SAS und R - ein ungleiches Paar

Dr. Peter Bewerunge
HMS
Analytical Software GmbH
Rohrbacher-Str. 16
D-69115 Heidelberg
peter.bewerunge@analytical-software.de

Andreas Mangold
HMS
Analytical Software GmbH
Rohrbacher-Str. 16
D-69115 Heidelberg
andreas.mangold@analytical-software.de

Zusammenfassung

Dieser Beitrag nimmt eine wertneutrale Gegenüberstellung der SAS Language und R Language vor. Die grundsätzlichen Konzepte hinter SAS und R werden vorgestellt und ein Vergleich der Funktionalitäten vorgenommen. Es werden Antworten gegeben auf Fragen wie z.B. „wie werden Programme erstellt“, „was ist der Unterschied zwischen der SAS Fensterumgebung und der RGui“ und „wie unterscheidet sich grundsätzlich der Programmcode?“
Ein Performancevergleich bei großen Datenmengen gibt Aufschluss über Geschwindigkeit und physikalische Limitierungen. Ein wichtiges Thema ist die Integrationsfähigkeit der jeweils anderen Software. Dazu wird auf die von SAS bereitgestellte Schnittstelle SAS IML 9.22 eingegangen. Aber auch eine entgegengesetzte Schnittstelle von R zu SAS wird vorgestellt.

Schlüsselwörter: SAS Fensterumgebung, PROC IML, Makro, R, RGui, R Funktionen, Performancevergleich

1 Gegenstand des Vergleichs

Dieser Beitrag stellt die SAS Language mit der SAS Fensterumgebung der R Language mit der RGui gegenüber.

Aus der Betrachtung ausgeklammert wird die SAS Produktlandschaft mit einer Vielzahl an Frontends, Servern, Middleware-Komponenten und Metadatenverwaltung. Ebenso wird nicht auf die unzähligen Erweiterungsmöglichkeiten von R im Bezug auf Pakete, Frontends und Servern eingegangen.

2 Konzepte von SAS und R

Hinter den beiden Softwareprodukten SAS und R stecken zwei unterschiedliche Konzepte. Ein augenscheinlicher Unterschied ergibt sich aus der kommerziellen Vermarktung der einen Software und der kostenlosen Verfügbarkeit der anderen als Open Source Software (OSS).

Die **S**tatistical **A**nalysis **S**oftware wurde Anfang der 1970er an der North Carolina State University von Jim Goodnight, Jon Sall und anderen Forschern entwickelt. Aus der zunächst in der landwirtschaftlichen Forschung eingesetzten Software entwickelte sich 1976 die Firma SAS Institute mit Sitz in Cary, North Carolina. Heute (März 2011) ist SAS mit 11.500 Angestellten und ca. 2.43 Milliarden Jahresertrag (in 2010) die weltweitgrößte Software-Firma in Privatbesitz [1]. Die Kunden verteilen sich auf ca. 50.000 Unternehmen in ca. 100 Ländern. Dabei liegt der Fokus der SAS Software u.a. auf Datenintegration / Data Warehousing und Bussiness Intelligence, aber auch in den Bereichen Forschung und Entwicklung ist SAS stark vertreten.

R basiert auf der (Programmier-) Sprache S und wurde in den frühen 1990ern von Robert Gentleman und Ross Ihaka an der University of Auckland entwickelt. Zunächst sollte die Software lediglich in der Lehre eingesetzt werden. Motiviert durch positive Rückmeldungen wurde R 1993 unter die GNU General Public Licence auf einem FTP-Server kostenlos zur Verfügung gestellt [2]. Mittlerweile (März 2011) ist die Nutzergemeinde auf geschätzte 2 Millionen Personen angewachsen. Der Fokus der Software liegt auf statistischen Berechnungen und Grafiken. Die offizielle Stimme des R-Projekts ist die R-Foundation, ein gemeinnütziger Verein mit Sitz in Wien. Die Mitglieder bestehen hauptsächlich aus dem R Development-Core-Team. Dieser Verein besitzt und verwaltet auch die Copyrights der Software und deren Dokumentation. Zusätzlich kann R durch eine Vielzahl an freiverfügbaren Paketen nach Bedarf erweitert werden. Diese Pakete werden nicht zwingend von dem R Development Core-Team bereitgestellt, sondern werden vielmehr von weltweit verteilten Programmieren auf unterschiedlichen Plattformen entwickelt und der R-Community zur Verfügung gestellt.

3 Installation und Support

SAS bezieht man typischerweise über ein Download-Center oder eine DVD. Der Installationsprozess dauert vergleichbar lange und beinhaltet in seiner Basis-Version ein breites Angebot an Prozeduren wie z.B. PROC CORR, PROC PLOT usw. aber u.a. auch die mächtigen Makro-Funktionalitäten. R hingeben wird man von einer freiverfügbaren Quelle über das Internet beziehen, dem sogenannten Comprehensive R Archive Network (CRAN). Das Binary (R-2.12.2) ist ca. 40 MB groß und die Installation ist schnell durchgeführt. In der Basisversion von R sind die sogenannten Basisfunktionalitäten, Recommended- und Base-Pakete enthalten. Diese stellen Funktionalitäten wie z.B. arithmetische Rechenoperationen, Grafikerstellung und statistische Funktionen zur Verfügung.

