Geoanalyse mit PROC GINSIDE
(K)eine Lösung im Zeitalter von großen Datenmengen?!

Florian Kleene
LVM Versicherung
Kolde-Ring 21
48126 Münster
f.kleene@lvm.de

Jens Blecking
LVM Versicherung
Kolde-Ring 21
48126 Münster
j.blecking@lvm.de

Zusammenfassung

Geoanalysen gewinnen nicht nur im Rahmen des strategischen CRM stetig an Bedeutung. Für Standortplanungen, Wettbewerbsbeobachtung oder Vergleiche mit zugekauften Marktdaten ist es unerlässlich herauszufinden, wo sich Kunden geografisch befinden.
Der Vorgang der exakten Geocodierung einer Adresse zu einer geografischen Position (e.g. Kolde-Ring 21, 48151 Münster → 51.949°N, 7.614°E) ist bspw. durch Webservices möglich und wurde u.a. bereits auf der 15. KSFE vorgestellt. Oftmals ist jedoch gar nicht die exakte geografische Position eines einzelnen Kunden relevant, sondern man möchte nahe beieinander liegende Kunden miteinander "verklumpen".
Die offizielle Kreisgemeindestruktur bietet eine hierarchische Gliederung Deutschlands von Bundesländern und Regierungsbezirken über Landkreise und Gemeinden bis hin zu Wohnquartieren. Die Zuordnung eines Kunden zu diesen Gebieten ist jedoch – anders als bei Postleitzahlen – oftmals nicht im operativen System erfasst worden. Die Zuordnung zu einem der Gebiete bzw. grafisch gesehen einem Polygon muss daher nachträglich erfolgen.
Mathematisch gesehen handelt es sich bei der Fragestellung um einen "Punkt-In-Polygon-Test", der mit der Schnittpunktmethode oder auch der Winkelmethode durchgeführt werden kann. SAS bietet seit der Version 9.2 mit der Prozedur PROC GINSIDE eine entsprechende Implementierung. Die Prozedur ist seither stetig weiter entwickelt und die Performance optimiert worden. Dennoch bietet die Prozedur für (sehr) viele zu testende Punkte und/oder (sehr) viele zu testende Gebiete keine akzeptablen Laufzeiten.
Der Beitrag erläutert anschaulich die Verfahren der Schnittpunkt- und Winkelmethode. Anschließend werden die Syntax, Eingabe- und Ausgabedaten der Prozedur an einem kleinen Beispiel vorgestellt. Eine Analyse der Einflüsse auf das Laufzeitverhalten bildet schließlich die Grundlage für zwei Lösungsansätze, mit denen auch ~3 Mio. Punkte (Kunden) noch effizient ~85.000 Polygonen (Wohnquartieren), die durch ~11 Mio. Punkte definiert sind, zugeordnet werden können.

Schlüsselwörter: PROC GINSIDE, Punkt-in-Polygon-Test, Geoanalyse, Big Data, Schnittpunktmethode, Winkelmethode, Kartenerstellung, Performanceverbesserung

1 Zielsetzung

Vielen Daten, die heute in operativen Systemen erfasst werden, lassen sich geografische Informationen zuordnen, da die Adresse des Kunden, Patienten etc. bekannt ist. Die Transformation von Adressen in geografische Koordinaten, d. h. Längen- und Breitengrad, wurde bereits auf der 14. KSFE und 15. KSFE [1, 2] vorgestellt (e.g. Carl-Neuberg-Str. 1, 30625 Hannover → 52.3838°N, 9.8061°E). Die Auswertung und Speicherung exakter Koordinaten zu einem Kunden kann aus datenschutzrechtlichen Gründen problematisch sein, sodass eine Zusammenfassung mehrerer Kunden zu einer Einheit („Verklumpung“) erforderlich wird. Andererseits werden exakte geografische Koordinaten typischerweise auch gar nicht benötigt, da man lediglich erfassen möchte, in welcher Region sich eine größere statistische Masse an Kunden befindet und Einzelkunden damit irrelevant werden. Ziel sollte daher zusätzlich sein, unterschiedliche Granularitätsebenen, d. h. eine Hierarchie in Bezug auf die Region, anbieten zu können, um verschiedenen Anspruchsgruppen gerecht zu werden.

Als Darstellungsform der Ergebnisse eignen sich neben einer rein tabellarischen Aufstellung der Kennzahlen insbesondere Karten, die entsprechend der Kennzahlen eingefärbt sind. Ein mögliches Ergebnis könnte die in Abbildung 1 dargestellte Karte der Gemeinde Hannover sein. Eingefärbt wurde die Karte anhand der Anzahl der Gebäude pro Gebiet in Relation zur jeweiligen Fläche, wobei ein Gebiet umso dunkler eingefärbt wurde, je höher die Gebäudedichte ist. Als Granularität wurde in diesem Beispiel die Ebene der Stadtteile gewählt.

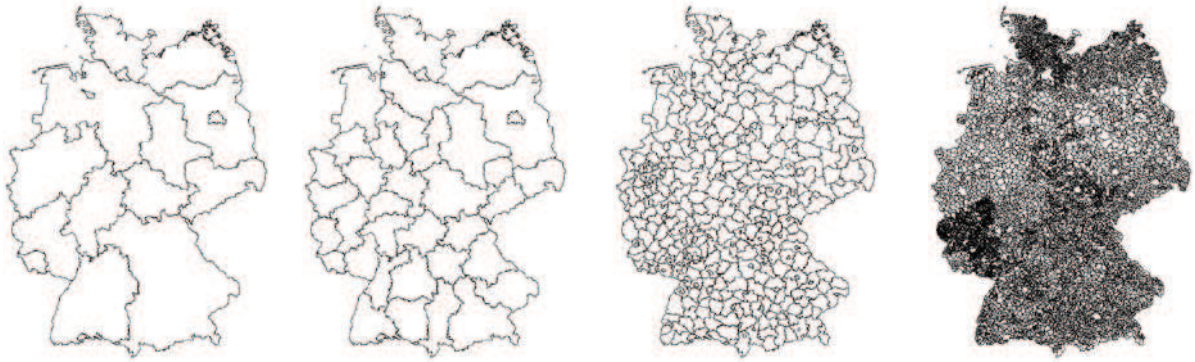
Abbildung 1: Häuserdichte auf Stadtteilebene in der Gemeinde Hannover

Eine einfache Möglichkeit wäre, die Postleitzahlen der deutschen Post zu verwenden, da diese meistens bereits im operativen System erfasst wurden. Eine Hierarchie ließe sich dann über die Anzahl der verwendeten Stellen der PLZ abbilden. Die Karten in Abbildung 2 zeigen von links nach rechts, wie zunächst nur die erste Stelle, dann die ersten zwei Stellen, usw. bis hin zu allen fünf Stellen der PLZ zur Festlegung der Gebiete verwendet wurden. Die Anzahl der Gebiete steigt dabei stetig von 10 Gebieten bei Verwendung der ersten Stelle der PLZ, über 95, 671, 3122 bis hin zu 8217 Gebieten bei Verwendung aller fünf Stellen an.

