Performance von SAS beim Aggregieren großer Datensätze. Vergleich von vier Möglichkeiten

Falk Hoffmann
Universität Bremen
ZeS, Abteilung Gesundheitsökonomie,
Gesundheitspolitik und Versorgungsforschung
Außer der Schleifmühle 35-37
28203 Bremen
hoffmann@zes.uni-bremen.de

Zusammenfassung

Das Ziel dieser Arbeit war es, am Beispiel von Arzneimittelroutinedaten verschiedene Möglichkeiten zur Aggregierung großer Daten in SAS bezüglich der benötigten Rechenzeit zu vergleichen. Es wurden Datensätze zwischen 1 und 100 Mio. Zeilen (bzw. verordneter Präparate) erzeugt und die Anzahl Packungen je Präparat aufsummiert und in einen Ausgabedatensatz geschrieben. Dies geschah mittels PROC MEANS, PROC FREQ, PROC SQL und dem FIRST-Statement im DATASTEP. Generell fanden sich zwischen PROC MEANS und PROC FREQ nur geringfügige Unterschiede in der Rechenzeit, wobei PROC SQL und DATA FIRST deutlich langsamer waren als die beiden erstgenannten Varianten. Eine schnellere Berechnung wird erreicht, wenn sich weniger Variablen im Datensatz befinden. Obwohl PROC SQL und DATA FIRST bei den hier durchgeführten Auswertungen großer Datensätze deutlich länger für die Analysen benötigten, dürfen die Möglichkeiten dieser Operationen insgesamt nicht unterschätzt werden.

Schlüsselwörter: Routinedaten, PROC MEANS, PROC FREQ, PROC SQL

1 Einleitung

Im Gesundheitswesen fallen umfangreiche, ursprünglich zu Abrechnungszwecken generierte Routinedaten an, die elektronisch erfasst sind und über die letzten Jahre immer häufiger auch für Forschungszwecke genutzt werden. Diese Daten liegen den Krankenkassen unterschieden nach Leistungsbereichen vor. Innerhalb der Gesetzlichen Krankenversicherung (GKV), in der aktuell etwa 85% aller Deutschen versichert sind, wird besonders der Arzneimittelsektor als transparent angesehen. Seit über 20 Jahren wird jährlich der Arzneiverordnungs-Report veröffentlicht, der mittlerweile alle zu Lasten der GKV in öffentlichen Apotheken abgegebenen Rezepte analysiert [1]. Auch einzelne Krankenkassen liefern mittlerweile regelmäßige Auswertungen ihrer Verordnungszahlen [2]. Insgesamt wurden zu Lasten der GKV im Jahr 2006 685 Mio. Packungen Arzneimittel in öffentlichen Apotheken eingelöst, dies sind 9,7 Packungen pro Versichertem und Jahr [1]. Die Arzneimittelverordnungsdaten liegen den Krankenkassen in elektronischer Form vor. In diesen Datensätzen entspricht eine Zeile einem verordneten Arzneimittel, wobei berücksichtigt werden muss, dass auch mehrere Packungen eines

Präparates verordnet werden können (diese Variable wird als so genannter „Faktor“ bezeichnet). In etwa drei Viertel der Fälle wird nur eine Packung eines Präparates verschrieben, trotzdem muss dieser Fakt und damit die Variable „Faktor“ bei Analysen berücksichtigt werden.

Bei der Auswertung der üblicherweise in Primärerhebungen anfallenden Datenmengen wird die Wahl der dafür verwendeten Methode in aller Regel von den Gewohnheiten des Nutzers abhängen, da heutzutage moderne Rechner schnell die nötigen Ergebnisse liefern. Bei Datensätzen mit zahlreichen Variablen und mehreren Millionen Zeilen, wie es bei Arzneimittelroutinedaten von Krankenkassen der Fall ist, spielen Effizienzaspekte schon eher eine Rolle. Insgesamt existieren natürlich auch in SAS, gemäß dem Sprichwort „Viele Wege führen nach Rom“, verschiedene Varianten, um das gleiche Ergebnis zu produzieren. Bei alltäglichen Analysen großer Datenmengen in SAS waren uns Unterschiede bezüglich der benötigten Zeit zwischen diesen Varianten aufgefallen.