Der Support von SAS ist über unterschiedliche Kanäle organisiert. Ein Kanal ist der „Electronic Mail Interface Technical Support“, der es online möglich macht, eine Anfrage abzuschicken, die dann von einem SAS-Mitarbeiter bearbeitet wird. Ebenso bietet SAS eine Hotline (06221- 415 200) für telefonische Anfragen an. Darüber hinaus gibt es ein durchdachtes Trainingskonzept, das nicht nur das Erlernen der SAS-Programmiersprache im Fokus hat, sondern auch auf die SAS Produktlandschaft ein-

geht. Für viele Produkte werden außerdem Zertifizierungsprüfungen angeboten. Für R existiert zwar keine Kundenhotline aber eine Community, die sich in Mailing-Listen mit unterschiedlichen Themengebieten organisiert. Über die R-Search-Funktion können u.a. R Webpages mit Manuals, Hilfeseiten und Mailing-Archive durchsucht werden. Für den Fall, dass man einen Bug in der Software und den dazugehörigen Paketen entdeckt, wurde ein Bug-Tracking-System eingerichtet. Zusätzlich gibt es von kommerziellen Anbietern R-Distributionen und Services.

4 Funktionsvergleich

4.1 Wie werden Programme erstellt?

Programme enthalten nacheinander ablaufende Code-Anweisungen. Diese werden typischerweise genutzt um Daten in Form von Statistiken, Grafiken und Berichte aufzuarbeiten. Diese Programme werden dann in der Regel als Skripte abgelegt und können zu einem späteren Zeitpunkt wiederverwendet werden. Programme können aber auch komplexe, in sich geschlossene Programmeblöcke mit einer Schnittstelle nach außen sein (z.B. SAS-Makros oder R-Funktionen).

Sowohl SAS als auch R bietet eine Umgebung zur Programmentwickelung an. Die Unterschiede in der Handhabung und im Funktionsumfang zwischen der SAS Fensterumgebung und der RGui werden im Folgenden gegenüber gestellt.

4.1.1 SAS Fensterumgebung versus RGui

Die SAS Fensterumgebung (Display Manager) besitzt einen Editor zum Erstellen und Ausführen von Programmen inkl. Syntax Highlighting, siehe Abb. 1. Schickt man die Anweisungen eines Programms ab, werden diese nacheinander vom SAS-Prozessor (Interpreter) ausgeführt. In das LOG-Fenster werden Informationen über die ausgeführten Anweisungen gespeichert, wie z.B. die ausgeführten Codezeilen, Rückmeldungen des Systems (Fehler, Warnungen, Berechnungsdauer usw.), aber auch Hinweise auf z.B. die Anzahl an Beobachtungen in einer Tabelle. Zugriff auf Daten, Ergebnisse und Berichte sind ebenfalls über diese Umgebung realisiert.