1-stellige PLZ 2-stellige PLZ 3-stellige PLZ 4-stellige PLZ 5-stellige PLZ

Abbildung 2: Postleitzahlengebiete (hierarchisch) in Deutschland

Ein Problem, das bei dieser Vorgehensweise entsteht, resultiert daraus, dass andere statistische Kennzahlen typischerweise **nicht** auf der Granularität der PLZ-Gebiete erhoben werden – man die selbst erhobenen Kennzahlen (Anzahl Kunden, Beiträge, etc.) jedoch häufig gerne in Relation zu allgemeinen statistischen Kennzahlen (Anzahl Einwohner, Kaufkraft, etc.) betrachten möchte. Das statistische Bundesamt verwendet hingegen im Allgemeinen die offizielle Kreisgemeindestruktur (KGS) in Deutschland. Diese ist ebenfalls hierarchisch aufgebaut und bietet als Granularitätsebenen Bundesländer (16 KGS2 Gebiete), Regierungsbezirke (38 KGS3 Gebiete), Landkreise (403 KGS5 Gebiete) und Gemeinden (11.368 KGS8 Gebiete) – siehe Abbildung 3. Anbieter von statistischen Kennzahlen wie bspw. Nexiga oder die GfK erweitern die Struktur häufig noch um weitere Ebenen wie bspw. Stadtteile (KGS12) und Wohnquartiere (KGS22). Die Schlüsselung auf diesen zusätzlichen Ebenen ist jedoch nicht normiert und unterliegt einer regen zeitlichen Dynamik.



Bundesländer Regierungsbezirke Landkreise Gemeinden

Abbildung 3: Kreisgemeindestruktur in Deutschland

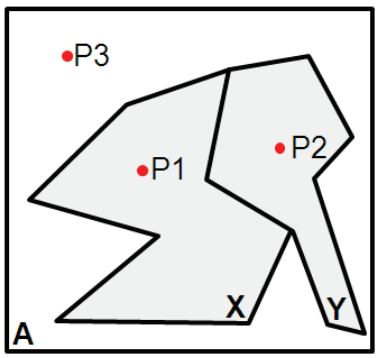
Nun könnte man mutmaßen, dass ein einfaches Mapping zwischen den Strukturen der Postleitzahlen (PLZ) und der Kreisgemeindestruktur (KGS) existiert. Dies ist leider nicht immer der Fall wie die Beispiele in Abbildung 4 zeigen:

a) 1:1 b) 1:N c) N:1 d) Überschneidung

Abbildung 4: Übereinstimmung von Postleitzahlen- und Kreisgemeindegebieten

a) In diesem Fall liegen die Grenzen des PLZ Gebiets und der Gemeinde deckungsgleich, d. h. 1:1, übereinander. Beispiel hierfür ist die Gemeinde Geeste in Niedersachsen mit der PLZ 49744 und dem KGS8 Schlüssel 03454014.
b) Städte bilden in der KGS häufig eine einzelne Gemeinde, sind jedoch nach Postleitzahlen feingliedriger aufgeteilt. Beispiel hierfür ist die Stadt Oldenburg mit dem KGS8 Schlüssel 03403000. Es existieren hingegen mehrere PLZ Gebiete (1:N) mit den Nummern 26121 bis 26135.
c) Auf der Insel Föhr zeigt sich das umgekehrte Bild. Für die gesamte Insel gilt die PLZ 25938. Sie ist jedoch in der KGS in mehrere Gemeinden wie bspw. Alkersum mit dem KGS8 Schlüssel 01054005, Borgsum mit dem KGS8 Schlüssel 01054015, usw. aufgeteilt (N:1).
d) Schließlich gibt es noch den Fall, dass die Strukturen gar nicht in einander aufgehen und sich Überschneidungen ergeben. In diesen Fällen „vertragen“ sich die Strukturen nicht und ein Mapping ist **nicht** möglich.

Da die Zuordnung von einzelnen Kunden zur KGS in den operativen Systemen meistens nicht vorliegt und auch kein „sauberes“ Mapping von PLZ zur KGS möglich ist, muss die Zuordnung nachträglich zum Analysezeitpunkt erfolgen. Mathematisch betrachtet handelt es sich dabei um einen Punkt-in-Polygon-Test, bei dem zu den Koordinaten einzelner Kunden (Punkte) das jeweilige KGS Gebiet (Polygon), in dem der Kunde lebt, ermittelt wird.



Es soll also festgestellt werden, dass der Punkt P1 in das Polygon X, der Punkt P2 in das Polygon Y und der Punkte P3 in das Gebiet A, d. h. außerhalb der Polygone X und Y fällt (siehe Abbildung 5). Es ergeben sich damit zwei mögliche Fragestellungen, die zum gleichen Ergebnis führen würden. Man geht den Fragen nach,

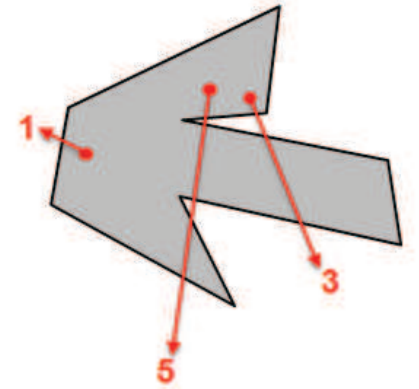
1) **ob** ein Punkt innerhalb eines bestimmten Gebiets liegt,
2) **in welches** Gebiet ein bestimmter Punkt fällt.

