1:N Matching von Fällen und Kontrollen: Propensity Score vs. PROC SQL

Andreas Deckert
Institute of Public Health
INF 324
69120 Heidelberg
a.deckert@uni-heidelberg.de

Zusammenfassung

Mit Hilfe des so genannten Matching versucht man in Beobachtungsstudien, bei denen keine randomisierte Gruppenzuteilung möglich ist (z.B. retrospektive Fall-Kontrollstudien in der Epidemiologie), Verzerrungen des Ergebnisses durch unterschiedliche Altersstrukturen und/oder andere Faktoren in der Fall- und in der Kontrollgruppe zu vermeiden. Dazu werden die Kontrollen entsprechend der Struktur der Fälle ausgewählt. Eine Möglichkeit Matching in SAS zu realisieren bietet die Prozedur PROC SQL gefolgt von einer Nachbereitung des Ergebnisses. Die Art der Nachbereitung bestimmt dabei letztendlich die Trefferquoten. Eine weitere häufig verwendete Methode ist die Anwendung von sogenannten Propensity Scores zur Identifizierung von Fällen und Kontrollen mit ähnlichen Strukturen. Zwei verschiedene Formen der Nachbearbeitung von PROC SQL sowie Propensity Scores werden hier an einem konkreten Beispiel näher auf ihre Tauglichkeit hin untersucht und miteinander verglichen. Für PROC SQL wird des Weiteren eine weiterentwickelte Methode einschließlich Makro vorgestellt, welche die ursprüngliche Trefferquote erheblich steigert.

Schlüsselwörter: Fall-Kontrollstudie, Matching, PROC SQL, Propensity Score, PROC POWER

1 Einleitung

Die zufällige Zuteilung von Probanden zu verschiedenen Gruppen eines prospektiven experimentellen Studiendesigns ist die Voraussetzung dafür, dass das Ergebnis allein auf die absichtlich erzeugten Unterschiede in wenigen kontrollierten Faktoren zwischen den Gruppen zurückgeführt werden kann. Ohne Randomisierung könnten die Unterschiede auch durch weitere an die kontrollierten Faktoren gekoppelte ungleich verteilte Faktoren verursacht werden, was zu einer Schein-Assoziation zwischen kontrollierten Faktoren und Ergebnis führen kann (Confounding).

Die Durchführung von experimentellen Studien ist jedoch aus ethischen oder auch aus rein praktikablen Gründen oft nicht möglich. Das betrifft z.B. Studien in der Epidemiologie, bei denen es darum geht, Risikofaktoren für bestimmte Erkrankungen aufzudecken. Dazu wird in Fall-Kontrollstudien einer Fallgruppe von erkrankten Personen eine gesunde Kontrollgruppe gegenübergestellt und es werden meistens retrospektiv Unter-

schiede in der Verteilung von Risikofaktoren zwischen diesen beiden Gruppen untersucht.

Bei diesen Studiendesigns kann Confounding auf die Assoziation von Risikofaktor und Ergebnis (Outcome) eine große Rolle spielen. Bei bekannten oder vermuteten zusätzlichen Einflussfaktoren wie z.B. dem Alter versucht man daher, deren Effekte durch eine geschickte Wahl von Fällen und Kontrollen so zu minimieren, dass nur noch die unabhängigen Effekte auf das Ergebnis wirken können.

1.1 Matching

Eine der Methoden, um Confounding-Effekte in Fall-Kontrollstudien einzudämmen, ist das so genannte Matching. Dabei werden die Kontrollen gezielt derart ausgewählt, dass hinsichtlich bestimmter Faktoren Strukturgleichheit in Fällen und Kontrollen vorliegt. Beim Häufigkeitsmatching wird dies erreicht, indem innerhalb von Merkmalsstrata Kontrollen zufällig so gezogen werden, dass im Ergebnis die Häufigkeiten von Personen mit bestimmten Merkmalsausprägungen in der Fall- und Kontrollgruppe gleich groß sind. Beim individuellen Matching wird dagegen zu jedem Fall direkt eine passende Kontrolle gesucht, deren Merkmale (oft innerhalb eines gewissen Toleranzbereiches) mit denen des Falls übereinstimmen. Hier wird im Folgenden das individuelle Matching behandelt.

1.2 Individuelles 1:N Matching

Um die Wahrscheinlichkeit (Power) zu erhöhen, einen Unterschied im Vorhandensein von Risikofaktoren zwischen den Gruppen zu finden, kann man den Umfang der Kontrollgruppe gegenüber der Fallgruppe vergrößern. Statt genau einer Kontrolle pro Fall werden nun zu jedem Fall mehrere Kontrollen ausgewählt. Mit einem einfachen Beispiel kann man illustrieren, wie sich mit steigendem Verhältnisfaktor *N* die Power erhöht: Angenommen die Prävalenz eines Risikofaktors in der Kontrollgruppe sei 6% und die Größe der Fallgruppe sei auf 300 Personen begrenzt. Wie groß muss das Verhältnis von Kontrollen zu Fällen mindestens sein, damit man einen real vorhandenen 5%igen Unterschied in der Prävalenz zwischen Fall- und Kontrollgruppe (OR[1]=2) mit einer Wahrscheinlichkeit von 80% (Power) tatsächlich entdecken kann? Bei Annahme einer χ^2-Verteilung kann hier mit der Prozedur `PROC POWER` die Power für jede Relation zwischen Fällen und Kontrollen berechnet werden:

```
PROC POWER;
     TWOSAMPLEFREQ
     TEST = pchi
     ALPHA = .05
     ODDSRATIO = 2
     REFPROPORTION = 0.06
     GROUPWEIGHTS = (N 1)
```

[1] Odds Ratio

```
      NTOTAL = 300+300*N
      POWER = .;
RUN;
```

Erhöht man innerhalb von `PROC POWER` das Verhältnis von Kontrollen zu Fällen mit `GROUPWEIGHTS = (N 1)`, erhält man steigende Wahrscheinlichkeiten für die Power. Durch Einbinden von `PROC POWER` in ein Makro lässt sich die Power-Funktion für steigende *n* leicht darstellen (s. Abb. 1).