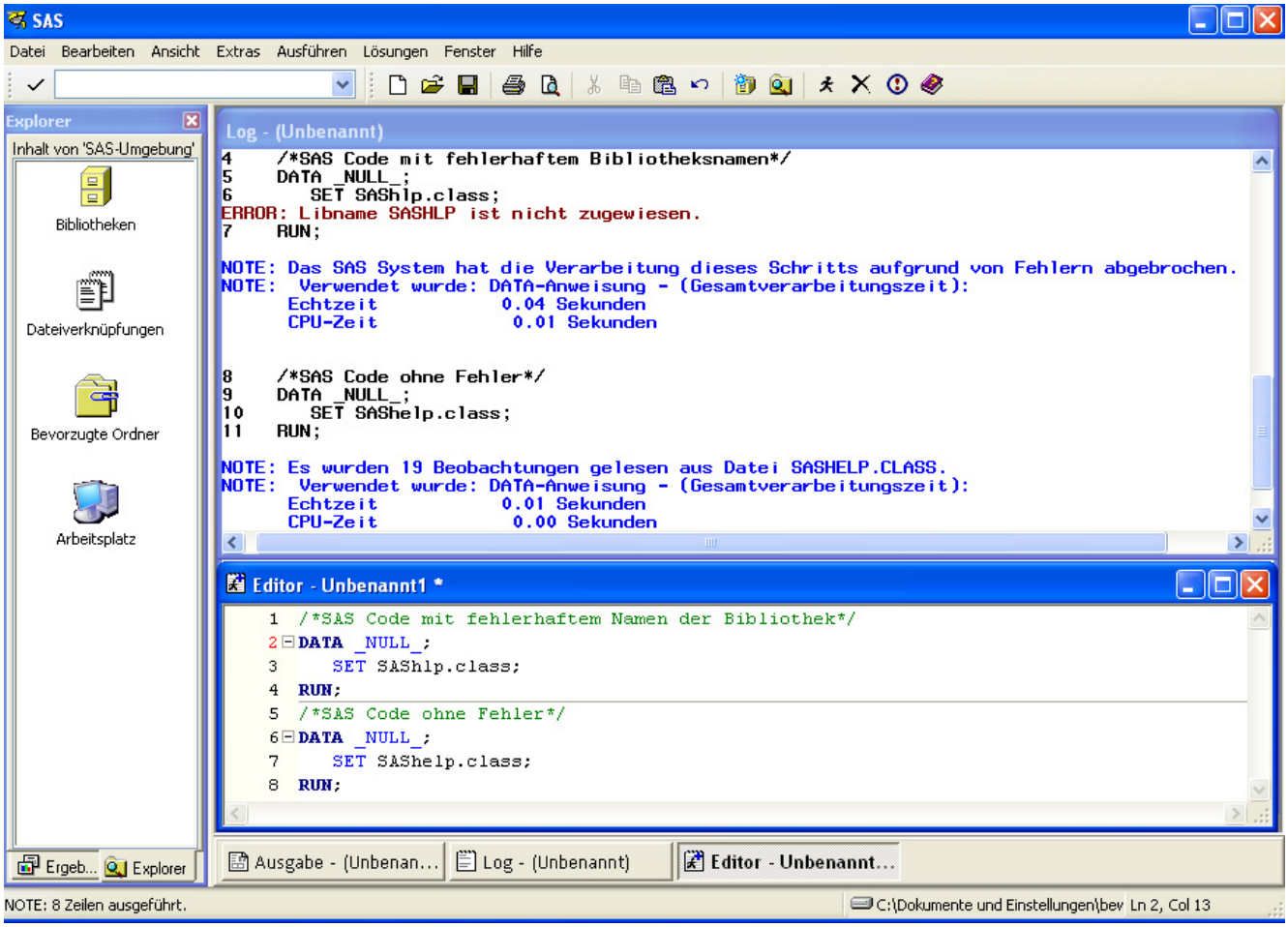

Abbildung 1: SAS Windowing Environment. Wesentlicher Bestandteil sind der Editor, das LOG und die Explorerleiste.

Beim Entwickeln von Programmen bietet die SAS Fensterumgebung einige nützliche Unterstützungsmöglichkeiten. So wird z.B. ein Werkzeug für die Erstellung von SQL-Abfragen auf Tabellen unter dem Menüpunkt „Extras" angeboten. Ebenso hilfreich ist die Möglichkeit sogenannte Keyboard Makros (nur in der englischen Version) zu nutzen. Durch Zuweisung einer Tastenkombination können dann vordefinierte Codesnippets aufgerufen werden. Aber auch selbsterstellte Codesnippets wie z.B. Codegerüste für immer wiederkehrende IF-ELSE Bedingungen können hier angelegt werden.

R bietet unter Windows die sogenannte RGui als Anwendungsoberfläche an. Diese besteht hauptsächlich aus der R Konsole, siehe Abbildung 2. Es ist vorgesehen in die Konsole Kommandos, sogenannte Expressions, einzugeben. Anders als bei der SAS Fensterumgebung, werden die Expressions dann zeilenbasiert durch Bestätigen mit der Return-Taste ausgeführt. Der Interpreter liest die Expression und gibt direkt das Ergebnis bzw. eine Fehlermeldung in die Konsole aus. Die R Konsole dient somit nicht nur als Codeeingabe-, sondern auch als Ausgabe-Fenster für Ergebnisse und Meldungen, ähnlich dem SAS-Log. Ein integrierter Skript-Editor erleichtert das Erstellen von Pro-

grammen und ermöglicht, dass Codezeilen einzeln oder als Block an die Konsole geschickt werden.

Besonders hilfreich ist der Menüpunkt „Pakete.“ Hier bekommt der Anwender Zugriff auf eine Vielzahl von Zusatzpaketen.

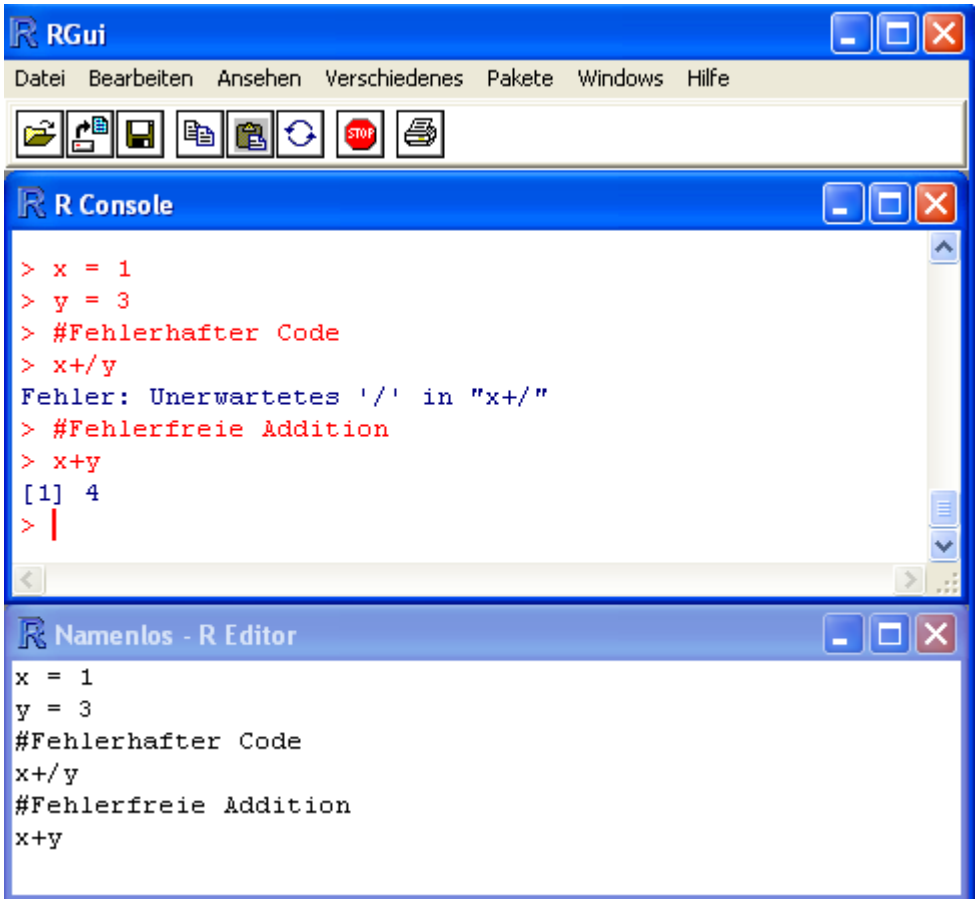

Abbildung 2: RGui. Der Editor ermöglicht das Erstellen von Programmen, die Zeilenweise an die R Console geschickt werden können. Die R Console ist gleichzeitig das LOG.