Abbildung 5: Punkt-in-Polygon-Test schematisch

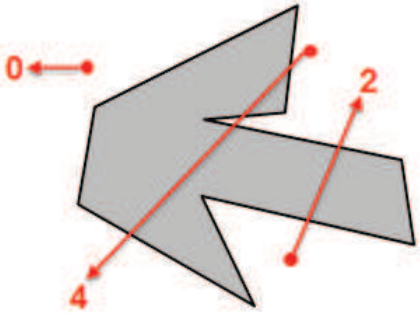
2 Mathematischer Hintergrund

2.1 Schnittpunkte, Punkt-in-Polygon-Test nach Jordan

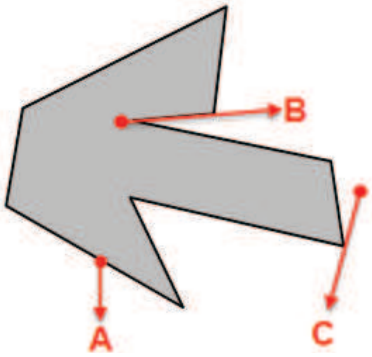
Die Schnittpunktmethode beruht auf dem jordanschen Kurvensatz, der besagt, dass „jede geschlossene Jordan Kurve eine euklidische Ebene in zwei disjunkte Gebiete zerlegt, deren gemeinsamer Rand die Jordankurve ist“. Um zu ermitteln, ob ein Punkt sich innerhalb oder außerhalb eines Polygons befindet, wird bei dieser Methode ein Strahl ausgehend vom angefragten Punkt in eine beliebige Richtung versendet. Dabei wird gezählt, wie viele Schnittpunkte der Strahl mit den Kanten des Polygons aufweist. Es können anschließend drei Fälle unterschieden werden:



Liegt der angefragte Punkt **innerhalb** des Polygons, so ergibt sich **eine ungerade Anzahl an Schnittpunkten** mit den Kanten des Polygons. In der Grafik ist die Anzahl der Schnittpunkte durch die Ziffern 1, 3 und 5 an den jeweiligen Strahlen angedeutet.



Liegt der angefragte Punkt **außerhalb** des Polygons, so ergibt sich **eine gerade Anzahl an Schnittpunkten** mit den Kanten des Polygons bzw. kein Schnittpunkt. In der Grafik ist dies wiederum durch die Ziffern 0, 2 und 4 an den Strahlen angedeutet.



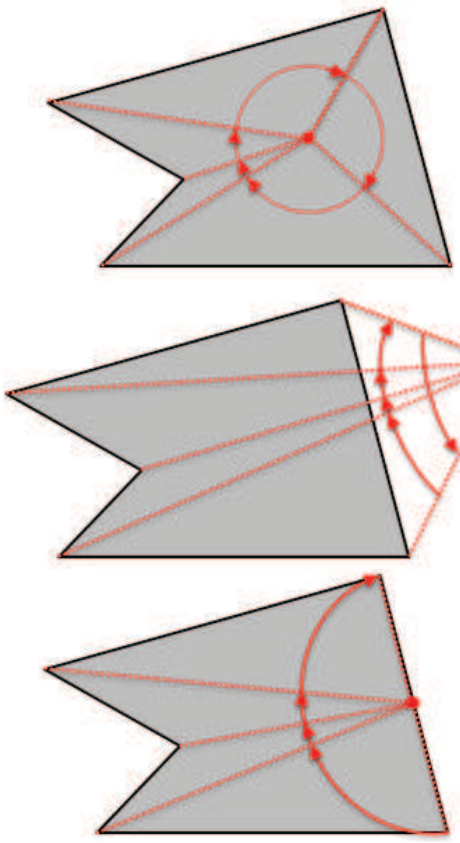
Liegt der Punkt auf einer der Kanten des Polygons (A), so handelt es sich um eine Definitionsfrage, ob der Rand mit zum Polygon gehört, oder nicht. Verläuft der Strahl „über“ eine Kante des Polygons (B), so ergeben sich unendlich viele Schnittpunkte und der Test muss mit einem Strahl in eine andere Richtung wiederholt werden. Ebenso sollte der Test wiederholt werden, wenn der Strahl eine Ecke des Polygons trifft (C).

Abbildung 6: Schnittpunktmethode grafisch

Wenn die Anzahl der Eckpunkte des Polygons n ist, beträgt Laufzeit des Verfahrens für einen angefragten Punkt O(n), da für den Test jede Kante des Polygons genau einmal betrachtet werden muss.

2.2 Winkelmethode

Eine zweite Möglichkeit den Punkt-in-Polygon-Test durchzuführen bietet die sog. Winkelmethode. Dabei werden vom angefragten Punkt ausgehend die Winkel zu allen paarweise benachbarten Eckpunkten des Polygons bestimmt und diese anschließend summiert. Winkel im Uhrzeigersinn werden dabei mit positivem und Winkel gegen den Uhrzeigersinn mit negativem Vorzeichen belegt. Es lassen sich wiederum drei Fälle unterscheiden:



Liegt der Punkt **innerhalb** des Polygons, so ergibt sich eine Winkelsumme von **360° im Winkelmaß bzw. 2 * PI im Bogenmaß**.

Liegt der Punkt **außerhalb** des Polygons, so ergibt sich eine Winkelsumme von **0° im Winkelmaß bzw. 0 im Bogenmaß**.

Falls der Punkt auf einer der Kanten des Polygons liegt, so stellt sich erneut die Frage nach der Definition, d. h. ob die Kante zum Polygon gehört oder nicht. Die Winkelsumme beträgt dann 180° im Winkelmaß bzw. PI im Bogenmaß.

Abbildung 7: Winkelmethode grafisch

Da auch bei diesem Test alle benachbarten Eckpunkte des Polygons durchlaufen werden müssen, hat auch dieser Algorithmus die Laufzeit O(n), wenn n die Anzahl der Eckpunkte des Polygons ist.

Um der Frage nachzugehen, **in welches Polygon** ein bestimmter Punkt fällt, müssten beide oben genannten Tests jeweils für alle in Frage kommenden Polygone wiederholt werden.

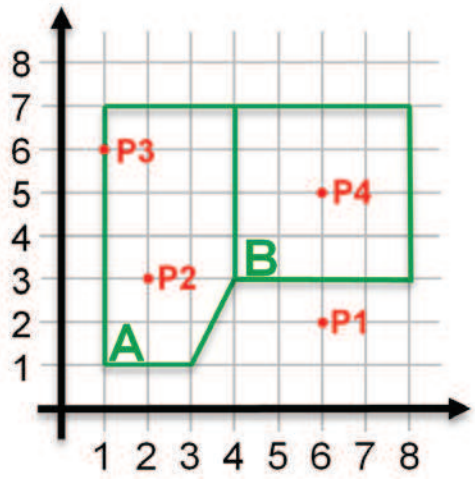
3 Lösung mit SAS – PROC GINSIDE

Seit der Version 9.2 bietet SAS mit der Prozedur PROC GINSIDE, die sich im Paket SAS Graph befindet, die Möglichkeit an, einen Punkt-in-Polygon-Test durchzuführen. Die Prozedur ermöglicht es beide oben aufgeworfenen Fragestellungen zu beantworten:

1) **Liegt** ein Punkt innerhalb eines bestimmten Gebiets?
2) **In welches** Gebiet fällt ein bestimmter Punkt?