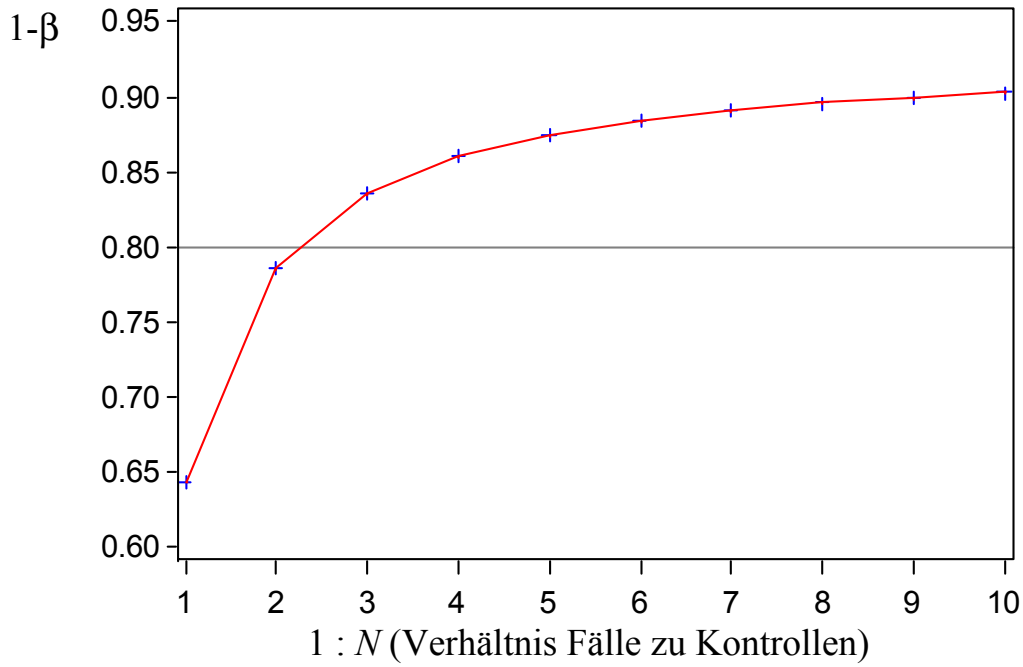


Abbildung 1: Powersimulation für p_K=0.06, OR=2, α=0.05, 300 Fälle

Die oben geforderte Power von 80% wird in dem Beispiel also bei einem Größenverhältnis zwischen Fall- und Kontrollgruppe von 1:3 erreicht.

Die Powersimulation verdeutlicht, dass eine Erhöhung des Verhältnisses von Fällen zu Kontrollen von 1:1 auf 1:2 den größten Effekt erzielt. Jede weitere Erhöhung verringert den Zugewinn an Power. Bei großen *N* steht dem Zugewinn an Power ein ungleich stärker steigender Aufwand für Rekrutierung, Logistik und steigende Kosten für die Untersuchungen gegenüber, weshalb *N* meist zwischen 2 und 4 gewählt wird.

2 Problemstellung

Bei einer epidemiologischen Studie zum Gesundheitsstatus von Migranten sollten gesunde Personen aus einer Migranten-Datenbank als Kontrollen für ein Interview eingeladen werden. Mittels Matching sollte sichergestellt werden, dass bestimmte Merkmale der Kontrollgruppe ähnlich der Fallgruppe sind. In der Studie war die Auswahl der Kontrollen zusätzlich auf eine bestimmte Region begrenzt, was den möglichen Kontrollpool stark einschränkte. Ein erster Versuch das Matching mit einem in der Literatur

beschriebenen Verfahren (siehe [1]) mittels `PROC SQL` durchzuführen (Methode 1), führte zu einer unbefriedigenden Anzahl von identifizierten Fall-Kontroll-Paaren. Die Trefferquote konnte jedoch mit einer weiterentwickelten Nachbereitung deutlich verbessert werden (Methode 2). Die Anwendung der ebenfalls in der Literatur beschriebenen Methode der Propensity Scores (Methode 3; siehe [2]) brachte ähnliche Ergebnisse wie Methode 1. Im Folgenden werden zwei unterschiedliche Szenarien einer Fall-Kontrollstudie simuliert und dann in Kapitel 3 die drei genannten Methoden auf die simulierten Daten angewendet und deren Ergebnisse diskutiert.

2.1 Ausgangssituation

Angenommen es handelt sich um eine epidemiologische Fall-Kontrollstudie zur Erforschung von Risikofaktoren in einer Gruppe zugezogener Migranten. Von Interesse sind Unterschiede im Risikoprofil von erkrankten im Vergleich zu gesunden Migranten. Die Variablen Geschlecht, Alter und Zuzugsdatum stehen im Verdacht, sowohl mit dem Auftreten der Krankheit als auch mit den Expositionen in Verbindung zu stehen. Es soll daher ein 1:2-Matching von Fällen und Kontrollen nach Alter, Geschlecht und Zuzugsdatum durchgeführt werden. 300 Fällen steht ein begrenzter Pool mit 900 Personen gegenüber, die als Kontrollen in Frage kommen. Als relevanter Zuzugszeitraum gelten die Jahre von 1990 bis 2010. Da es sich um einen begrenzten Kontrollpool handelt, dürfen Fälle und Kontrollen in Alter und Zuzugsdatum jeweils um ±3 Jahre abweichen.

2.2 Szenariensimulation

Der Erfolg des Matching soll an zwei Studienszenarien getestet werden.

Szenario I:

- Gleiche Altersverteilung in der Fallgruppe und im Pool der möglichen Kontrollen
- Frauenanteil in beiden Gruppen jeweils 50%.

Szenario II:

- Linksschiefe Altersverteilung im Pool der möglichen Kontrollen
- Frauenanteil: Fallgruppe 30%, Gruppe der möglichen Kontrollen 50%

Die Verteilung der Zuzugsdaten soll in beiden Szenarien gleich sein. Die Altersverteilungen wurden in SAS mit einer Gompertz-Verteilung entsprechend folgender Formel simuliert (siehe auch [3]):

$$ALTER = \frac{c}{\alpha} \cdot LOG\left(\frac{1 - \alpha \cdot LOG(U)}{\lambda \cdot \exp(\beta \cdot X)} \right)$$

Dabei ist *U* eine gleichverteilte Zufallsvariable und X die binäre Variable für die Gruppenzugehörigkeit. Die Wahl der anderen Variablen beeinflusst das Aussehen der Gompertz-Verteilung. Abbildung 2 (s. folgende Seite) zeigt die simulierten Altersverteilungen in den verschiedenen Szenarien.