Ziel der Arbeit war es, am Beispiel von Arzneimittelroutinedaten verschiedene Möglichkeiten zur Aggregierung von Daten in SAS bezüglich der benötigten Rechenzeit systematisch zu vergleichen.

2 Methodik und weiteres Vorgehen

Aus einem Verordnungsdatensatz mit insgesamt 50 Mio. Zeilen wurden jeweils 100 einfache Zufallsstichproben mit Zurücklegen im Umfang von 1 bis 100 Mio. Zeilen gezogen. Insgesamt 17.512 verschiedene Präparate (Variable „kurzname“) befanden sich in diesem Datensatz. Es sollte dann die Anzahl verordneter Packungen je Präparat jeweils mit den 4 zu vergleichenden Varianten aggregiert und in einen Datensatz geschrieben werden. Dieses Vorgehen wurde einmal mit den beiden relevanten Variablen („kurzname“, „faktor“) und einmal mit allen 9 im Ursprungsdatensatz befindlichen Variablen durchgeführt. Die pro Berechnung benötige Dauer wurde über die Systemzeit mit den Funktionen TIME und DATE gemessen.

Die Berechnungen geschahen auf vier Wegen, die im Folgenden mit der jeweiligen Syntax aufgezeigt sind:

Variante 1 (PROC MEANS):

```
PROC MEANS DATA=&out NOPRINT;
    OUTPUT OUT=kurzname_means
    SUM(faktor)=summe;
    CLASS kurzname;
RUN;
```

Variante 2 (PROC FREQ):

```
PROC FREQ DATA=&out NOPRINT;
    TABLES kurzname / OUT=kurzname_freq;
    WEIGHT faktor;
RUN;
```

Variante 3 (PROC SQL):

```
PROC SQL;
    CREATE TABLE kurzname_sql AS SELECT kurzname,
    SUM(faktor) AS summe FROM &out GROUP BY kurzname;
    QUIT;
RUN;
```

Variante 4 (DATA FIRST):

```
PROC SORT DATA=&out;
    BY kurzname;
RUN;

DATA &out;
    SET &out;
    BY kurzname;
    IF FIRST.kurzname THEN summe=faktor;
        ELSE summe+faktor;
    IF LAST.kurzname;
RUN;
```

Folgende Variablen befanden sich im Datensatz:

- kurzname (Format: $35.)
- faktor (Format: 12.)
- picnr (Format: $20.)
- pos (Format: 3.)
- pznz (Format: 7.)
- dar (Format: $3.)
- einzelbrutto (Format: 12.2)
- ddd_wido (Format: 12.3)
- atc_wido (Format: $7.)

Für alle Berechnungen mit SAS 9.1 für Windows wurde ein handelsüblicher Pentium 4 mit 3,06 GHz und 512 MB RAM genutzt.

