`-Befehl die Maximalen und Minimalen Distanzen und Fahrtzeiten ermittelt und in der generierten Variablen festgehalten. Im nächsten Schritt werden zuerst die Nutzer pro Einrichtung gezählt und

erneut mit dem egen-Befehl in einer neuen Variable eingetragen.

Da Einrichtungen mehrmals genannt wurden, werden alle Duplikate gelöscht und Einrichtungen hinzugespielt, welche nicht in der Befragung genannt wurden.

Um die Ausprägungen der Variable über die Anzahl der Nutzer verständlich zu halten, werden Missings (entstanden durch den vorangegangenen `merge`-Befehl) durch eine Null ersetzt.

4 Zusammenfassung

Mit dem Fortschreiten der modernen Medien und der Möglichkeit, Nutzer in die Datengewinnung miteinzubeziehen, wachsen frei zugängliche Kartendienste und Geoinformationssysteme schneller denn je. Gerade für Einzelpersonen oder finanziell eingeschränkte Forschungsprojekte ermöglichen diese Lösungen neue Wege der Datengewinnung. Distanzberechnungen auf Straßenebene mit einem Computer ohne Internetzugang und fremde Hilfe selbst durchzuführen, stellt eine kostengünstige, benutzerfreundliche und (daten-)sichere Alternative zu Online-Routing-Diensten dar.

Die hier dargelegten Abläufe sollen nicht nur über die im Projekt „BildungsLandschaft Oberfranken (BiLO)“ entwickelten Verfahren aufklären und die zur Hilfe genommenen Programme und Ados dokumentieren, sondern auch Anregungen für andere interessierte Nutzer liefern. Natürlich fällt der erste Einstieg auf Grund der technischen Hürden nicht immer leicht, jedoch sinken diese Hürden recht schnell, je länger man sich mit den Abläufen und der zugrunde liegenden Literatur beschäftigt.

Obwohl die Beispiele stark auf die Sozialforschung zugeschnitten sind, sollte der hier vorliegende Text auch als Anstoß gesehen werden, über die Sozialforschung hinaus zu denken und geographische Lösungsansätze auch auf andere Fragestellungen in anderen Themengebieten und Wissenschaften anzuwenden, denn die Verfahren bilden fachübergreifende, praktische, alltägliche Anwendungen unter Berücksichtigung der hohen Ansprüche von Datenschutzkonzepten.

Literatur

Abernathy, D. (2016). *Using geodata and geolocation in the social sciences: Mapping our connected world*. SAGE Publications Ltd.

Bayerische Landesamt für Statistik, M. (2016). *Genesis-online (bayern).* Zugriff auf https://www.statistikdaten.bayern.de/genesis/

Geofabrik GmbH Karlsruhe und OpenStreetMap Contributors. (2017). *Geofabrik.de.* Zugriff auf http://download.geofabrik.de/

Google Developers. (2018). *Google Maps APIs Pricing and Plan.* Zugriff auf https://developers.google.com/maps/pricing-and-plans/

Hintze, P. & Lakes, T. (2009). *Geographically Referenced Data in Social Science. A service paper for SOEP data users* (Bericht). DIW Berlin. German Institute for Economic Research.

Huber, S. & Rust, C. (2016). Calculate travel time and distance with OpenStreetMap data using the Open Source Routing Machine (OSRM). *The Stata Journal*, *16* (2), 416-423.

Leibniz-Institut für Bildungsverläufe e.V. (2018). *Datenschutz- und Datensicherheitskonzept des Projektes „BildungsLandschaft Oberfranken (BiLO)“.* (Unveröffentlicht)

Luxen, D. (2018a). *Git hub: Project-osrm/osrm-backend.* Zugriff auf https://github.com/Project-OSRM/osrm-backend

Luxen, D. (2018b). *Project-osrm open source routing machine.* Zugriff auf http://project-osrm.org/

Madelin, M., Grasland, C., Mathian, H., Sanders, L. & Vincent, J.-M. (2009). Das „maup“: Modifiable areal unit – problem oder fortschritt? *Informationen zur Raumentwicklung*, *Heft 10/11*, 645-660.