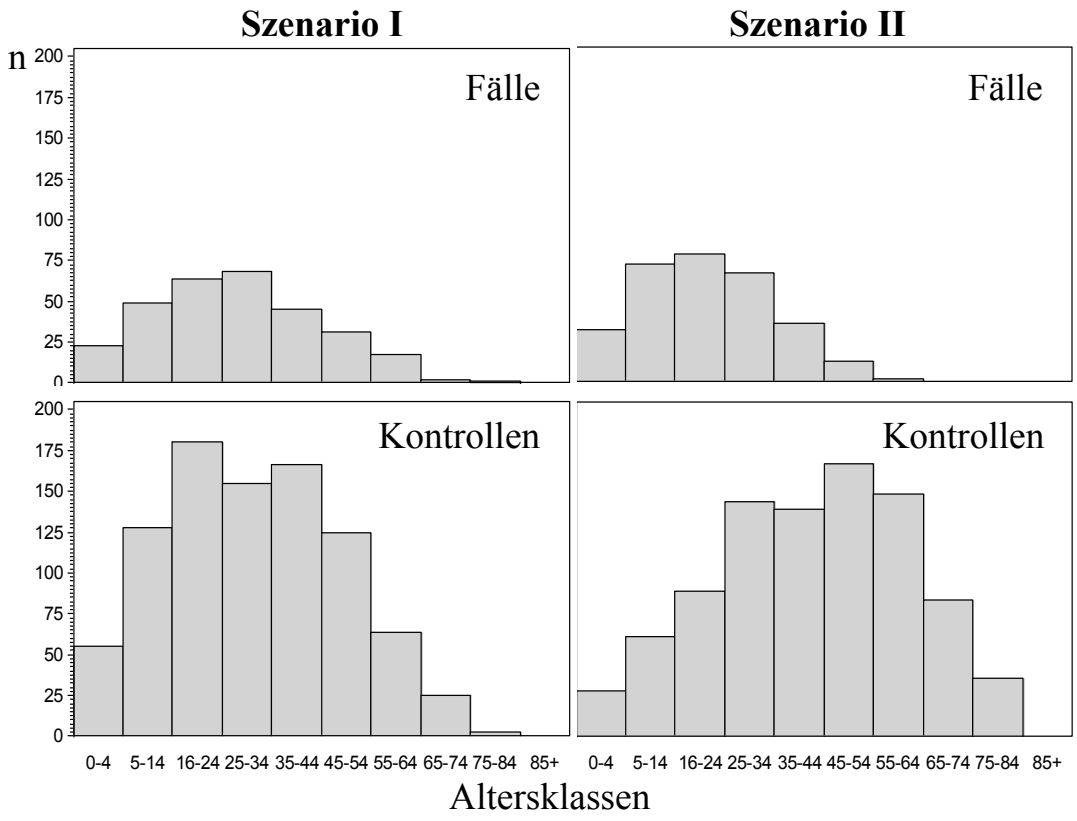


Abbildung 2: Simulation der Altersverteilungen in Szenario I und Szenario II

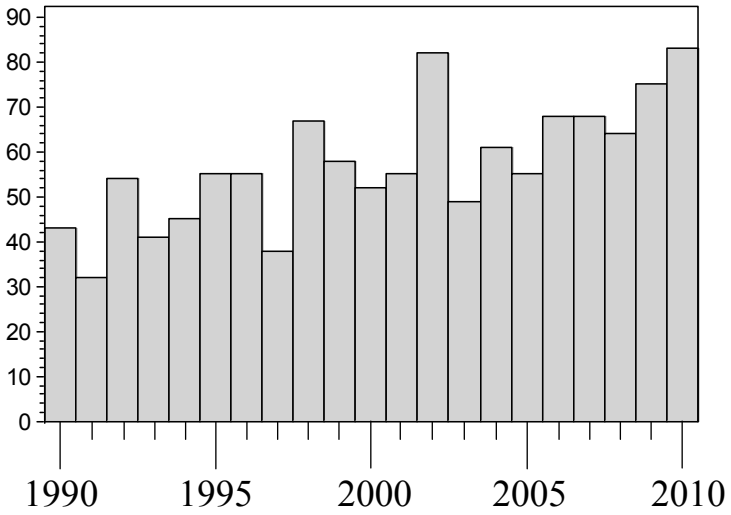


Abbildung 3: Simulation der Zuzugsdaten (hier Szenario I)

Für die Zuzugsdaten (s. Abbildung 3 oben) wurde zunächst eine Gleichverteilung für den Zeitraum von 1990 bis 2010 angenommen. Die simulierten Zuzugsdaten mussten allerdings nachträglich unter Berücksichtigung der Reihenfolge von Geburtsdatum und Zuzugsdatum in einigen Fällen korrigiert werden: Falls das Zuzugsdatum vor dem Geburtsdatum lag, wurde als neues Zuzugsdatum das Geburtsdatum plus die Differenz zwischen Geburtsdatum und altem Zuzugsdatum gewählt (mit Begrenzung auf 2010).

Damit ergibt sich z.B. für Szenario I eine leicht steigende Zunahme der Zuzüge über den gesamten Zeitraum[2], was durchaus einer realen Situation entsprechen kann.

Da es oftmals zu einem Fall mehrere passende Kontrollen gibt und umgekehrt gleichzeitig zu manchen Fällen keine, nur eine oder genau zwei passende Kontrollen existieren, gibt es keinen Algorithmus, der ohne die Berücksichtigung aller möglichen Kombinationen für jede Studiensituation und jede Verteilung der Matching-Variablen die optimale Fall-Kontroll-Zuweisung finden kann. Eine ideale Lösung (die Berechnung aller möglichen Kombinationen mit nachträglicher Auswahl der Kombination mit den meisten korrekten Zuweisungen) ist sowohl hinsichtlich des Programmieraufwandes als auch hinsichtlich der benötigten Rechenleistung sehr aufwendig. Daher versucht man mit einfacheren Methoden annähernd optimale Lösungen zu finden. Zwei dieser Methoden werden hier auf ihre Tauglichkeit hinsichtlich der oben beschriebenen simulierten Szenarien untersucht und verglichen.