3 Ergebnisse

3.1 Nur die beiden relevanten Variablen im Datensatz

Die benötigten Rechenzeiten sind für eine ausgewählte Anzahl an Auswertungen in Tabelle 1 gezeigt und der Verlauf ist in Abbildung 1 dargestellt. Verwendet man nur die beiden benötigten Variablen „kurzname“ und „faktor“, liefert PROC MEANS nahezu bei allen durchgeführten Berechnungen am schnellsten Ergebnisse (n=50 Mio. Zeilen in 1 Min.; n=100 Mio. Zeilen in 1,9 Min.). Anschließend folgte PROC FREQ (n=50 Mio. Zeilen in 1,7 Min.; n=100 Mio. Zeilen in 2,6 Min.), welches bei nahezu allen Berechnungen, wenn auch nur geringfügig, länger an Zeit brauchte als PROC MEANS. Für die gleichen Ergebnisse benötigten PROC SQL (n=50 Mio. Zeilen in 11,4 Min.; n=100 Mio. Zeilen in 29,6 Min.) und DATA FIRST (n=50 Mio. Zeilen in 14,5 Min.; n=100 Mio. Zeilen in 29,3 Min.) mehr als das Zehnfache der Zeit von PROC MEANS bzw. PROC SQL. Bis etwa 70 Mio. Zeilen pro Datensatz kommt PROC SQL im Vergleich zu DATA FIRST kontinuierlich schneller zum Ergebnis, bei größeren Datensätzen verschwinden diese Unterschiede.

Tabelle 1: Vergleich der benötigten Rechenzeit (in Minuten), wenn sich nur die beiden benötigten Variablen im Datensatz befinden

Anzahl Zeilen	PROC MEANS	PROC FREQ	PROC SQL	DATA FIRST
10 Mio.	0,2	0,2	1,5	2,3
20 Mio.	0,4	0,4	4,0	4,8
30 Mio.	0,7	0,8	6,2	8,2
40 Mio.	0,8	1,4	9,5	11,3
50 Mio.	1,0	1,7	11,4	14,5
60 Mio.	1,2	1,2	15,0	18,3
70 Mio.	1,6	1,6	18,9	19,2
80 Mio.	1,6	2,4	21,5	23,4
90 Mio.	2,0	1,9	26,1	24,4
100 Mio.	1,9	2,6	29,6	29,3