4.1.2 SAS Code versus R Code

Der SAS Code setzt sich aus Blöcken zusammen, die durch eine Anfangs- und Endanweisung definiert sind. Ein zentrales Element der SAS Sprache ist der Datenschritt (Data Step), der es ermöglicht Daten in vielfältiger Weise einzulesen und zu bearbeiten. Dieser beginnt mit der DATA-Anweisung und endet mit der RUN-Anweisung. In Abbildung 3 ist die Erstellung eines SAS Data Sets, ein DATA STEP und die Prozedur MEANS beispielhaft dargestellt. Einem Programmierer stehen numerische und alphanumerische Datentypen, Tabellen, Matrizen (nur IML) und Makrovariablen zur Verfügung.

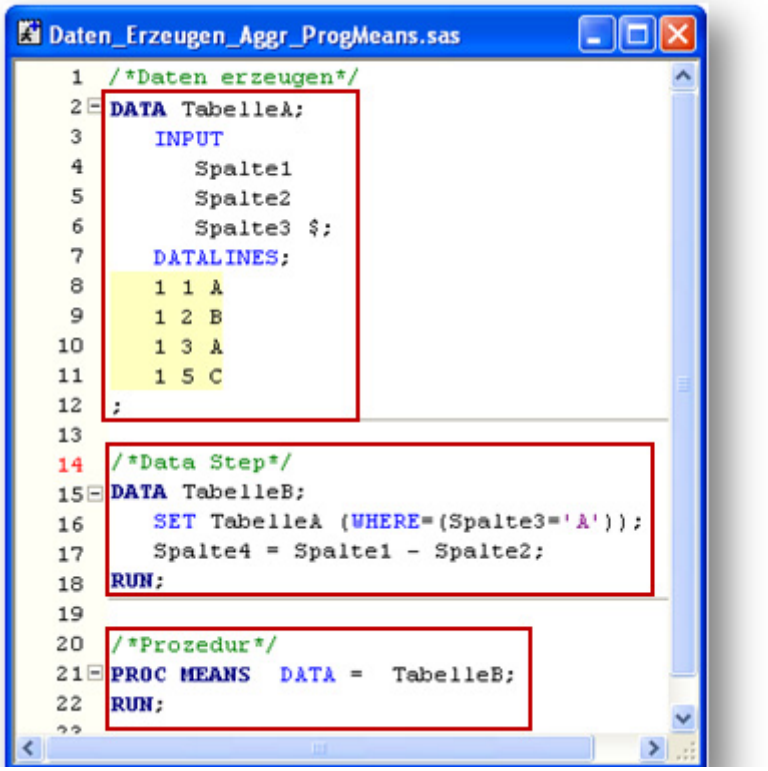

Abbildung 3: SAS Code ist in Blöcken aufgebaut.
Elementarer Bestandteil ist der Data Step und Prozeduren.

R Code hingegen ist nicht zwingend als Block aufgebaut, sondern ist vielmehr eine zeilenweise Aneinanderreihung von Codeanweisungen. In Abbildung 4 ist exemplarisch R Code dargestellt. R ist auf Matrixmanipulationen optimiert. Dazu tragen objektorientiere Datenstrukturen wie z.B. Vektoren, Matrizen, Dataframes (Matrizen, die u.a. auch alphanumerische Werte enthalten können) aber auch ein Klassenmodel bei. Spezielle Funktionalitäten sind in Funktionen gekapselt. So berechnet z.B. die Funktion summary() eine zusammenfassende Statistik über eine Matrix, siehe Abbildung 4.