3 Matching mit `PROC SQL` und Propensity Scores

Es existieren einige Ansätze, die versuchen, ein individuelles Matching in SAS mit Hilfe von Datasteps, Sortieralgorithmen und Häufigkeitstabellen zu lösen. Alle diese Ansätze erfordern einen hohen Programmieraufwand und sind oft nicht direkt auf beliebige Situationen übertragbar. Ein Matching mit PROC SQL erscheint dagegen auf Anhieb sinnvoll, da hierbei zunächst alle möglichen Kombinationen von Fällen und Kontrollen (ohne Beschränkung auf die einmalige Verwendung einer Kontrolle) in einer Tabelle erstellt werden ("Many-to-many match") und man dann nachträglich aus dieser Vielzahl „nur noch“ die Fall-Kontroll-Paare so aussuchen muss, dass möglichst vielen Fällen auch Kontrollen zugewiesen werden können.

Eine weitere Möglichkeit bietet sich mit der Berechnung von Propensity Scores. Dabei wird für jede Person die Vorhersage-Wahrscheinlichkeit berechnet, aufgrund der individuellen Matching-Variablen-Struktur ein Fall zu werden. Dazu wird ein logistisches Regressionsmodell erstellt mit der binären Variable Fallzugehörigkeit als Outcome und den Matching-Variablen als Einflussgrößen. Kontrollen mit ähnlicher Struktur erhalten dadurch ähnliche Vorhersage-Wahrscheinlichkeiten wie vergleichbare Fälle. Anhand der Wahrscheinlichkeiten kann dann nachträglich eine Fall-Kontroll-Zuordnung mit Hilfe verschiedener Algorithmen vorgenommen werden.

3.1 Anwendung von `PROC SQL`

Die folgende Vorgehensweise (Methode 1) ist in einem Artikel von Kawabata et.al. beschrieben [1]. Die Datensätze zu den Fällen und Kontrollen enthalten jeweils eine eindeutige ID sowie Variablen zu Alter, Geschlecht und Zuzugsdatum. Zur Vorbereitung

[2] Für Szenario II fällt die Steigung aufgrund der linksschiefen Altersverteilung der Kontrollen geringer aus.

des Matching muss zunächst der Toleranzbereich für das Alter (hier in Form des Geburtsjahres) und das Zuzugsdatum in der Kontrollgruppe generiert werden.
Zuzugsjahr = year(Zuzugsdatum);

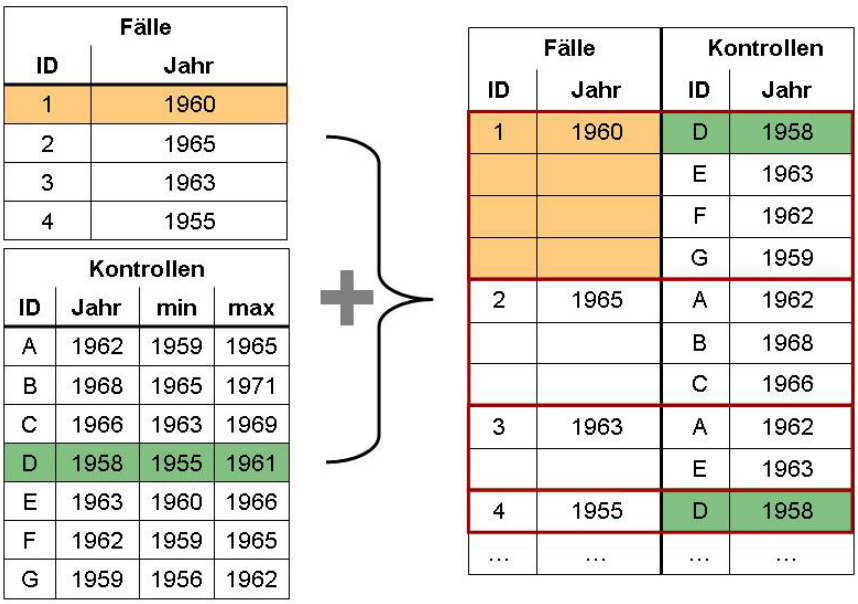
```
DATA Kontrollgruppe; SET Kontrollgruppe;
    Geburtsjahr = year(Geburtsdatum);
    min_Geburtsjahr = Geburtsjahr - 3;
    max_Geburtsjahr = Geburtsjahr + 3;
    min_Zuzugsjahr = Zuzugsjahr - 3;
    max_Zuzugsjahr = Zuzugsjahr + 3;
RUN;
DATA Fallgruppe; SET Fallgruppe;
    Geburtsjahr = year(Geburtsdatum);
    Zuzugsjahr = year(Zuzugsdatum);
RUN;
```

Danach kann dann die Verknüpfung von Fällen und Kontrollen mit `PROC SQL` erfolgen:

```
PROC SQL;
     CREATE TABLE Abgleich AS SELECT
         A.ID AS Fall_ID, B.ID AS Kontroll_ID,
         A.Geburtsjahr AS Fall_Gebjahr,
         B.Geburtsjahr AS Kontrolle_Gebjahr,
         A.Zuzugsjahr AS Fall_Zuzug,
         B.Zuzugsjahr AS Kontrolle_Zuzug,
         A.Geschlecht AS Fall_Geschlecht,
         B.Geschlecht AS Kontrolle_Geschlecht,
         FROM Fallgruppe A, Kontrollgruppe B
         WHERE ((A.Geburtsjahr between
                 B.min_Geburtsjahr AND B.max_Geburtsjahr)
             AND (A.Zuzugsjahr between
                 B.min_Zuzugsjahr AND B.max_Zuzugsjahr)
             AND A.Geschlecht = B.Geschlecht);
QUIT;
```

Hier wird mit `PROC SQL` eine neue Tabelle „Abgleich“ erstellt und die Matching-Variablen werden so umbenannt, dass sie sich für Fälle und Kontrollen unterscheiden. Der Vorteil dieser Prozedur verbirgt sich in der Where-Anweisung, die die Kontrollen entsprechend den Matching-Bedingungen mit den Fällen verknüpft. Die Ergebnistabelle enthält alle möglichen Fall-Kontroll-Kombinationen (s. Abbildung 4).