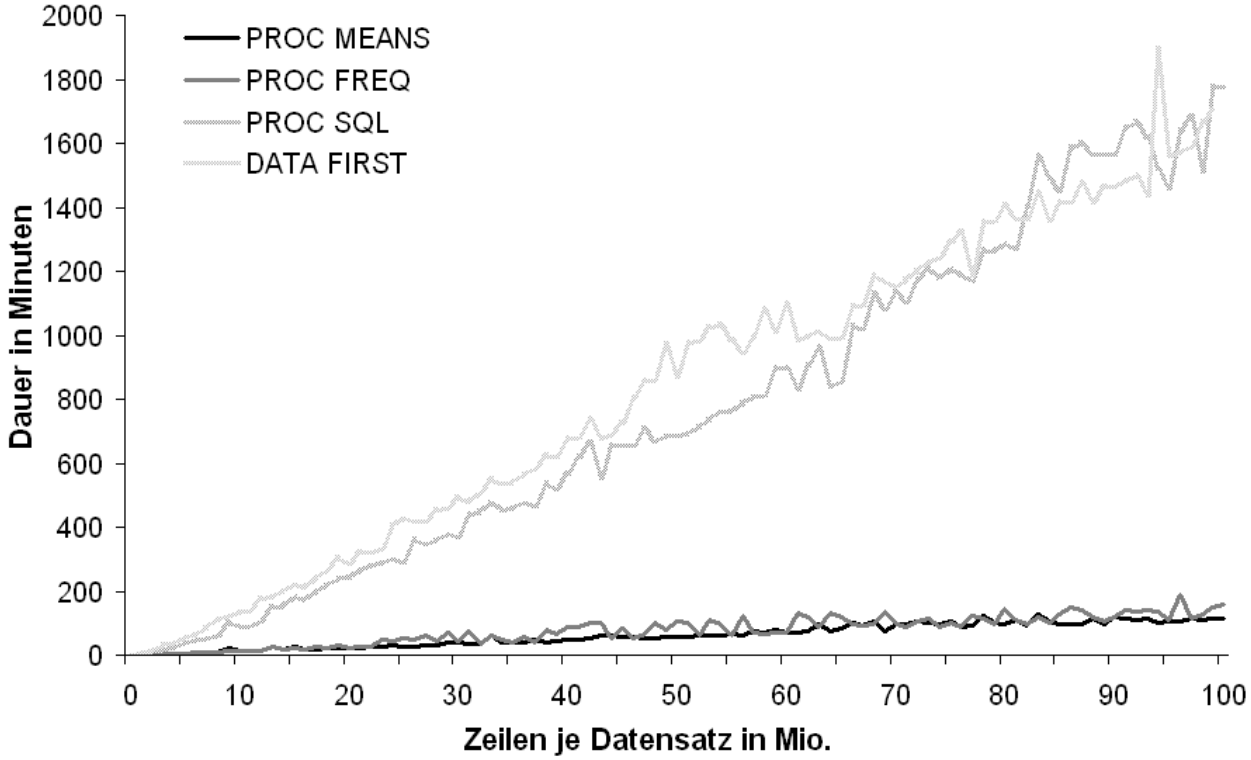

Abbildung 1: Vergleich der Rechenzeit, wenn sich nur die beiden benötigten Variablen im Datensatz befinden

3.2 Alle 9 Variablen im Datensatz

Die benötigten Rechenzeiten sind für eine ausgewählte Anzahl an Auswertungen in Tabelle 2 gezeigt und der Verlauf ist in Abbildung 2 dargestellt. Verbleiben alle 9 Variablen in Analysedatensatz, liefern PROC MEANS (n=50 Mio. Zeilen in 2,2 Min.; n=100 Mio. Zeilen in 5,0 Min.) und PROC FREQ (n=50 Mio. Zeilen in 2,1 Min.; n=100 Mio. Zeilen in 4,3 Min.) in nahezu identischer Rechenzeit Ergebnisse, wobei insgesamt auch keinerlei Tendenzen zu Gunsten einer Variante zu finden sind. PROC SQL benötigt für die gleichen Analysen durchschnittlich in etwa das Sechsfache an Zeit (n=50 Mio. Zeilen in 13,0 Min.; n=100 Mio. Zeilen in 35,0 Min.) und DATA FIRST ist nochmals deutlich langsamer (n=50 Mio. Zeilen in 35,8 Min.; n=100 Mio. Zeilen in 63,6 Min.).

Grundsätzlich sind, mit der Ausnahme von PROC SQL, die anderen durchgeführten Rechnungswege mit allen 9 Variablen deutlich langsamer als wenn sich lediglich die beiden notwendigen Variablen im Datensatz befinden.

Tabelle 2: Vergleich der benötigten Rechenzeit (in Minuten), wenn sich alle 9 Variablen im Datensatz befinden

Anzahl Zeilen	PROC MEANS	PROC FREQ	PROC SQL	DATA FIRST
10 Mio.	0,4	0,4	2,1	5,1
20 Mio.	0,8	0,7	4,5	11,7
30 Mio.	1,3	1,2	7,2	17,7
40 Mio.	1,8	1,7	9,8	28,2
50 Mio.	2,2	2,1	13,0	35,8
60 Mio.	2,6	2,5	16,0	35,9
70 Mio.	3,6	3,4	21,5	40,0
80 Mio.	3,3	3,3	22,0	46,4
90 Mio.	4,0	3,9	28,9	52,8
100 Mio.	5,0	4,3	35,0	63,6