Minn, M. (2018). *mmqgis.* Zugriff auf http://michaelminn.com/linux/mmqgis/

OpenStreetMap Foundation. (2018a). *Faq.* Zugriff auf https://wiki.openstreetmap.org/wiki/FAQ

OpenStreetMap Foundation. (2018b). *Open database license.* Zugriff auf https://wiki.openstreetmap.org/wiki/Open_Database_License

OpenStreetMap Foundation. (2018c). *Stats.* Zugriff auf https://wiki.openstreetmap.org/wiki/Stats

QGIS Development Team. (2018). *QGIS Geographic Information System. Open Source Geospatial Foundation Project.* Zugriff auf http://qgis.osgeo.org

Anhang

A Stata Beispieldatensatz

Das hier aufgeführte do-File kann als ganzes kopiert und von Stata 14 (oder neuer) ausgeführt werden, um somit den Beispieldatensatz zu generieren. Zu beachten ist lediglich, zuerst einen Arbeitsordner anzugeben (bei `cd "..."`).

```
*******************************************************************************
*** Erstellen der Koordianten und eines Beispieldatensatzes -----------***
*******************************************************************************
*** Autor:   Johannes Hofmann -----------------------------------------***
*** Projekt: "BildungsLandschaft Oberfranken (BiLO)" ------------------***
*** Datum:   01.05.2018 -----------------------------------------------***
*******************************************************************************

*** Präambel ***
cd "..."
version 14
set more off
capture log close

clear

*** Datensatz mit Koordinaten der Personen erstellen ***
input id_bev lat_bev long_bev
101 49.904063 10.911655
102 49.943561 11.597442
103 50.302206 11.912612
104 50.253937 10.960922
105 50.098413 10.995597
106 50.093861 11.448440
107 49.941204 11.584396
108 49.895265 10.910110
109 49.709976 11.073532
110 49.941204 11.581649
111 50.233737 11.306991
112 50.096506 11.436080
113 49.937669 11.587142
114 50.033033 12.008743
115 50.306591 11.916732
end

*** Variablenlabel hinzufügen ***
label var id_bev "ID der Person"
label var lat_bev "Geographische Breite der Person"
label var long_bev "Geographische Länge der Person"
```

```
*** Datensatz mit Koordinaten der Personen sichern ***
save Koordinaten_bev.dta, replace

clear

*** Datensatz mit Koordinaten der Einrichtungen erstellen ***
input id_an lat_an long_an str4 typ
201 50.302206 11.919479 typ1
202 50.090337 11.433334 typ2
203 49.902890 10.911140 typ1
204 49.719725 11.059434 typ3
205 49.934133 11.567916 typ2
206 50.183640 11.791763 typ2
207 50.232859 11.323471 typ3
208 50.098266 10.989761 typ1
209 49.841224 10.496749 typ2
210 50.300451 11.132583 typ1
end

*** Variablenlabel hinzufügen ***
label var id_an "ID des Anbieters"
label var lat_an "Geographische Breite des Anbieters"
label var long_an "Geographische Länge des Anbieters"
label var typ "Einrichtungstyp"

*** Datensatz mit Koordinaten der Einrichtungen sichern ***
save Koordinaten_an.dta, replace

clear

*** Beispieldatensatz erstellen ***
input id_bev var1
101 203
102 204
103 203
104 209
105 201
106 203
107 206
108 206
109 208
110 207
111 210
112 206
113 203
114 201
115 203
```

```
end

*** Variablenlabel hinzufügen ***
label var id_bev "ID der Person"
label var var1 "Antwort zum besuchten Angebot"

*** Beispieldatensatz sichern ***
save Beispieldatensatz.dta, replace

exit
```