Würde man nun aus der entstandenen Tabelle jeweils einfach nach Fällen sortieren und z.B. die ersten beiden Kontrollen zu jedem Fall als Treffer auswählen, dann ist es sehr wahrscheinlich, dass man zu einigen Fällen keine passenden Kontrollen findet, da diese schon vorher anderen Fällen zugewiesen wurden.



Fälle	
ID	Jahr
1	1960
2	1965
3	1963
4	1955

Kontrollen			
ID	Jahr	min	max
A	1962	1959	1965
B	1968	1965	1971
C	1966	1963	1969
D	1958	1955	1961
E	1963	1960	1966
F	1962	1959	1965
G	1959	1956	1962

Fälle		Kontrollen	
ID	Jahr	ID	Jahr
1	1960	D	1958
		E	1963
		F	1962
		G	1959
2	1965	A	1962
		B	1968
		C	1966
3	1963	A	1962
		E	1963
4	1955	D	1958
…	…	…	…

Abbildung 4: Kombination von Fällen und Kontrollen durch `PROC SQL`

Um den Fällen mit nur wenigen passenden Kontrollen den Vorzug zu geben, werden also zunächst die passenden Kontrollen pro Fall gezählt.

```
PROC SORT DATA = Abgleich; BY Fall_ID; RUN;
DATA Abgleich_2 (keep = Fall_ID Anzahl_K); SET Abgleich;
     BY Fall_ID;
     RETAIN Anzahl_K;
     IF first.Fall_ID THEN Anzahl_K = 1;
          ELSE Anzahl_K + 1;
     IF last.Fall_ID THEN OUTPUT;
RUN;
```

Die Anzahl der Kontrollen wird in die Ursprungstabelle übertragen und gleichzeitig eine Zufallszahl generiert. Ab hier wird der Prozess später in mehreren Durchläufen mit neuen Zufallszahlen neu gestartet (s. unten), wozu ***seed*** jedes Mal variiert wird. Dann werden die Fall-Kontroll-Paare nach Kontrollen und innerhalb von gleichen Kontrollen nach der Anzahl der Kontrollen pro Fall und innerhalb dieser nach den generierten Zufallszahlen sortiert.

```
DATA Abgleich_3; MERGE Abgleich Abgleich_2;   ✸
     BY Fall_ID;
     z_zahl=uniform(seed);
RUN;
PROC SORT DATA = Abgleich_3; BY Kontroll_ID Anzahl_K z_zahl; RUN;
```

Als nächstes wird die jeweils erste Kontrolle ausgewählt, die anderen Fall-Kontroll-Paarungen werden verworfen.

```
DATA Abgleich_4; SET Abgleich_3;
     BY Kontroll_ID;
     IF first.Kontroll_ID;
RUN;
```

Nun kann es Fälle geben, für die mehr als zwei verschiedene Kontrollen übrig geblieben sind. Also wird im nächsten Schritt noch nach den Fällen und den Zufallszahlen sortiert[3] und die ersten beiden Fall-Kontroll-Paare als Ergebnismenge ausgewählt.

```
PROC SORT DATA = Abgleich_4; BY Fall_ID z_zahl; RUN;
DATA Final Unvollstaendig; SET Abgleich_4;
     BY Fall_ID;
     RETAIN num;
     IF first.Fall_ID THEN num = 1;
     IF num le 2 THEN DO; /*1:2 Matching*/
          OUTPUT Final;
          num + 1;
     END;
     IF last.Fall_ID THEN DO;
          IF num le 2 THEN OUTPUT Unvollstaendig;
     END;
RUN;
```

Die Prozedur wird nun ab ✸ mit neuen Zufallszahlen mehrmals wiederholt[4] und dann letztendlich diejenige „Final"-Tabelle mit den meisten gefundenen Fall-Kontroll-Paaren ausgewählt. Dabei werden hier nur die Fälle mit jeweils 2 Kontrollen berücksichtigt, einfache Fall-Kontrollzuordnungen entfallen.

Wendet man die Prozedur in der beschriebenen Weise auf die 2 simulierten Szenarien an, können in *Szenario I* (gleiche Altersverteilung) 146 von 300 Fällen wie gewünscht zwei Kontrollen zugeordnet werden. Zu 23 Fällen findet sich jeweils noch eine Kontrolle. Für *Szenario II* (ungleiche Altersverteilung) können nur für 123 Fälle jeweils zwei passende Kontrollen gefunden werden und für 30 Fälle jeweils eine Kontrolle.

Diese schlechten Ergebnisse überraschen auf den ersten Blick, vor allem in Bezug auf *Szenario I*. Bei näherer Betrachtung lässt sich feststellen, dass dieser Algorithmus in bestimmten Situationen zu viele mögliche Fall-Kontroll-Paarungen verwirft und zu grob aussortiert. Die schlechten Trefferraten scheinen vor allem beim Abgleich mit Toleranzbereichen aufzutreten, da dann nach dem Sortieren selbst die Fälle mit wenigen Kontrollen am Anfang der Liste eine relativ hohe Anzahl möglicher Kontrollen aufwei-

[3] Da die Zufallszahlen bei jedem Durchlauf neu gesetzt werden, stehen hier jedes Mal andere Fälle an erster Stelle wodurch sich andere Paarungen ergeben und dadurch auch Schwankungen in der Anzahl der Treffer.

[4] Für 100 Wiederholungen wurden ca. 2 Minuten Rechenzeit benötigt. Bei 500 Wiederholungen lag die Rechenzeit bei ca. 5 Minuten (durchschnittlicher Arbeitsplatzrechner; Intel ® Core™ 2 Duo CPU, 2.4 GHz, 2 GB RAM). Mit 500 Wiederholungen wurde aber auch eine finale Tabelle mit mehr Zuordnungen gefunden als mit 100 Wiederholungen.

sen. In Abbildung 5 wird dies an einem Beispiel illustriert: Hier liegt eine Situation vor, bei der an den Rändern des Toleranzbereiches zu Fall 1 weitere Fälle liegen, zu denen die gleichen Kontrollen wie zu Fall 1 passen. Da aber Fall 1 jeweils die niedrigsten Zufallszahlen und zudem insgesamt weniger Kontrollen im Vergleich zu den meisten anderen Fällen hat, wird Fall 1 durch den Algorithmus immer an die erste Stelle innerhalb der gleichen Kontrollen gesetzt[5] (außer bei Kontrolle D). Das führt dann beim Auswählen der jeweils ersten Kontrolle und im weiteren Verlauf dazu, dass hier die Hälfte aller möglichen korrekten Paarungen unterschlagen wird.