PROC GINSIDE wurde seither weiter entwickelt und mit der Version SAS 9.3 der optionale Parameter <INCLUDEBORDER> eingeführt. In der Version SAS 9.4 wurden weiterhin die optionalen Parameter <KEEPMAPVARS> und <DROPMAPVARS> ergänzt. Ob es sich bei dem intern verwendeten Algorithmus um die Schnittpunktmethode und/oder die Winkelmethode handelt und ob ggf. weitere Optimierungen vorgenommen wurden, ist nicht veröffentlicht. Im Folgenden ist der allgemeine Aufbau der Prozedur dargestellt. Die Auswirkungen der einzelnen Parameter werden anhand eines kleinen Beispiels (siehe work.map und work.punkte sowie die grafische Visualisierung in Abbildung 8) demonstriert:

```
PROC GINSIDE
     DATA    = points-data-set
     MAP     = map-data-set
     OUT     = output-data-set
         <INSIDEONLY>
         <INCLUDEBORDER>
         <KEEPMAPVARS>       (=Default)
         <DROPMAPVARS>;
     ID id-variable(s);
RUN;
```



work.map

	id	x	y
1	A	1	1
2	A	1	7
3	A	4	7
4	A	4	3
5	A	3	1
6	A	1	1
7	B	4	3
8	B	4	7
9	B	8	7
10	B	8	3
11	B	4	3

work.punkte

	nr	x	y
1	P1	6	2
2	P2	2	3
3	P3	1	6
4	P4	6	5

Abbildung 8: Punkt-in-Polygon-Test mit SAS (Code, Grafik, Eingabe Datasets)

Das PROC GINSIDE Statement beinhaltet drei Data Sets, die die zu prüfenden Punkte (DATA=), die Gebiete, d. h. Polygone (MAP=) und das Ergebnis Data Set (OUT=) benennen. Neben diesem Statement wird noch exakt ein ID Statement erwartet, welches eine oder mehrere Variablen benennt, die die „Namen der Polygone“ bezeichnen. Diese

Variable(n) werden im Ergebnis Data Set ergänzt und zeigen die Zuordnung des jeweils angefragten Punktes zu einem der Polygone an.

Ruft man die Prozedur ohne weitere Parameter auf, so enthält das Ergebnis Data Set für jeden der angefragten Punkte (DATA=) exakt eine Zeile (siehe Abbildung 9, work.ergebnis1). Punkte, die nicht zugeordnet werden konnten, weil sie in keinem der Polygone liegen, enthalten in der bzw. in den ID Variablen ein Missing Value. Der Rand gehört per Default nicht zum Polygon.

Ergänzt man den optionalen Parameter <INSIDEONLY>, so werden die Punkte, die nicht in eines der Polygone fallen, verworfen. Das Ergebnis Data Set enthält demzufolge nur Punkte, die zugeordnet werden konnten (siehe Abbildung 9, work.ergebnis2).

Mit dem ebenfalls optionalen Parameter <INCLUDEBORDER> kann gesteuert werden, dass auch der Rand eines Polygons diesem noch zugeordnet wird. Dann wird im oben gezeigten Beispiel auch der Punkt P3 noch dem Polygon A zugeordnet. Im Ergebnis Data Set wird automatisch die Variable _ONBORDER_ ergänzt. Diese hat dann den Wert 1, wenn der entsprechende Punkt auf den Rand des Polygons fällt (siehe Abbildung 9, work.ergebnis3). Liegt ein Punkt auf den Grenzen mehrerer Polygone, so wird dieser dem zuerst verarbeiteten Polygon zugeordnet. Welches dieses ist lässt sich leider nicht einfach vorhersagen. Möchte man für diese Punkte hingegen selbst festlegen, welchem Polygon sie zugeordnet werden, so empfiehlt sich der wiederholte Aufruf von PROC GINSIDE mit jeweils nur einem Polygon und den Optionen <INCLUDEBORDER> und <INSIDEONLY>.

```
PROC GINSIDE
      DATA  = work.punkte
      MAP   = work.map
      OUT   = work.ergebnis1;
      ID id;
RUN;

PROC GINSIDE
      DATA  = work.punkte
      MAP   = work.map
      OUT   = work.ergebnis2
      INSIDEONLY;
      ID id;
RUN;
PROC GINSIDE
      DATA  = work.punkte
      MAP   = work.map
      OUT   = work.ergebnis3
      INCLUDEBORDER;
      ID id;
RUN;
```

work.ergebnis1

	nr	x	y	id
1	P1	6	2	
2	P2	2	3	A
3	P3	1	6	
4	P4	6	5	B

work.ergebnis2

	nr	x	y	id
1	P2	2	3	A
2	P4	6	5	B

work.ergebnis3

	nr	x	y	id	_ONBORDER_
1	P1	6	2		0
2	P2	2	3	A	0
3	P3	1	6	A	1
4	P4	6	5	B	0

Abbildung 9: Punkt-in-Polygon-Test mit SAS (Code, Ergebnis Datasets)

Spezialfall Enklaven[1]:
Als Enklaven bezeichnet man Gebiete, die vollständig von anderen Gebieten umschlossen werden (Beispiel: Bundesland Berlin befindet sich vollständig im Bundesland Brandenburg). Damit der Punkt-in-Polygon-Test auch in diesen Fällen das gewünschte Ergebnis liefert, sind die Geometriedaten häufig wie nachfolgend gezeigt aufgebaut. Dadurch kann es dazu kommen, dass die Variable _ONBORDER_ fälschlicherweise den Wert 1 bekommt, obwohl sich der Punkt auf keiner „echten“ Grenze befindet (siehe schematische Darstellung in Abbildung 10).

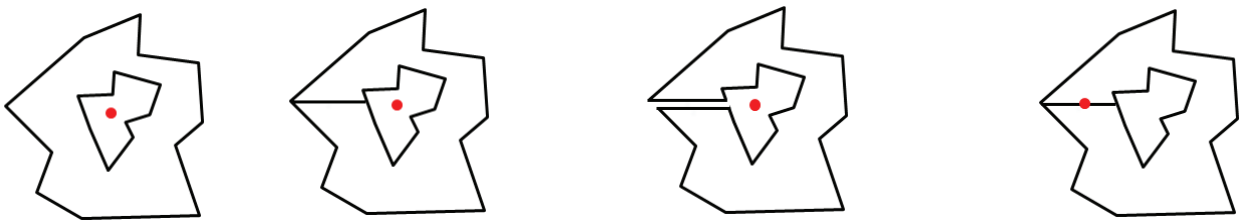


Abbildung 10: Problem durch die Repräsentationen von Enklaven

4 Performance

Nachdem im vorigen Abschnitt die grundsätzliche Funktionsweise der Prozedur vorgestellt wurde, soll nun die Performance von PROC GINSIDE bei Verarbeitung größerer Datenmengen analysiert werden. Es ist zu erwarten, dass die Laufzeit insbesondere von den nachfolgenden Faktoren abhängt:

1) Anzahl der zu testenden Punkte
2) Anzahl der in Frage kommenden Polygone
3) Anzahl der Eckpunkte, die die Polygone definieren
4) Quote der Punkte, die einem Polygon zugeordnet werden können
5) Verwendete SAS Version, Hardware, Serverauslastung

Die nachfolgende Tabelle 1 zeigt die Laufzeiten für ausgewählte Kombinationen der obigen Faktoren. Die Punkte 4) und 5) wurden in den Analysen mit einer Zuordnungsquote von 97% bis 99%, der SAS Version 9.4 und gleicher Hardware jeweils konstant gehalten. Durch wiederholte Berechnung und Bestimmung des Mittelwerts wurde zudem versucht unabhängig von der Serverauslastung zu werden:

[1] Hinweis: Zerfällt ein Gebiet in mehrere unverbundene Teilgebiete, wie bspw. das Bundesland Niedersachsen mit den ostfriesischen Inseln, so ist dies datentechnisch häufig über eine sog. SEGMENT Variable gelöst. Dabei hat jedes Einzelgebiet einen unterschiedlichen Wert in der SEGMENT Variable – allen Gebieten gemeinsam ist jedoch der KGS „03“ für Niedersachsen.

Tabelle 1: Laufzeiten bei unterschiedlichen Eingaben

	KGS2 Bundesland	KGS3 Reg. Bezirk	KGS5 Landkreis	KGS8 Gemeinde	KGS22 Wohnquartier
Polygonanzahl	16	38	403	11.368	85.213
Eckpunkte	286.000	468.000	1,268 Mio.	3,813 Mio.	10,886 Mio.
100.000 Punkte	55 sec	36 sec	14 sec	2 min	28 min
200.000 Punkte	5 min	3 min	68 sec	9 min	87 min
3 Mio. Punkte	29 min	18 min	7 min	56 min	~8 h

Man erkennt, dass die Laufzeit unabhängig von der Anzahl der in Frage kommenden Polygone mit der Anzahl der zu testenden Punkte stetig zunimmt. Betrachtet man die unterschiedlichen Granularitätsebenen, so stellt man fest, dass die Laufzeit zunächst von Bundesländern über Regierungsbezirke bis hin zu Landkreisen abnimmt. Ab der Ebene der Gemeinden steigt sie hingegen sprunghaft an. Dieser Effekt setzt sich auch auf der Ebene der Wohnquartiere fort. Würde man versuchen 3 Mio. Punkte direkt den 85.213 Wohnquartieren, die durch 10,886 Mio. Eckpunkte definiert sind, zuzuordnen, so kann eine Laufzeit von ca. 8h erwartet werden. Da der intern verwendete Algorithmus nicht veröffentlicht ist, kann über den sprunghaften Anstieg der Laufzeit ab der KGS8 Ebene lediglich spekuliert werden. Möglicherweise ist die Größe des RAM hierfür ursächlich.

Um die Laufzeit zu reduzieren, empfiehlt die SAS Dokumentation das Punkte Data Set und das Map Data Set möglichst *klein* zu halten, d. h. für die Berechnung unnötige Spalten aus den Data Sets zu entfernen und auch die Anzahl Zeilen möglichst gering zu halten. Dies bedeutet für das Punkte Data Set insbesondere, dass man keine Punkte doppelt testen sollte. Im Regelfall ist jedoch bereits fachlich sichergestellt, dass dieser Fall nur sehr selten eintritt, da Kunden, Patienten und Lieferanten meist unterschiedliche Adressen haben.

Das Map Data Set sollte in der Hinsicht klein gehalten werden, dass sich keine unnötigen Gebiete in diesem befinden. Ist man bspw. sicher, dass es sich bei den Punkten nur um Kunden aus Deutschland handelt, sollten sämtliche Geometrieinformationen, die bspw. die USA betreffen, eliminiert werden. Nicht gemeint ist dagegen die Einschränkung der Präzision der Kartendaten, die typischerweise über die Variable *Density*[2] gesteuert wird. Zwar ist es auch auf diesem Weg möglich die Anzahl der Eckpunkte deutlich zu reduzieren ohne dass optisch extreme Unterschiede zu erkennen sind (siehe Karten in Abbildung 11), es ergeben sich dadurch allerdings „Lücken“ an den Rändern des Kartenmaterials. Punkte, die gerade auf diese „Lücken“ entfallen, können dann fälschlicherweise nicht zugeordnet werden, sodass von dieser Vorgehensweise abgeraten werden muss.

[2] Die *Density* Variable kann SAS seitig mit der Prozedur PROC GREDUCE errechnet werden. Intern wird zur Ermittlung „wichtiger Punkte“ der Douglas Peucker Algorithmus verwendet.

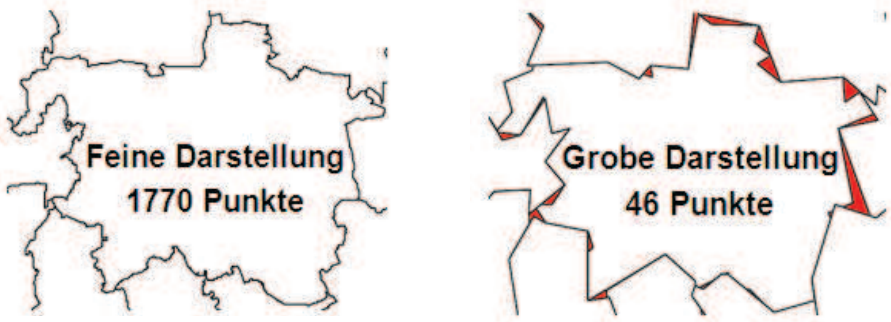


Abbildung 11: Unterschiedlich feine Repräsentationen der Gemeinde Hannover

Somit sind in einem möglichen Zielszenario alle die Zeit beeinflussenden Faktoren 1), 2) und 3) exogen vorgegeben, sofern eine Stichprobe der zu testenden Punkte nicht ausreichend ist. Es soll daher noch einmal speziell auf die Laufzeit in Abhängigkeit der Anzahl der zu testenden Punkte eingegangen werden:

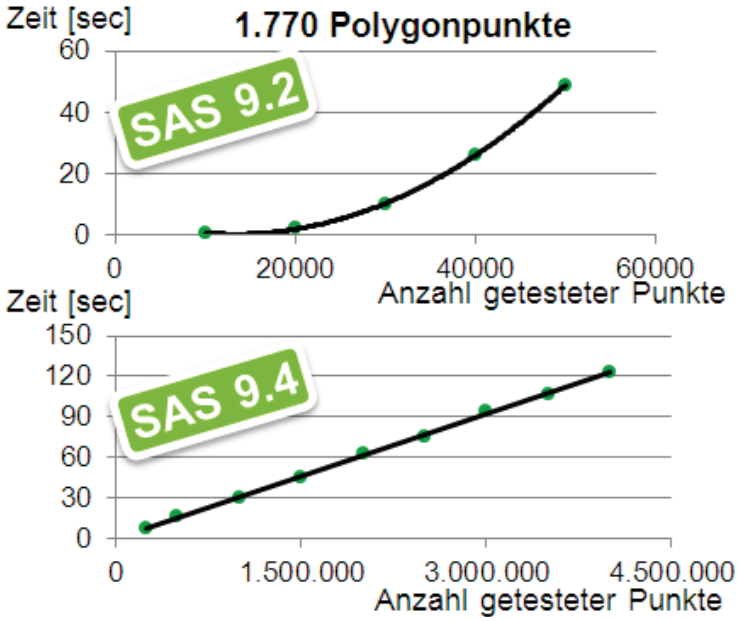


Abbildung 12: Laufzeit bei unterschiedlichen Anzahlen getesteter Punkt

Betrachtet man Abbildung 12 (Achtung: Unterschiedliche Skalierung der Achsen), so erkennt man mindestens eine quadratische Laufzeit in SAS 9.2, d. h. mit zunehmender Anzahl an zu testenden Punkten steigt die Laufzeit überproportional an. Es empfiehlt sich an dieser Stelle eine Divde & Conquer Strategie bei der die zu testenden Punkte nacheinander in kleineren „Häppchen“ (e.g. à 10.000 Punkte) der Prozedur zugeführt werden. In der Version SAS 9.4 ist dieser „Fehler“ behoben. Die Laufzeit steigt jetzt – wie mathematisch zu erwarten war (siehe Abschnitt 2) – linear, d. h. mit O(n) an.

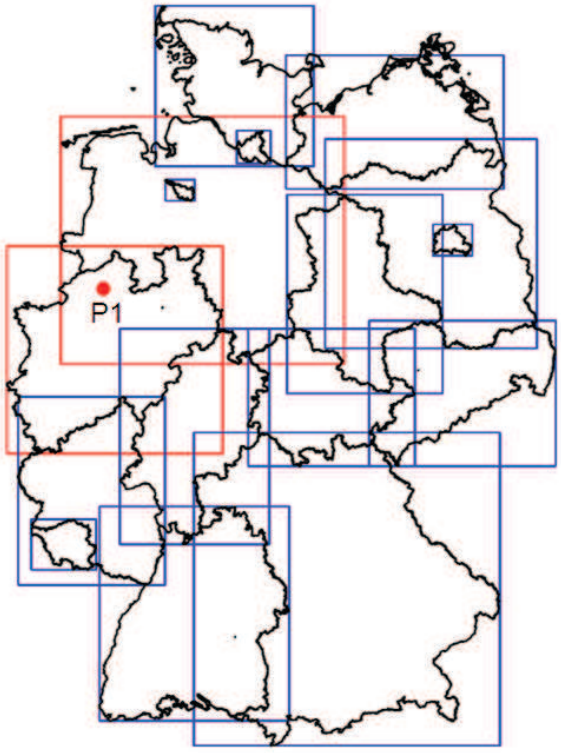
5 Optimierungsansätze

Nachfolgend wird aufgezeigt, wie die Performance der Zuordnung gesteigert, d. h. die Laufzeit stark verkürzt werden kann, ohne dabei Abstriche bezüglich der Genauigkeit machen zu müssen. Grundidee der beiden vorgestellten Ansätze ist es die Anzahl der notwendigen exakten Punkt-in-Polygon-Tests zu verringern, indem der „Suchraum" vorab verkleinert wird und der exakte Test damit überflüssig gemacht wird.

5.1 Einhüllende Rechtecke

Möchte man für einen Punkt feststellen, **in welchem Bundesland** (allgemein in welchem Polygon) er sich befindet, so kann man den Test beschleunigen, indem man vorab feststellt, welche Bundesländer für den Punkt generell in Frage kommen.

Dazu bestimmt man zunächst die die Bundesländer einhüllenden Rechtecke, d. h. jeweils das Minimum und Maximum der X- und Y-Koordinaten[3] aller Eckpunkte eines Bundeslands. Ob sich ein Punkt innerhalb oder außerhalb eines Rechtecks befindet ist im Vergleich zum exakten Punkt-in-Polygon-Test deutlich einfacher zu ermitteln, da dazu lediglich eine *if-Abfrage* notwendig ist. Anschließend nutzt man den Fakt aus, dass ein Punkt sich nur dann in einem Polygon befinden kann, wenn er sich auch innerhalb des einhüllenden Rechtecks des Polygons befindet. Andernfalls kann auf den exakten Test verzichtet werden, da der Punkt definitiv außerhalb liegen muss.



Im links abgebildeten Beispiel kann für den Punkt P1 mit l=7.6 und b=51.5 durch einfache *if-Abfragen* festgestellt werden, dass er sich entweder in Niedersachsen oder in NRW befinden muss und nicht im Saarland sein kann. Dazu werden lediglich 16 *if-Abfragen* der folgenden Form benötigt:

```
if l >=  6.65 and l <= 11.60 and
   b >= 51.29 and b <= 53.90 then nds=1;
if l >=  5.87 and l <=  9.46 and
   b >= 50.32 and l <= 52.53 then nrw=1;
if l >=  6.35 and l <=  7.41 and
   b >= 49.11 and l <= 49.64 then saar=1;
usw.
```

Abbildung 13: Funktionsweise der Rechteckmethode am Beispiel der 16 Bundesländer

[3] X- und Y-Koordinaten erhält man aus den Längen- und Breitengraden eines Punktes indem man bspw. die Gauss-Krüger-Transformation anwendet.