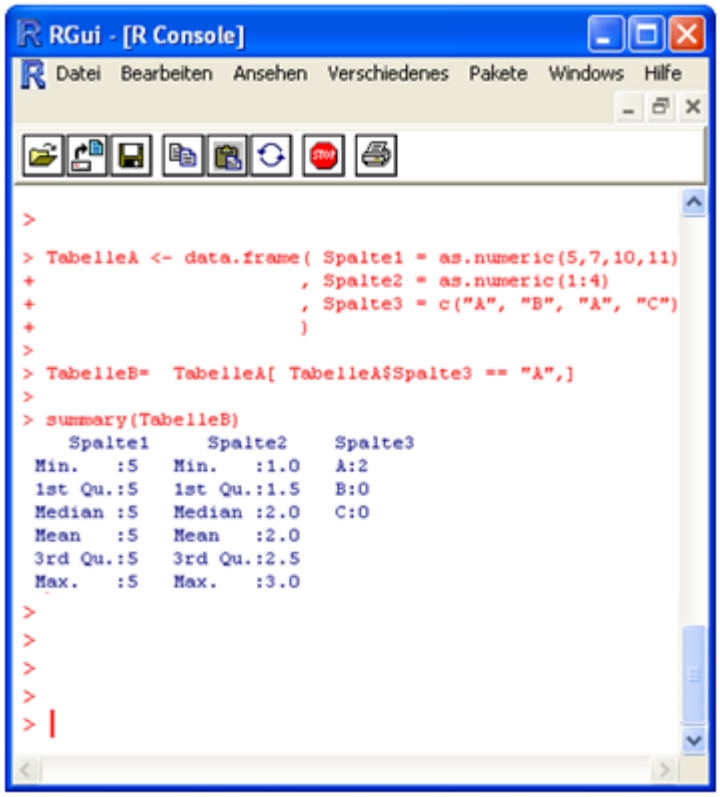

Abbildung 4: R-Code ist zeilenbasiert aufgebaut. Hier ist die Zuweisung eines Dataframes, mit anschließendem Filtern und der Erstellung einer zusammenfassenden Statistik mit der Funktion summary() gezeigt.

4.1.3 SAS Makro versus R Funktionen

Die SAS-Makro-Funktionalität stellt ein leistungsfähiges Werkzeug dar, um immer wiederkehrende Programmanweisungen in Form von z.B. Prozeduren oder Datenschritten in ein Programm zu verpacken. Ein SAS-Makro wird durch das Schlüsselwort %MACRO eingeleitet, gefolgt von dem Makronamen und wird durch %MEND beendet, siehe Abbildung 5. Ein Parameterkopf ermöglicht die Übergabe von Parametern an ein Makro, welche wiederum intern z.B. in Prozeduren verarbeitet werden können. Der Aufruf eines Makros beginnt mit dem %-Zeichen, gefolgt von dem Makronamen und ggf. dem Parameterkopf in Klammern. In der Regel wird man logisch zusammenhängende Makros als Makro-Bibliotheken zusammenfassen, um diese im Anwendungsfall gemeinsam laden zu können.

```
%MACRO BerechneMean (p_s_LibRef =
                   , p_sTableName =
                   , p_sOutTableName =
                     )
   %LET l_sTableRef = &p_s_LibRef..&p_sTableName.;
   PROC MEANS DATA = &l_sTableRef. MEAN ;
      VAR Height;
      CLASS Sex;
      OUTPUT OUT&=p_sOutTableName.;
   RUN;
%MEND BerechneMean;

/*Aufruf des Makros*/
%BerechneMean(  p_s_LibRef      = SasHelp
              , p_sTableName    = Class
              , p_sOutTableName = OutData
             );
```

Abbildung 5: SAS-Makro wird eingesetzt, um wiederkehrenden Programmcode zu kapseln. Der Aufruf erfolgt über das %-Zeichen, gefolgt von dem Makronamen.

Ebenso wie bei SAS kann man mit R wiederkehrende Programmsequenzen als eigenen Programmcode in R Funktionen kapseln. Einem Funktionsnamen wird über das Schlüsselzeichen „←“ und dem Schlüsselwort „function“ eine solche R-Funktion zugewiesen, siehe Abbildung 6. Vergleichbar zu SAS können Parameter übergeben werden, die in der Funktion verarbeitet werden. Anders als bei SAS wird hier keine Textersetzung vorgenommen, sondern dem Aufrufenden der Funktion über den Befehl „return“ ein Objekt zurückgeben. Der Aufruf erfolgt über den Funktionsnamen, gefolgt von den Parametern in Klammern.

```
> berechneMean <- function(p_vector){
+     l_result  <- mean(p_vector)
+     return(l_result)
+ }


> berechneMean( c(4,6,7,8,3,6,10))
[1] 6.285714
```

Abbildung 6: R-Funktionen stellen das Gegenstück zu SAS-Makros dar. Die Zuweisung erfolgt über das Schlüsselwort „function.“ Die benamte Funktion kann anschließend mit Parametern gefüllt aufgerufen werden.

4.2 Performancevergleich bei großen Datenmengen

Um einen Vergleich der Performance bei großen Datenmengen zu erzielen, wurden vergleichbare Operationen mit dem SAS- und dem R-System durchgeführt. Die Berechnungen wurden mit SAS 9.2 und R-2.11.1 auf einem Intel Core Duo CPU mit 2.53 GHz und 2GB RAM durchgeführt.

Dazu wurde ein Datensatz durch Konkatenieren von zwei Datensätzen erzeugt. Anschließend wurde dieser Datensatz bzgl. einer Spalte sortiert und in einem dritten Verarbeitungsschritt eine Selektion bezogen auf eine Spalte durchgeführt. Der Pseudocode dazu lautet:

```
start = Systemzeit;

Erzeuge Daten durch konkatenieren
        von Daten1 und Daten2;

Sortiere Daten nach Spalte1;

Selektiere * wo SpalteXY = „ABCD“;

ende  = Systemzeit;

dauer = ende – start;
```

Als Datengrundlage wurde der in der SAS-Bibliothek „SasHelp“ abgelegte Datensatz „Zipcode“ verwendet (512 Byte/Zeile). Der Datensatz wurde zunächst auf 10,000 Zeilen (ca. 5 MB) reduziert und anschließend die in dem Pseudocode dargestellten Schritte durchgeführt. Der durch das Konkatenieren erzeugte Datensatz (20,000 Zeilen / 10 MB) wurde als Grundlage für den nächsten Durchlauf eingesetzt. Dies wurde bis zu einer Dateigröße von 10 GB mit ungefähr 20 Millionen Zeilen wiederholt, siehe Tabelle 1.