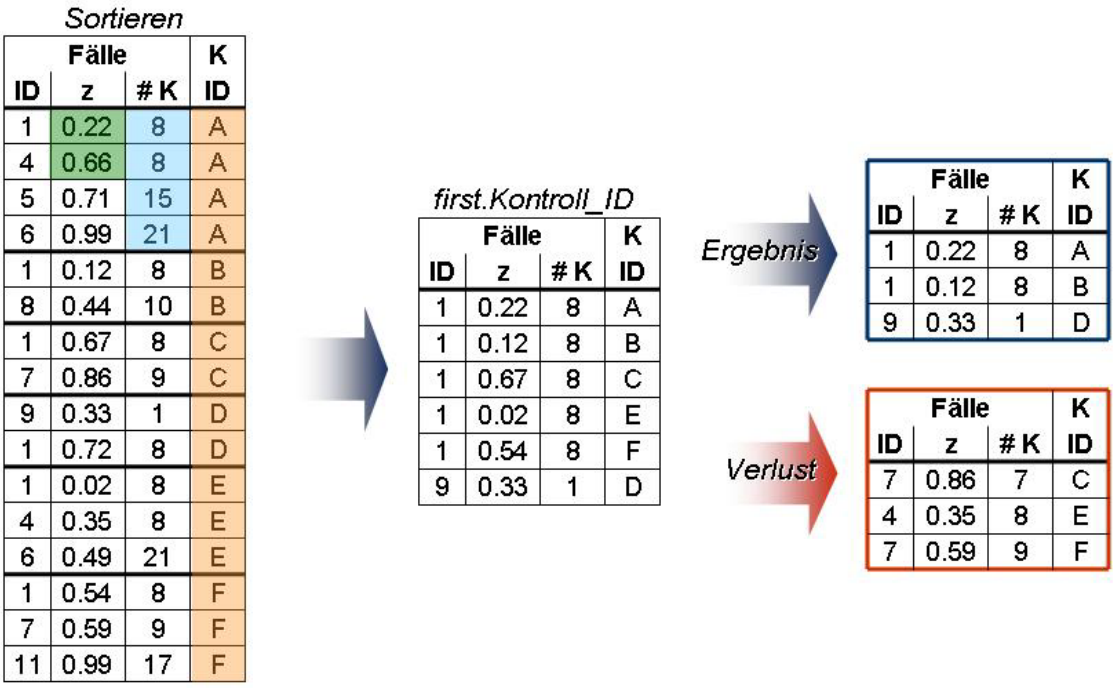


Abbildung 5: Beispiel für die Vorgehensweise des in Kawabata et.al. beschriebenen Algorithmus.[6]

3.2 Modifizierter SQL-Abgleich

Durch ein vom Autor entworfenes iteratives Verfahren (Methode 2) lässt sich die oben beschriebene Schwäche des Algorithmus beheben. Dazu wurde ein Makro entworfen, welches die Ergebnistabelle nach den beiden Sortierschritten mit der Ursprungstabelle abgleicht und danach aus der reduzierten Ursprungstabelle eine weitere Ergebnistabelle durch Sortieren erstellt, die an die erste Ergebnistabelle angehängt wird. Dieser Vorgang wird so oft wiederholt, bis die Ursprungstabelle keine Einträge mehr enthält.

[5] Genau dies war ja aber gefordert worden, damit die Fälle mit wenigen Kontrollen auch eine Chance auf ein erfolgreiches Matching haben.

[6] "#K" steht für Anzahl Kontrollen pro Fall, "K ID" ist die ID der Kontrollen und "z" die Zufallszahl. Das hier aufgeführte Beispiel ist zwar ziemlich unwahrscheinlich, es treten jedoch durchaus häufiger Konstellationen auf, bei denen sich immer wieder ähnliche Anordnungen für verschiedene Fälle ergeben.

In Abbildung 6 ist die Vorgehensweise des Makros anschaulich dargestellt. Entsprechend den eingefassten Fall-Kontroll-Paaren der ersten Ergebnistabelle (rechts) werden in der sortierten Ursprungstabelle (links) alle Kontroll- und alle Falleinträge getrennt eliminiert, da diese nicht mehr berücksichtigt werden dürfen. Mit der reduzierten Ursprungstabelle wird dann der Vorgang solange wiederholt, bis diese leer ist.