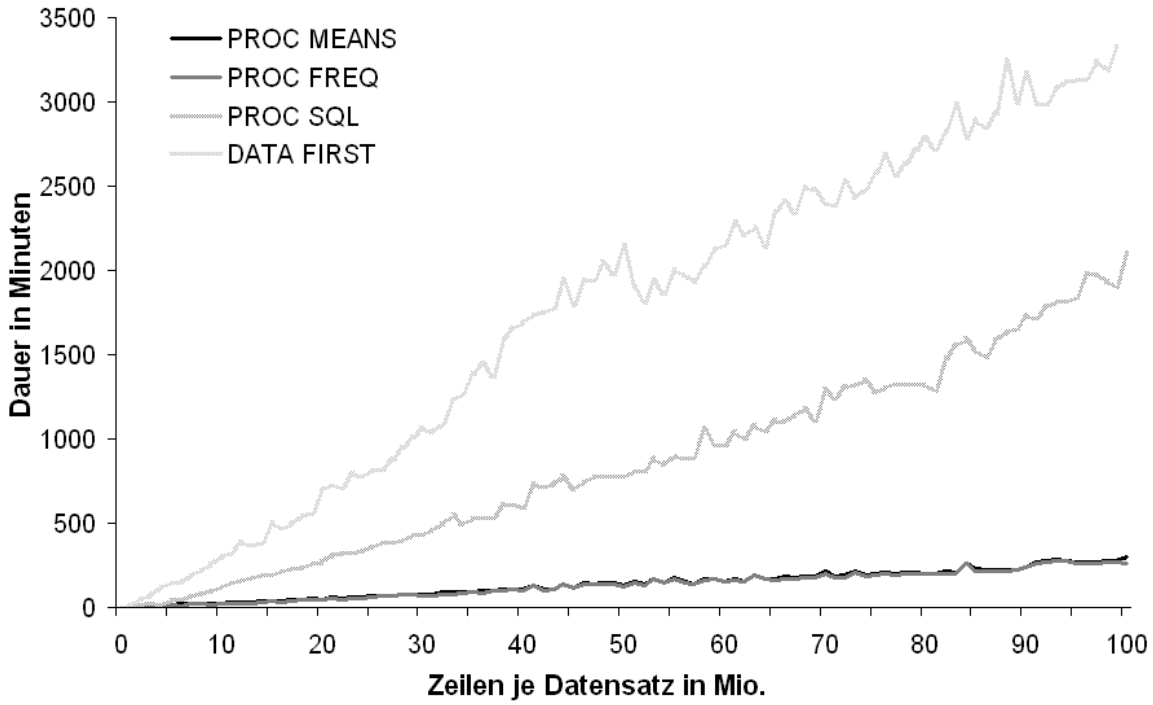

Abbildung 2: Vergleich der Rechenzeit, wenn sich alle 9 Variablen im Datensatz befinden

4 Fazit

Das Aggregieren von Datensätzen ist in SAS über verschiedene Wege möglich, wobei bei größeren Datensätzen PROC SQL und das FIRST-Statement im DATASTEP aufgrund der benötigten Rechenzeit nicht empfohlen werden können. Die deutlich schlechtere Performance von DATA FIRST bei Berücksichtigung einer größeren Anzahl von Variablen bzw. Zeilen im Datensatz ist auf die Zeit für das vorab benötigte Sortieren zurückzuführen. Eine schnellere Berechnung wird, mit Ausnahme von PROC SQL erreicht, wenn sich weniger Variablen im Datensatz befinden. Nicht unterschätzt werden darf allerdings die Möglichkeit, mit PROC SQL Operationen einfach durchzuführen, die in Prozeduren oder im DATASTEP erheblichen Programmieraufwand bedeuten (s.B. inner join bei many-to-many-merge oder count distinct) [3]. Auch das FIRST-Statement ist für viele Operationen nützlich, es ist allerdings beim Aggregieren (großer Datensätze) weder vom Programmieraufwand noch von der benötigen Zeit zu empfehlen.

Literatur

[1] U. Schwabe, D. Paffrath (Hrsg.): Arzneiverordnungs-Report 2007. Heidelberg, Springer, 2008.

[2] G. Glaeske, K. Janhsen (Hrsg.): GEK-Arzneimittel-Report 2007. St. Augustin, Asgard, 2007.

[3] W. Hu: Top Ten Reasons to Use PROC SQL. SUGI 29 Proceedings, Paper 042-29. Montreal, Canada, 2004.