Um die Performance von SAS zu optimieren, kamen drei SAS-Optionen in unterschiedlicher Kombination zum Einsatz.

a) **tagsort**:
   - Speichert nur die BY-Variablen und die Anzahl der Beobachtungen in der temporären Datei.
   - Benötigt weniger temporären Speicherplatz.
   - Laufzeit ist in der Regel höher.

b) **compress**:
   Beobachtungen in einem neu erzeugten Datensatz werden durch das sogenannte „Run Length Encoding“ Verfahren komprimiert. Dabei werden wiederkehrende Zeichen in alphanumerischen Variablen gelöscht und durch die Anzahl der Wie-

derholungen ersetzt, z.B. wird der String „AAAABBBCC“ komprimiert als „4A3B2C“ dargestellt.)

c) **memlib**:
Die angegebene Bibliothek wird im Arbeitsspeicher prozessiert. Die Daten werden also bis zum Beenden von SAS im Arbeitsspeicher zur Bearbeitung vorgehalten.

Tabelle 1: Darstellung der erzeugten Datentabellen mit der Anzahl der Beobachtungen und die sich daraus ergebende Dateigröße in MB.

Anzahl Beobachtungen	Dateigröße [MB]
20,000	10.24
40,000	20.48
80,000	40.96
160,000	81.92
320,000	163.84
640,000	327.68
1,280,000	655.36
2,560,000	1,310.72
5,120,000	2,621.44
10,240,000	5,242.88
20,480,000	10,485.76

Um die Auswirkung dieser SAS-Optionen auf die Performance genauer untersuchen zu können, wurden Kombinationen gebildet, siehe Tabelle 2.

Tabelle 2: Eingesetzte SAS-Optionen tagsort, compress, memlib für die Durchführung einer Performanceoptimierung. Die Optionen wurden in den abgebildeten Kombinationen eingesetzt.

	tagsort	compress	memlib
Kombination 1			
Kombination 2	X		
Kombination 3		X	
Kombination 4	X	X	
Kombination 5	X	X	X

Die Berechnungsdauer der einzelnen Bearbeitungsschritte wurde aufgezeichnet und anschließend über der Größe des Datensatzes [GB] aufgetragen, siehe Abbildung 7.

Die in Abbildung 7 dargestellten Ergebnisse des Performancevergleichs zwischen SAS und R zeigen auf, dass R (rote Linie) gegenüber SAS ohne die Optionen tagsort, compress und memlib (dunkelblaue Linie) einen erheblichen Geschwindigkeitsvorteil besitzt. Durch Kombination der Optionen targsort und compress (hellblaue Linie) ver-

kürzt sich die Berechnungsdauer von SAS beträchtlich und kommt durchaus in die Nähe der Berechnungsdauer von R. Das größte Optimierungspotential zeigt sich durch die gleichzeitige Verwendung der Optionen tagsort, compress und memlib (orangene Linie). Hier kann SAS mit einer hohen Berechnungsgeschwindigkeit punkten und zeigt annähernd die gleiche Performance wie R.

Sowohl R als auch SAS (mit tagsort, compress und memlib) brechen ab einer bestimmten Datengröße die Berechnung ab. Dies liegt darin begründet, dass in beiden Fällen die Daten im Arbeitsspeicher vorgehalten werden und dadurch eine physikalische Limitierung durch die Größe des Arbeitsspeichers vorgegeben ist.