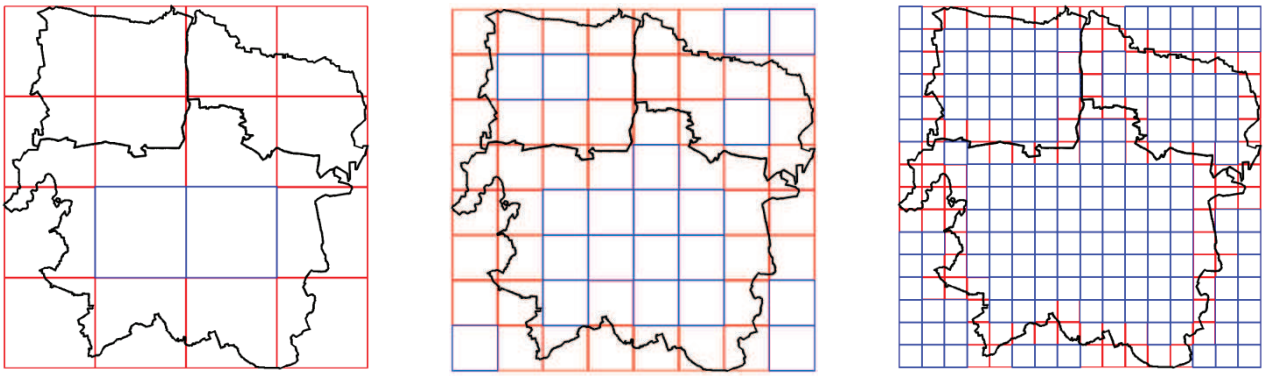
Anschließend führt man für jedes der 16 Bundesländer einmal die Prozedur PROC GINSIDE mit den jeweils in Frage kommenden Punkten aus und setzt die dabei Option <INSIDEONLY>. Fügt man die 16 Ergebnis Data Sets schließlich zusammen, erhält man dasselbe Gesamtergebnis, dass man bei einem „Gesamtaufruf" erhalten hätte – nur deutlich schneller. Die Laufzeit kann weiter optimiert werden, indem sukzessive auf in Frage kommende Bundesländer, Regierungsbezirke und Landkreise getestet wird. Wenn die Anzahl der Rechtecke hingegen sehr groß wird (e.g. 85.213 Gebiete auf der Ebene der Wohnquartiere), sinkt die Performance, da zu viele *if-Abfragen* notwendig sind. Insgesamt funktioniert die Methode natürlich insbesondere dann gut, wenn die Gebiete nahezu rechteckig sind, da dann die Überlappungen der Gebiete gering sind und somit für jeden Punkt nur wenige exakte Punkt-in-Polygon-Tests durchgeführt werden müssen.

5.2 Rasterung des Suchraums

Bei der Rasterung des Suchraums wird ein quadratisches (ggf. auch rechteckiges) Raster mit festgelegter Detailauflösung über die zu Grunde liegende Karte gelegt. Es wird anschließend für jedes Rasterquadrat ermittelt, ob es sich

1) vollständig innerhalb eines Polygons befindet
2) vollständig außerhalb eines Polygons befindet oder
3) von einer Polygonkante geschnitten wird.

Analog zur Methode der einhüllenden Rechtecke kann durch einfache *if-Abfragen* effizient festgestellt werden, ob sich ein Punkt innerhalb oder außerhalb eines Rasterquadrates befindet. Im Unterschied zur Methode der einhüllenden Rechtecke muss im Anschluss jedoch nur noch für Punkte, die unter 3) fallen der exakte Punkte-in-Polygon-Test durchgeführt werden. Für alle anderen Punkte kann der exakte Test entfallen, da das Ergebnis bereits eindeutig feststeht.



2 von 4*4 = 16
12,5% der Fläche eindeutig

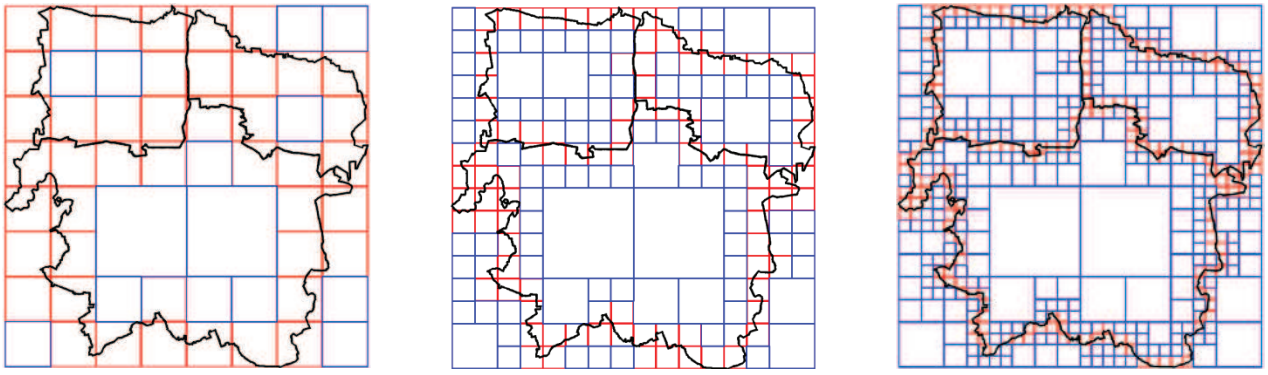
20 von 8*8 = 64
31,3% der Fläche eindeutig

157 von 16*16 = 256
61,3% der Fläche eindeutig

Abbildung 14: Funktionsweise der Rastermethode am Beispiel der Gemeinden Hannover, Isernhagen, Langenhagen mit unterschiedlicher Rastergranularität

Auch für diese Methode muss ein Trade-Off in Bezug auf die Detailauflösung gefunden werden: Wählt man das Raster zu grob, kann nur für wenige Punkte die finale Entscheidung gemäß 1) oder 2) getroffen werden. Es muss dann weiterhin für einen Großteil der Punkte der exakte Punkt-in-Polygon-Test durchgeführt werden. Im Beispiel links in Abbildung 14 fallen bspw. nur zwei der 4*4 = 16 Rasterquadrate eindeutig in die Gemeinde Hannover. Wählt man das Raster hingegen sehr fein, so fallen zwar nur wenige Punkte unter den Fall 3) – es werden dann jedoch auch sehr viele *if-Abfragen* benötigt, die die Performance schmälern (siehe Beispiel rechts in Abbildung 14, 16*16 = 256 Rasterquadrate).

Man kann den Ansatz der Rasterung weiter optimieren, indem man mit einem vergleichsweise groben Raster startet und anschließend jeweils nur jene Rasterquadrate weiter verfeinert, die unter den Fall 3) fallen.[4] Somit steigt die Anzahl der *if-Abfragen* nicht mehr quadratisch mit der Verfeinerung der Rastergröße an, sondern es wird nur für die Rasterquadrate ein feiner auflösendes Raster gewählt, für die ansonsten keine eindeutige Entscheidung getroffen werden könnte. Das Raster für die Gemeinden Hannover, Isernhagen und Langenhagen würde dann wie folgt aussehen:



14 von 58 [64]
31,3% der Fläche eindeutig

91 von 190 [256]
61,3% der Fläche eindeutig

260 von 487 [1024]
77,8% der Fläche eindeutig

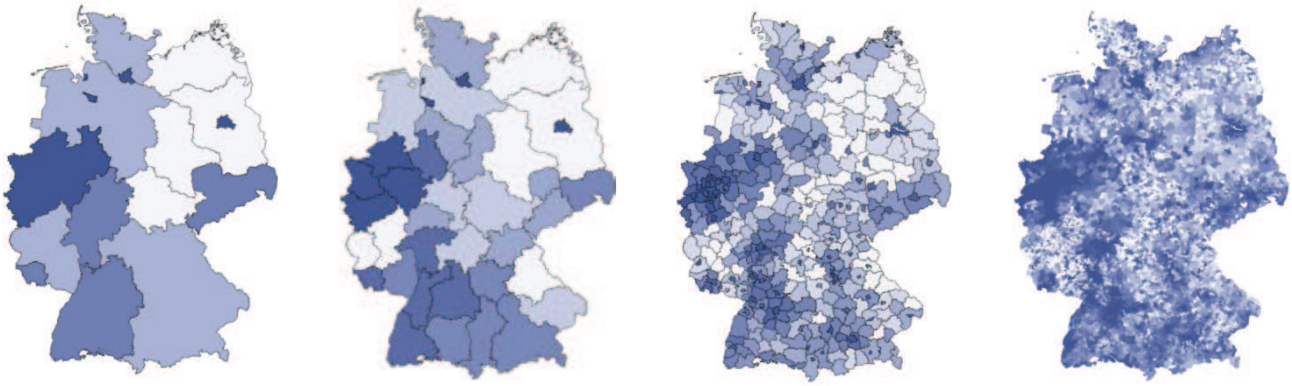
Abbildung 15: Funktionsweise der hierarchischen Rastermethode

Die Rastermethode ist im Vergleich zur Rechteckmethode deutlich komplexer in der Verarbeitung, da das Raster zunächst aufgebaut und ggf. intelligent verfeinert werden muss. Für jedes Rasterquadrat muss zudem einmalig ermittelt werden, wie es in Relation zu den ursprünglichen Gebieten gelegen ist, d. h. ob es sich innerhalb oder außerhalb aller Gebiete befindet oder ob es von einer der Kanten „geschnitten“ wird. Der Aufbau des Rasters lohnt sich daher insbesondere dann, wenn man sehr häufig sehr viele Punkte testen muss.

[4] Dieser Ansatz wird auch in einem Paper von Troy Martin Hughes verfolgt: “Winning the War on Terror with Waffles: Maximizing GINSIDE Efficiency for Blue Force Tracking Big Data”[3]

6 Fazit

SAS bietet mit der Prozedur PROC GINSIDE die Möglichkeit einen Punkt-in-Polygon-Test durchzuführen und unterstützt den Anwender damit bei geografischen Analysen. Eingeführt in der Version SAS 9.2 hat die Prozedur inzwischen einige zusätzliche Parameter wie bspw. <INCLUDEBORDER> erhalten. Hervorzuheben ist die große Performance Verbesserung in Bezug auf die Anzahl der zu testenden Punkte, die sich zwischen SAS 9.2 (mindestens quadratische Laufzeit) und SAS 9.4 (lineare Laufzeit) ergibt. Möchte man sehr viele Punkte testen und ist die Anzahl der in Frage kommenden Gebiete und deren Eckpunkte sehr groß, empfiehlt es sich einige Vorab-Optimierungen vorzunehmen. Die beteiligten Data Sets sollten daher zunächst möglichst klein gehalten werden, d. h. möglichst wenige Spalten und Zeilen enthalten, ohne dabei jedoch auf Details in den Kartendaten zu verzichten. Zusätzlich sollte man die Komplexität des Ausgangsproblems verringern, indem vorab entweder mit der Methode der einhüllenden Rechtecke oder durch Rasterung eine Vielzahl von exakten Punkt-in-Polygon-Tests vermieden werden kann. Generell empfiehlt sich eine Divide & Conquer Strategie, d. h. man sollte nicht versuchen „alle Punkte auf einmal“ zuzuordnen, sondern sukzessive, d. h. pro Bundesland oder Landkreis vorzugehen. Berücksichtigt man diese Hinweise, ist man in der Lage auch eine Vielzahl an Punkten noch effizient einer Vielzahl an Polygonen zuzuordnen. Abschließend dargestellt ist die Zuordnung der ca. 18 Mio. Gebäude in Deutschland, die kostenfrei aus OpenStreetMap [4] extrahiert werden können, zu den unterschiedlichen Granularitätsebenen der Kreisgemeindestruktur. Nach der Zuordnung wurde jeweils die absolute Gebäudehäufigkeit pro Gebiete gezählt und die Gebiete umso dunkler eingefärbt, je höher die Anzahl war (siehe Abbildung 16). Die Karten spiegeln somit in etwa die Bevölkerungsdichte in den jeweiligen Regionen wider. Während die Zuordnung in „einem Schritt“ mit der vorhandenen technischen Infrastruktur nicht möglich war, ließ sich die Zuordnung mit den oben genannten Optimierungen vergleichsweise schnell vornehmen. Im Extremfall wurden dabei ca. 18 Mio. Punkte (Gebäude) insgesamt 85.213 Polygonen (Wohnquartieren), die durch 10,886 Mio. Eckpunkte definiert waren, zugeordnet (siehe Abbildung 17).



16 Bundesländer
286.000 Eckpunkte

38 Regierungsbezirke
468.000 Eckpunkte

404 Landkreise
1,268 Mio. Eckpunkte

11.366 Gemeinden
3,813 Mio. Eckpunkte

Abbildung 16: Gebäudedichte auf unterschiedlichen Ebenen der Kreisgemeindestruktur in Deutschland (Gebäudeinformationen aus OpenStreetMap [4], Stand 12/2014)

85.213 Wohnquartiere, 10,886 Mio. Eckpunkte

Abbildung 17: Gebäudedichte auf Wohnquartierebene in Deutschland

Literatur

[1] Nikolaos Sitaridis, Gisela Büchele, Jon Genuneit: Geokodierungen mit HTTP-Anforderungen (SAS-Makro Geocode). In U. Rendtel, P. Schirmbacher, O. Kao, W. Lesener, R. Minkenberg (Hrsg.): KSFE 2010 – Proceedings der 14. KSFE. Shaker Verlag, Aachen, 279-284

[2] Dennis Cosfeld, Jens Blecking: Geokodierung mit SAS als Tool des Versicherungsmarketings. In C. Ortseifen, H. Ramroth, M. Weires, R. Minkenberg (Hrsg.): Ksfe 2011 – Voneinander lernen. Proceedings der 15. KSFE. Shaker Verlag, Aachen, 91-96

[3] Troy Martin Hughes: Winning the War on Terror with Waffles: Maximizing GINSIDE Efficiency for Blue Force Tracking Big Data. SAS Analytics 2013, Paper PO-18, *analytics.ncsu.edu/sesug/2013/PO-18.pdf*

[4] Geodaten aus OpenStreetMap, veröffentlicht unter OdbL