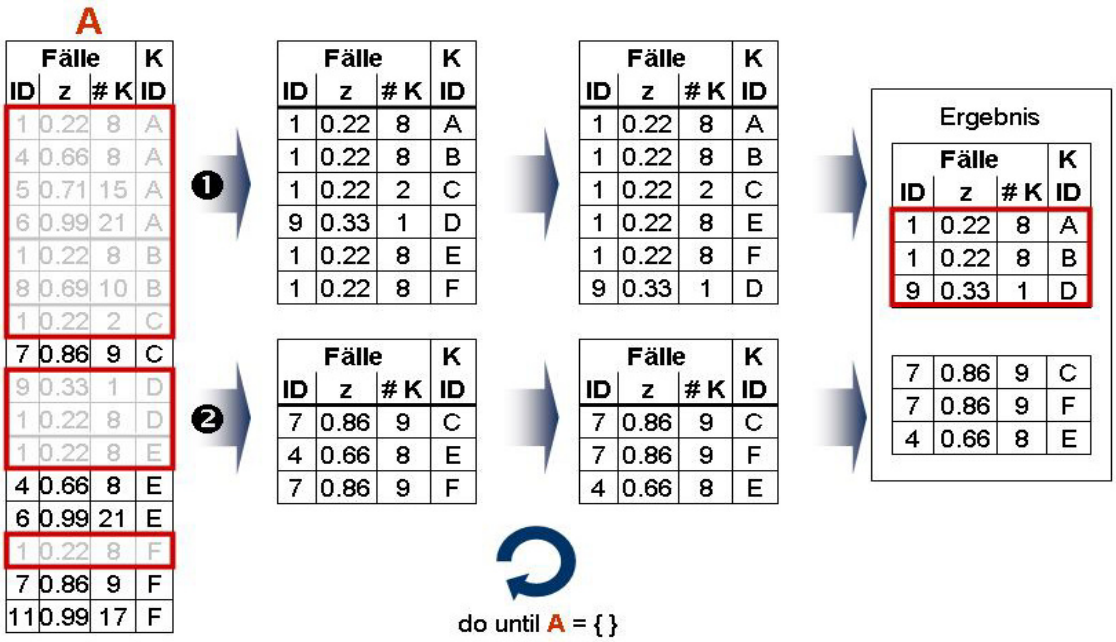


Abbildung 6: Modifizierter Algorithmus zur Optimierung des Abgleichs

Das entsprechende Makro hat folgenden Code:

```
%MACRO Optimierung(TabIn=, TabOut=, caseID=, controlID=,
numControls=, zzahl=);
   DATA _reduction; SET &TabIn.; RUN;
   PROC SQL NOPRINT; SELECT COUNT(*) INTO :_count FROM &TabIn.;
   QUIT;
   %DO %UNTIL (&_count. = %SYSEVALF(0));
       PROC SORT DATA =_reduction;
           BY &controlID. &numControls. &zzahl.; RUN;
       DATA _set1; SET _reduction; BY &controlID.;
           IF first.&controlID.;
       RUN;
       PROC SORT DATA = _set1; BY &caseID.; RUN;
       DATA _set2; SET _set1; BY &caseID.;
           IF (first.&caseID. or last.&caseID.); /*1:2-Matching*/
       RUN;
       PROC APPEND BASE = &TabOut. DATA =_set2; RUN;
       PROC SORT DATA = _reduction; BY &controlID.; RUN;
       PROC SORT DATA = _set2; BY &controlID.; RUN;
       DATA _reduction;
           MERGE _reduction _set2 (in = b keep = &controlID.);
```

```
            BY &controlID.;
            IF not b;
        RUN;
        PROC SORT DATA = _reduction; BY &caseID.; RUN;
        PROC SORT DATA = _set2; BY &caseID.; RUN;
        DATA _reduction;
            MERGE _reduction _set2 (in = b keep = &caseID.);
            BY &caseID.;
            IF not b;
        RUN;
        PROC SQL NOPRINT; SELECT COUNT(*) INTO :_count
            FROM _reduction; QUIT;
    %END;
%MEND Optimierung;
```

Dem Makro muss neben der Ursprungstabelle und den Fall- und Kontroll-IDs auch die Variable der Zufallszahl und die Variable für die Anzahl von Kontrollen pro Fall übergeben werden. Die einzelnen Arbeitsschritte des Makros sehen wie folgt aus:

- Übergabe der Anzahl der Einträge in der Ursprungstabelle an die DO-Schleife
- Sortieren der Ursprungstabelle nach Kontrollen, Auswählen der ersten Kontrolle
- Erneutes Sortieren, diesmal nach Fällen
- Auswählen des ersten und letzten Falles (1:2 Matching)[7] → Ergebnistabelle
- Hinzufügen der Ergebnistabelle zur Menge der Ergebnisse mit `PROC APPEND`
- Entfernen der Kontrollen der Ergebnisstabelle aus der Ursprungstabelle
- Entfernen der Fälle der Ergebnistabelle aus der Ursprungstabelle
- Erneutes Zählen der restlichen Einträge in der Ursprungstabelle und die Übergabe des Wertes an die DO-Schleife.

Mit diesem verbesserten Ansatz der Ergebnisaufbereitung konnten in *Szenario I* zu 273 Fällen jeweils zwei Kontrollen, zu 25 Fällen nur eine Kontrolle gefunden werden. In *Szenario II* werden noch für 169 Fälle 2 Kontrollen und für 57 Fälle eine Kontrolle gefunden. Methode 2 führt also zu erheblich verbesserten Ergebnissen.

3.3 Propensity Scores

In einem Artikel von Parsons et.al. wird eine Lösung des Matching-Problems mit Propensity Scores beschrieben [2] (Methode 3). Dazu wird mit einem logistischen Modell für jede Person die Vorhersage-Wahrscheinlichkeit berechnet, aufgrund ihrer individuellen Matching-Variablen-Struktur ein Fall zu sein. Das logistische Modell lässt sich für die hier untersuchten Szenarien wie folgt formulieren:

$$\log\left(\frac{p(F)}{1-p(F)}\right) = \alpha + \beta_1 \cdot ALTER + \beta_2 \cdot ZUZUG$$

[7] Eine für ein beliebiges 1:N Matching modifizierte Version kann auf E-Mail-Anfrage zur Verfügung gestellt werden.

bzw. $p(F) = P[F = 1 \mid X] = \dfrac{\exp(\alpha + \beta_1 \cdot ALTER + \beta_2 \cdot ZUZUG)}{1 + \exp(\alpha + \beta_1 \cdot ALTER + \beta_2 \cdot ZUZUG)}$

Man könnte das Geschlecht noch als zusätzliche Variable in das Modell aufnehmen. Da das Modell Wahrscheinlichkeiten und keine Eindeutigkeiten zuordnet, könnten dann aber Fällen Kontrollen unterschiedlichen Geschlechts zugewiesen werden, welche dann nachträglich entfernt werden müssten. Daher werden hier getrennte Modelle für Frauen und Männer gebildet. Der entsprechende Code in SAS sieht folgendermaßen aus:

```
PROC LOGISTIC DATA = Frauen;
     MODEL F_K = Alter Zuzug/
         SELECTION = NONE RISKLIMITS LACKFIT RSQUARE PARMLABEL;
     OUTPUT OUT = Propensity_f PROB = prob;
RUN;
```

Die Variable F_K ist eine binäre Variable, die bezeichnet, ob es sich um einen Fall oder um eine Kontrolle handelt. Für die hier beschriebenen Szenarien war es hilfreich, keine Variablenselektion vorzunehmen. Bei einer größeren Anzahl von Variablen können aber verschiedene Selektionen wie z.B. Stepwise zu einem reduzierten Modell mit besseren Ergebnissen führen. `PROC LOGISTIC` erstellt eine Tabelle, in der jedem Fall und jeder Kontrolle eine Vorhersagewahrscheinlichkeit zugeordnet ist. Nun muss man anhand dieser Wahrscheinlichkeiten die passenden Kontrollen den Fällen zuordnen. Parsons et.al. stellen dafür ein umfangreiches Makro bereit (s. [2]), das diese Zuordnung in einer Art Nearest Neighborhood-Abgleich durchführt[8].