B Stata Distanzberechnungen

Das hier aufgeführte do-File kann als ganzes kopiert und von Stata 14 (oder neuer) ausgeführt werden, um somit den angegebenen Verfahren in Kapitel 3 zu folgen. Zu beachten ist lediglich, zuerst einen Arbeitsordner (bei `cd "..."`) und die Kartenquelle (bei `osrmtime` unter `mapfile(...)`) anzugeben.

```
***-------------------------------------------------------------------***
*** Anwendungen der Distanzberechnung -----------------------------***
***-------------------------------------------------------------------***
*** Autor:   Johannes Hofmann -------------------------------------***
*** Projekt: "BildungsLandschaft Oberfranken (BiLO)" ----------------***
*** Datum:   01.05.2018 -------------------------------------------***
***-------------------------------------------------------------------***

*** Präambel ***
cd "..."
version 14
set more off
capture log close

*-----------------------------------------------------------------------*
* Berechnung der Distanz zum angegebenen Anbieter ---------------------*
*-----------------------------------------------------------------------*

*** Beispieldatensatz laden ***
use Beispieldatensatz.dta, clear

*** Hinzuspielen der Koordinaten der Personen ***
merge m:1 id_bev using Koordinaten_bev.dta, keepusing(lat_bev long_bev)
drop if _merge==2
drop _merge

*** Hinzuspielen der Koordinaten der genannten Einrichtungen ***
rename var1 id_an
merge m:1 id_an using Koordinaten_an.dta, keepusing(lat_an long_an)
drop if _merge==2
```

```
drop _merge
rename id_an var1

*** Datenaufbereitung vor Distanzberechnung ***
sort id_bev
destring lat_bev long_bev lat_an long_bev, replace

*** Distanzberechnung ***
osrmtime lat_bev long_bev lat_an long_an, mapfile(...) nocleanup

*** Überprüfung der Jumpdistances und des Fehlercodes ***
fre jumpdist1
fre jumpdist2
fre return_code

*** Löschen der Jumpdistances, des Fehlercodes und der Koordianten ***
drop jumpdist1 jumpdist2 return_code lat_bev long_bev lat_an long_an

*** Distanz in Kilometer umrechnen ***
gen dist_km = distance/1000
form %9.2f dist_km
label var dist_km "Distanz in Kilometer"
drop distance

*** Fahrtzeit in Minuten umrechnen ***
gen dur_mil = duration*1000
gen dur_min = minutes(dur_mil)
form %9.2f dur_min
label var dur_min "Fahrzeit in Minuten"
drop dur_mil
drop duration

*** Distanzberechnung sichern ***
save Distanz_bev_angeg_an.dta, replace

*-------------------------------------------------------------------------*
* Berechnung der Distanz zum nächstegelegenen Anbieter ------------------*
*-------------------------------------------------------------------------*

*** Koordianten der Anbieter laden ***
use Koordinaten_an.dta, clear

*** Vervielfältigung der Anbieter und temporär Zwischenspeichern ***
expand 15
sort id_an
save temp1.dta, replace
```

```
*** Koordianten der Personen laden ***
use Koordinaten_bev.dta, clear

*** Vervielfältigung der Personen ***
gen cluster = 1
expandcl 10, gen (sort_var) cluster (cluster)
sort sort_var id_bev
drop sort_var cluster

*** Vervielfältigten Datensatz der Anbieter hinzuspielen ***
merge 1:1 _n using temp1.dta, force
drop _merge

*** Temporären Datensatz löschen
erase temp1.dta

*** Datenaufbereitung vor Distanzberechnung ***
sort id_bev id_an
destring lat_bev long_bev lat_an long_bev, replace