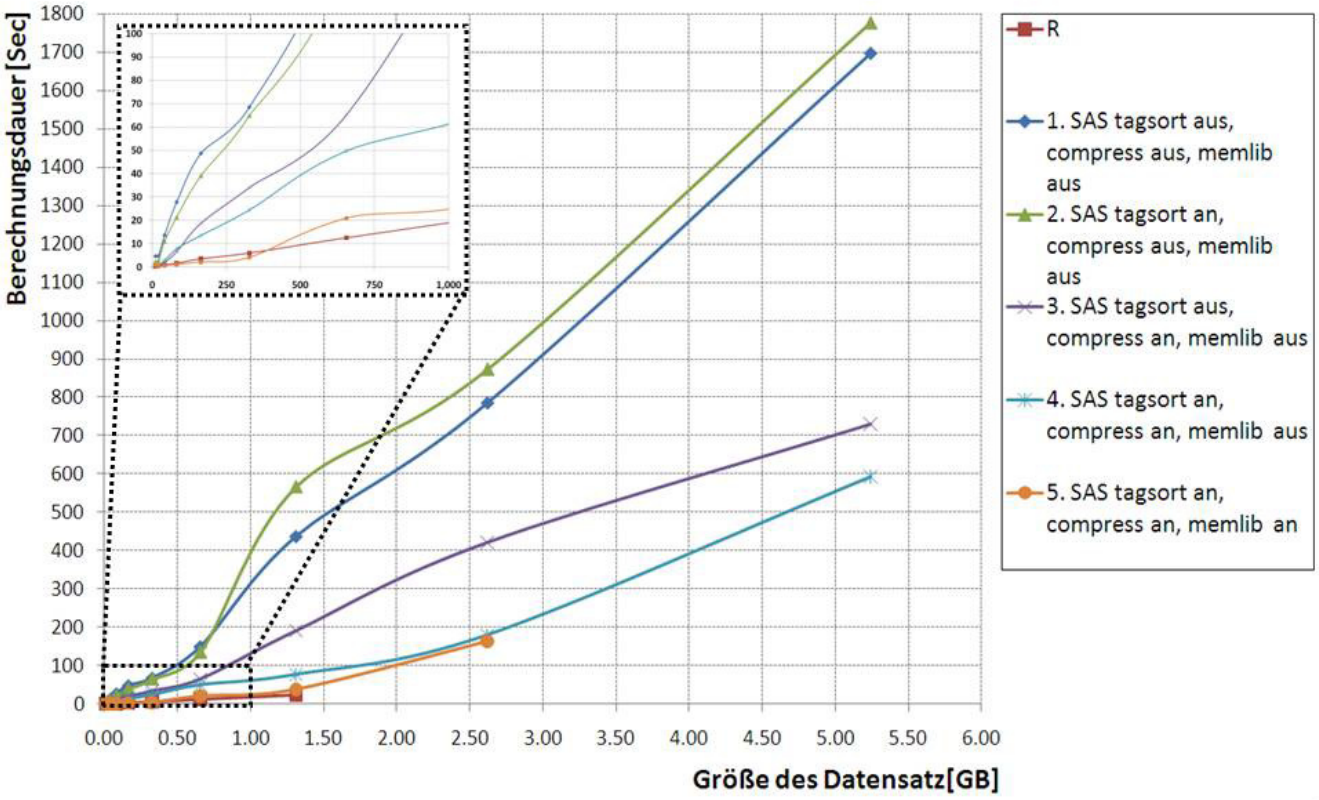

Abbildung 7: Performancevergleich zwischen SAS und R. SAS wurde mit den unterschiedlichen Kombinationen aus den Optionen tagsort, compress und memlib eingesetzt. Der Achsenabschnitt bis 1 GB und einer Berechnungsdauer von 100 Sekunden wurde vergrößert dargestellt.

Es lässt sich zusammenfassen, dass unter den vorgegebenen Bedingungen die Berechnungsdauer von SAS und R nahezu gleich ist. SAS hat bei großen Datenmengen die Nase vorne, da hier ohne die memlib-Option die Daten sequentiell von der Festplatte gelesen, bearbeitet und wieder geschrieben werden. Dadurch ergibt sich keine Limitierung durch die Größe des Arbeitsspeichers. Der Fairness wegen sei gesagt, dass R zwar die Daten im Arbeitsspeicher vorhält, aber durch freiverfügbare Zusatzpakete (z.B. Package „ff“) die Möglichkeit gegeben ist, Daten außerhalb von R zu bearbeiten.

5 Schnittstellen zwischen SAS und R

Schnittstellen zwischen SAS und R und umgekehrt bieten die Möglichkeit, auf die Funktionalitäten des jeweils anderen Systems zu zugreifen. Dadurch ergibt sich eine Vielzahl an Kombinationsmöglichkeiten der beiden Systeme.

5.1 Schnittstelle SAS IML zu R

Mit SAS IML 9.22 bietet SAS eine umfangreiche Schnittstelle zu R an. Die Lizenz für SAS IML 9.22 muss zusätzlich erworben werden. Um die Schnittstelle aktuell (März 2011) nutzen zu können, ist es auf Grund von veränderten Verzeichnisstrukturen nötig eine ältere Version von R zu installieren (z.B. R-2.10.1).

Nach dem SAS mit der RLANG Option gestartet wurde, steht dem Anwender eine leistungsstarke Schnittstelle zu R zur Verfügung. Diese Schnittstelle wurde in die Prozedur PROC IML eingebettet. Innerhalb des Startbefehls SUBMIT / R und des Endbefehls ENDSUBMIT kann beliebiger R-Code platziert werden, siehe Abbildung 8.

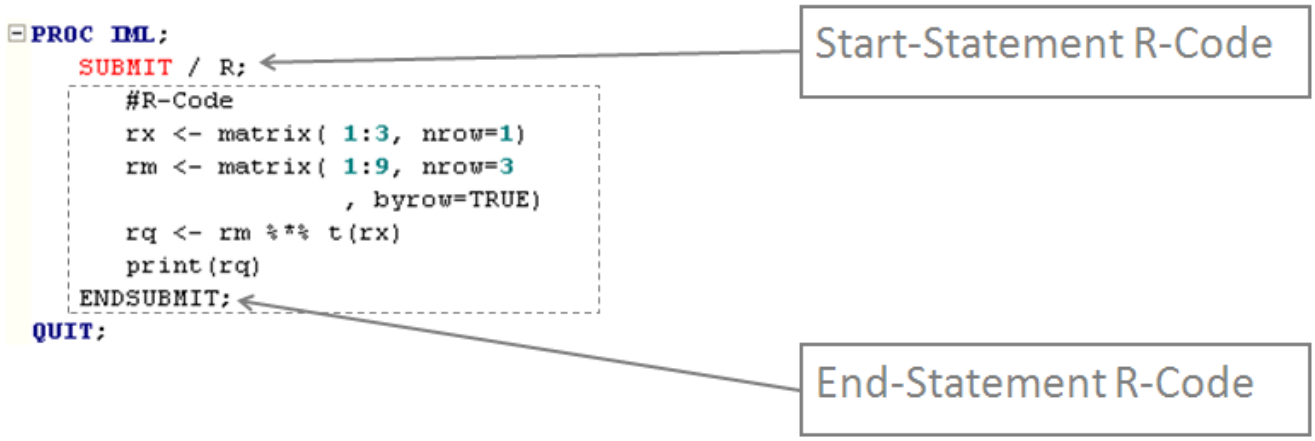

Abbildung 8: Schnittstelle PROC IML zu R. Zwischen die Start und End-Statements kann beliebiger R Code eingebettet werden.