Die Einschränkung des Matching-Ergebnisses auf Treffer innerhalb des Toleranzbereiches für das Alter und den Zuzug muss man nachträglich vornehmen, da die Ergebnistabelle grundsätzlich zu jedem Fall zwei Kontrollen enthält, diese aber in ihren tatsächlichen Werten erheblich von den Fällen abweichen können. Das liegt an der ausschließlichen Zuordnung über die berechneten Wahrscheinlichkeiten, beginnend mit 8-stelliger Übereinstimmung der Nachkommastellen und dann schrittweise reduziert bis auf die Übereinstimmung der jeweils ersten Ziffer.

Mit Propensity Score und nachfolgendem Nearest Neighborhood-Matching konnten für *Szenario I* nur zu 72 Fällen zwei Kontrollen gefunden werden, zu 118 Fällen jeweils eine Kontrolle. In *Szenario II* lieferte dieses Verfahren nur zu 32 Fällen zwei passende Kontrollen, zu 48 Fällen eine Kontrolle.

[8] Für das Matching anhand der Propensity Scores wurden verschiedene Verfahren entwickelt. Neben Nearest Neighborhood kann z.B. auch ein stratifiziertes Matching, Caliper Matching oder ein Difference-in-differences Matching durchgeführt werden. Je nach verwendeter Methode kann sich auch die Ergebnismenge sowohl in ihrer Größe als auch in den auftretenden Paarungen erheblich unterscheiden.

4 Vergleich von `PROC SQL` und Propensity Score

Die folgende Tabelle gibt einen Überblick über die mit den einzelnen Verfahren erzielten Ergebnisse. Zusätzlich zu den bereits beschriebenen Szenarien wurden die Prozeduren noch einmal mit 10.000 statt nur 900 möglichen Kontrollen getestet.

Tabelle 1: Vergleich der Ergebnisse der Methoden bei verschiedenen Szenarien

		300 Fälle, **900 Kontrollen**			300 Fälle, **10.000 Kontrollen**		
		2 K	1 K	keine	2 K	1 K	keine
Szenario I Gleiche Altersvert.	Methode 1 PROC SQL	146	23	131	186	1	113
	Methode 2 SQL modifiziert	273	25	2	298	2	0
	Methode 3 Propensity Score	72	118	110	288	8	4
Szenario II Ungleiche Altersvert.	Methode 1 PROC SQL	123	0	177	173	2	125
	Methode 2 SQL modifiziert	169	57	74	297	3	0
	Methode 3 Propensity Score	32	48	220	253	35	12

Aus der Tabelle wird ersichtlich, dass für die simulierten Szenarien Methode 2 immer die größte korrekte Treffermenge liefert. Für *Szenario II* mit 900 möglichen Kontrollen wird zwar zu 74 Fällen keine Kontrolle gefunden, eine vollständige Abdeckung aller Fälle mit Kontrollen scheint in diesem Szenario aber auch nicht möglich. Die Propensity Score-Methode (Methode 3) liefert für *Szenario I* bei 900 möglichen Kontrollen insgesamt zu mehr Fällen passende Kontrollen als Methode 1, unter diesen finden sich aber viele Fälle, denen nur eine Kontrolle zugeordnet wurde. Bei ungleicher Altersverteilung schneidet diese Methode schlechter ab als `PROC SQL` ohne Modifikation. Für einen großen Pool von möglichen Kontrollen hingegen nähern sich die Ergebnisse Methode 2 und 3 an.

Das relativ schlechte Ergebnis der Propensity Score Methode bei wenigen Kontrollen kann mehrere Ursachen haben. Zum einen gibt es wie erwähnt mehrere Methoden das Matching mit den erhaltenen Propensity Scores durchzuführen - hier wurde nur die Nearest Neighborhood-Methode angewandt. Außerdem können noch Verbesserungen durch Variation der Anzahl der Nachkommastellenvergleiche in dem bereitgestellten Makro erreicht werden. Zum anderen besteht das Regressionsmodell hier nur aus zwei Einflussgrößen, so dass diese Methode ihr Potential in den hier beschriebenen Szenarien womöglich nicht voll ausspielen kann. Propensity Scores bieten vor allem dann Vorteile, wenn ein Matching nach einer Vielzahl von Variablen durchgeführt werden soll

(z. B. bei einer klinischen Beobachtungsstudie mit sehr vielen Laborparametern) und wenn es sich um eine große Beobachtungsstudie handelt.

Abschließend bleibt festzuhalten, dass für Beobachtungsstudien mit einem relativ begrenztem Pool an möglichen Kontrollen und/oder nur wenigen Matching-Variablen ein Matching mit `PROC SQL` und einer wie hier vorgestellten nachfolgenden iterativen Bearbeitung der Ergebnisse sehr gute Resultate liefert.

Hinweis:
Zu Methode 2 wurde vom Autor ein flexibles Makro entwickelt, welches die komplette Matching-Prozedur enthält und verschiedene Matching-Szenarien bewerkstelligen kann. Dieses Makro kann auf Anfrage zur Verfügung gestellt werden (a.deckert@uni-heidelberg.de).

Literatur

[1] H. Kawabata, et.al.: Using SAS ® to Match Cases for Case Control Studies. SUGI 29, 173-29, Princeton, New Jersey

[2] L.S. Parson, et.al.: Performing a 1:N Case-Control Match on Propensity Score. SUGI 29, 165-29, Seattle, Washington

[3] R. Bender: Simulation von Überlebenszeiten mit Hilfe von SAS. http://www.rbsd.de/PDF/simcoxsas.pdf (Zugriff 12.02.2011)