*** Distanzberechnung ***
osrmtime lat_bev long_bev lat_an long_an, mapfile(...) nocleanup

*** Überprüfung der Jumpdistances und des Fehlercodes ***
fre jumpdist1
fre jumpdist2
fre return_code

*** Löschen der Jumpdistances, des Fehlercodes und der Koordianten ***
drop jumpdist1 jumpdist2 return_code lat_bev long_bev lat_an long_an

*** Long-Datensatz mit allen Distanzen speichern ***
save Distanz_bev_alle_an_long.dta, replace

*** Reshape von Long auf Wide (mit ID der Einrichtung als Schlüssel) ***
gen double_id_an = id_an
order id_bev double_id_an distance duration id_an, first
reshape wide id_an distance duration typ, i(id_bev) j(double_id_an)

*** Wide-Datensatz mit allen Distanzen speichern ***
save Distanz_bev_alle_an_wide.dta, replace

*** Long-Datensatz mit allen Distanzen laden ***
use Distanz_bev_alle_an_long.dta, clear

*** Nächste Einrichtung ermittlen ***
sort id_bev distance
```

```
quietly by id_bev: gen count = cond(_N==1,0,_n)
drop if count>1
drop count

*** Distanz in Kilometer umrechnen ***
gen dist_km = distance/1000
form %9.2f dist_km
label var dist_km "Distanz in Kilometer"
drop distance

*** Fahrtzeit in Minuten umrechnen ***
gen dur_mil = duration*1000
gen dur_min = minutes(dur_mil)
form %9.2f dur_min
label var dur_min "Fahrzeit in Minuten"
drop dur_mil
drop duration

*** Distanzberechnung sichern ***
save Distanz_bev_nächst_an.dta, replace

*------------------------------------------------------------------------*
* Wurde das nächstgelegene Angebot genutzt? ----------------------------*
*------------------------------------------------------------------------*

*** Distanzberechnung zu angegebenem Anbieter laden ***
use Distanz_bev_angeg_an.dta, clear

*** Variablen umbenennen ***
rename dist_km dist_km_angeg
rename dur_min dur_min_angeg

*** Datensatz mit nächstgelegenen Einrichtungen hinzuspielen ***
merge 1:1 id_bev using Distanz_bev_nächst_an.dta, keepusing(dist_km dur_min)
drop if _merge==2
drop  _merge

*** Dazugespielte Variablen umbenennen ***
rename dist_km dist_km_nächst
rename dur_min dur_min_nächst

*** Neue Varibale "ist_nächste" erstellen, um zu identifizeren, ***
*** ob nächste Einrichtung gewählt wurde ***
gen ist_nächste =.
label var ist_nächste "Wurde das nächstgelegene Anbebot genutzt?"

replace ist_nächste = 1 if dist_km_nächst == dist_km_angeg
```

```
replace ist_nächste = 0 if dist_km_nächst < dist_km_angeg
replace ist_nächste = -99 if dist_km_nächst > dist_km_angeg

label define nächste 0"nicht nächste" 1"ist nächste" -99"Fehler"
label val ist_nächste nächste

*** Zusätzlich gefahrener Weg ***
gen zusatz =  dist_km_angeg - dist_km_nächst
label var zusatz "Zusätzlich gefahrener Weg"

*** Distanzberechnung sichern ***
save Wurde_nächst_Angebot_genutzt.dta, replace

*-------------------------------------------------------------------------*
* Angebot im bestimmten Umkreis der Bevölkerung ------------------------*
*-------------------------------------------------------------------------*

*** Long-Datensatz mit allen Distanzen laden ***
use Distanz_bev_alle_an_long.dta, clear

*** Distanz in Kilometer umrechnen ***
gen dist_alle_km = distance/1000
form %9.2f dist_alle_km
label var dist_alle_km "Distanz in Kilometer"
drop distance