Darüber hinaus bietet diese Schnittstelle den komfortablen Datenexport/ -Import von Matrizen und Dataframes zwischen SAS und R über den Befehl „ExportDataSetToR“ bzw. „ImportDataSetFromR“ an. Diese Funktionalität ist näher in Abbildung 9 beschrieben.

Datenexport von SAS zu R:

Subroutine	SAS Quelle	R Ziel-Objekt
ExportDataSetToR	SAS data set	R data frame
ExportMatrixToR	SAS/IML matrix	R matrix

```
PROC IML;
    RUN ExportDataSetToR("Sashelp.Class", "df" );
    submit / R;
      names(df)
    endsubmit;
QUIT;
```

Datenimport von R zu SAS:

Subroutine	R Ziel-Objekt	SAS Quelle
ImportDataSetFromR	R expression	SAS data set
ImportMatrixFromR	R expression	SAS/IML matrix

Abbildung 9: PROC IML als Schnittstelle für Datenexport /-Import von Matrizen und Dataframes. Ein SAS Data Set wird mit dem Befehl „ExportDataSetToR“ („Sashelp.class“, „df“) zu R (df ← Name des R Data Frames) exportiert.
Der umgekehrte Fall kann mittels des Befehls „ImportDataSetFromR“ realisiert werden. Dabei wird ein R Data Frame in Form eines SAS Data Sets an SAS zurückgegeben.

5.2 Schnittstelle von R zu SAS

R implementiert keine vergleichbare Schnittstelle zu SAS. Jedoch ermöglicht die Anwendung von Systembefehlen, dass SAS-Programme im Batch-Modus ausgeführt werden können. Exportiere Daten z.B. als csv-Dateien können anschließend in R eingelesen werden, siehe Abbildung 10.

```
> system(
+          "C:\\Programme\\SAS\\SASFoundation\\9.2\\sas.exe
+           C:\\HMS\\Projekte\\KSFE2011\\ProcMeansExport.sas"
+       )

> sasData <- read.table(
+            file = "C:\\tmp\\Results.csv"
+          , sep = ",", header = TRUE )
```

Abbildung 10: Schnittstelle R zu SAS über System-Befehle. Zunächst wird ein Systembefehl abgesetzt, der zum Einen auf eine SAS.exe verweist und zum Anderen auf ein aufzurufendes SAS-Programm.

Für den Datenimport von SAS Transport Files (XPORT) stehen ebenfalls die R-Funktion „read.xport“ zur Verfügung.

6 Fazit

Bei dieser Gegenüberstellung zwischen SAS und R als ungleiches Paar wurde die SAS Plattform mit der großen Fülle an Frontends, Servern und ihrer Metadatenverwaltung ausgeklammert. Ebenso wurde nicht auf die große Anzahl an Paketen, Serverkomponenten und Frontends usw. von R eingegangen. Ziel dieses Beitrags war die wertfreie Gegenüberstellung der SAS Language und der R-Basisversion.

Unterschiede ergeben sich aus den Entstehungsgeschichten der beiden Produkte. SAS hat sich zu einer kommerziellen Software entwickelt und R wird freiverfügbar zum Herunterladen bereit gestellt. Weitere Unterschiede zeigen sich bei der Performance bei großen Datenmengen. SAS mit Vorteilen bei großen Datensätzen und Optimierungsmöglichkeiten, um in den Geschwindigkeitsbereich von R zu kommen.

Ähnlichkeiten sind bei den Entwicklungsumgebungen erkennbar. Die SAS-Fensterumgebung wie auch die RGui bieten einen Editor zum Erstellen von Programmen an und dokumentieren die durchgeführten Schritte in einem LOG. Weitere Gemeinsamkeiten ergeben sich bei der Kapselung von wiederverwendbarem Code. SAS hat dies mit der SAS-Makrosprache gelöst und R mit den R-Funktionen.

Gemeinsam ist SAS und R eine Schnittstelle, wodurch die Funktionalitäten des anderen zur Verfügung gestellt werden. SAS IML 9.22 mit der Prozedur PROC IML ist eine leistungsstarke Lösung, um R Code auszuführen. R hingegen implementiert keine entsprechende Schnittstelle zu SAS. Durch Systembefehle lassen sich aber SAS-Programme ausführen und gespeicherte Datensets einlesen.

Da dieser Beitrag eine neutrale Betrachtung der Eigenschaften von SAS und R darstellt, wurde auch keine Bewertung nach Kriterien wie „besser“ oder „schlechter“ abgegeben. Ebenfalls erhebt dieses Manuskript keinen Anspruch auf Vollständigkeit. Es wurden wesentliche Unterschiede, Ähnlichkeiten und Gemeinsamkeiten betrachtet und sind als Entscheidungshilfen bei der Wahl zwischen SAS oder R bzw. für die integrative Nutzung über Schnittstellen von SAS zu R und umgekehrt zu verstehen.

Literatur

[1] SAS Institute Inc. *SAS History*. Abgerufen am 09. Februar 2011 von http://www.sas.com/company/about/history.html#s1=1

[2] Ross Ihaka, R, *Past and Future History.* A Draft of a Paper for Interface 1998.