*** Gewählte Distanz der Person hinzuspielen ***
merge m:1 id_bev using Distanz_bev_angeg_an.dta, keepusing(dist_km)
drop if _merge==2
drop _merge

*** Ermittlung der Einrichtungen im persönlichen Umkreis ***
sort id_bev dist_alle_km
drop if dist_alle_km >dist_km

*** Löschen der Fahrzeiten und Distanz (gewählte Einrichtung) ***
drop duration dist_km

*** Pro Person die Einrichtungen durchzählen ***
quietly by id_bev: gen count = cond(_N==1,1,_n)

*** Von Long auf Wide, mit Identifikator "count" ***
reshape wide id_an dist_alle_km typ, i(id_bev) j(count)

*** mit ursprünglichem Wide-Datensatz mergen ***
merge 1:1 id_bev using Distanz_bev_alle_an_wide.dta, nogen keepusing(id_bev)
sort id_bev
```

```
*** Zählen, wieviele Einrichtungen es im definierten Umkreis gibt ***
egen Anz_Gesamt=rownonmiss(dist_alle_km*)
label var Anz_Gesamt "Anzahl der Einrichtungen im persönlichen Umkreis"

*** Zählen, wieviele Einrichtungen es pro Typ 1 im Umkreis gibt ***
gen Anz_Typ1=0
label var Anz_Typ1 "Anzahl von Typ 1 im persönlichen Umkreis"
foreach var of varlist typ* {
replace Anz_Typ1 = Anz_Typ1 + 1 if `var'=="typ1"
}

*** Zählen, wieviele Einrichtungen es pro Typ 2 im Umkreis gibt ***
gen Anz_Typ2=0
label var Anz_Typ2 "Anzahl von Typ 2 im persönlichen Umkreis"
foreach var of varlist typ* {
replace Anz_Typ2 = Anz_Typ2 + 1 if `var'=="typ2"
}

*** Zählen, wieviele Einrichtungen es pro Typ 3 im Umkreis gibt ***
gen Anz_Typ3=0
label var Anz_Typ3 "Anzahl von Typ 3 im persönlichen Umkreis"
foreach var of varlist typ* {
replace Anz_Typ3 = Anz_Typ3 + 1 if `var'=="typ3"
}

*** Nur ID der Personen und Anzahl der Einrichtungen behalten ***
keep id_bev Anz_Gesamt Anz_Typ1 Anz_Typ2 Anz_Typ3

*** Datensatz sichern ***
save Anz_an_Umkreis.dta, replace

*-------------------------------------------------------------------------*
* Einzugsgebiet von Einrichtungen ----------------------------------------*
*-------------------------------------------------------------------------*

*** Long-Datensatz mit allen Distanzen laden ***
use Distanz_bev_angeg_an.dta, clear

*** Indetifikator für Anbieter umbennen ***
rename var1 id_an

*** Maximal und Minimal gefahrener Weg pro Einrichtung ermitteln ***
bysort id_an: egen max_dist = max(dist_km)
label var max_dist "Maximale Distanz, die ein Nutzer zurückgelegt hat"
bysort id_an: egen min_dist = max(dist_km)
label var min_dist "Minimale Distanz, die ein Nutzer zurückgelegt hat"
```

```
bysort id_an: egen max_dur = max(dur_min)
label var max_dur "Maximale Fahrzeit, die ein Nutzer zurückgelegt hat"
bysort id_an: egen min_dur = min(dur_min)
label var min_dur "Minimale Fahrzeit, die ein Nutzer zurückgelegt hat"

*** Anzahl der Personen pro Einrichtung ermitteln ***
sort id_an id_bev
quietly by id_an: gen count = cond(_N==1,1,_n)
bysort id_an: egen nutzer = max(count)
label var nutzer "Gesamtzahl der Nutzer"

*** Duplizierte Einrichtungen löschen ***
quietly by id_an: gen anz = cond(_N==1,1,_n)
drop if anz>1

*** Nun überflüssige Variablen löschen ***
drop id_bev dist_km dur_min count anz

*** Einrichtungen hinzuspielen, welche nicht genannt wurden ***
merge m:1 id_an using Koordinaten_an.dta, nogen keepusing(id_an)
sort id_an

*** Anzahl Nutzer auf "0" setzten, wenn Einrichtung nicht genannt wurde ***
replace nutzer=0 if mi(nutzer)

*** Datensatz sichern ***
save Einzugsgebiet_an.dta, replace
exit
```