Synergien einer Verbindung von SAS und R

Luzia Tinten
viadee
Unternehmensberatung AG
Anton-Bruchhausen-Straße 8
48147 Münster
luzia.tinten@viadee.de

Zusammenfassung

Die gängigen Statistik-Programm-Pakete haben unterschiedliche Stärken und Schwächen und unterscheiden sich in ihrer Handhabung. Meistens geht es darum, für welches System man sich entscheidet. Eher selten wird diskutiert, wie sich Funktionalität verschiedener Pakete kombinieren lässt. In diesem Beitrag geht es um die Verknüpfung von SAS mit dem Open Source Statistik-Paket R. Diskutiert werden zum einen die Gründe für ein Zusammenwirken von SAS und R, zum anderen aber auch die technischen Ansätze und Möglichkeiten, beide Pakete zu verbinden.

Schlüsselwörter: R, SAS/IML, PROC IML, SUBMIT/R, ExportDataSetToR, ImportDataSetFromR, ggplot2, Hmisc, sas7bdat, SASxport, foreign

1 Einleitung

Bei der Entscheidung für eine Software spielen neben den eigentlichen Funktionalitäten Dinge wie Sicherheit, Support sowie Kosten und Lizenzgebühren eine Rolle. Während es Settings gibt, in welchen es sinnvoll ist, sich für ein Paket zu entscheiden, sind Situationen denkbar, in welchen eine Kombination unterschiedlicher Pakete zielführend ist. So kann es Funktionen geben, die im bislang verwendeten Paket nicht vorhanden sind, oder in einem anderen Paket effizienter implementiert sind. Bei der Zusammenarbeit mehrerer Parteien, welche standardmäßig verschiedene Software verwenden, kann die Kombination der Software-Pakete zielführender sein, als das Einigen auf eine Software. Im klassischen Business Intelligence Umfeld findet häufig SAS Anwendung, während unter Data Scientists oft R das Mittel der Wahl ist. Eine Zusammenarbeit der Data Scientists mit dem BI-Bereich ist in vielen Situationen notwendig. Hier kann die Kombination von SAS und R eine sinnvolle Lösung darstellen. Darüber hinaus spielt der individuelle Kenntnisstand eine entscheidende Rolle. Ein Anwender einer Software möchte z.B. Funktionen eines anderen Software-Pakets nutzen, ohne die gesamte Ausführung mit dieser Software erlernen zu müssen.
Im Folgenden geht es um die Kombination von SAS mit dem Open Source Statistik-Paket R aus Sicht eines SAS Anwenders. Diskutiert werden deshalb Verfahren zum Datenaustausch zwischen SAS und R sowie das Nutzen der R-Funktionalität von SAS aus.

Mit SAS/IML bietet SAS eine Schnittstelle zu R. Es lässt sich R-Funktionalität innerhalb von SAS nutzen und die Schnittstelle bietet eine komfortable Möglichkeit des Datentransfers zwischen SAS und R.
In Abschnitt 2 gibt es zunächst eine kurze Vorstellung des Statistik-Pakets R, in den Abschnitten 3 und 4 folgen alternative Methoden zur Datenübertragung von SAS nach R bzw. R nach SAS. In Abschnitt 5 werden die Möglichkeiten von SAS/IML bei der Kombination von SAS mit R vorgestellt. Das Potenzial, R-Code innerhalb von SAS aufzurufen, wird anhand eines Beispiels zur Datenvisualisierung demonstriert. Es folgt in Abschnitt 6 eine Diskussion der gewonnenen Erkenntnisse.

2 R

R (www.r-project.org) ist eine populäre Open Source Statistik-Umgebung und Programmiersprache. R wurde in den 1990er Jahren von Robert Gentleman und Ross Ihaka an der University of Auckland entwickelt und basiert auf der Programmiersprache S. Zunächst lediglich für den Einsatz in der Lehre vorgesehen, wurde die Software schließlich öffentlich zur Verfügung gestellt und erfreut sich großer Beliebtheit. Der Fokus der Software liegt auf statistischen Berechnungen und der Erstellung von Grafiken. R kann durch eine Vielzahl freiverfügbarer Pakete nach Bedarf erweitert werden. Diese Pakete werden von einer weltweit breiten Community auf unterschiedlichen Plattformen entwickelt und zur Verfügung gestellt. Somit ergibt sich ein sehr großer Funktionsumfang, mit einer immensen Anzahl an Paketen für Datenzugriffe, Datenmanipulation, Datenanalysen und insbesondere Datenvisualisierung. Die Software wird kontinuierlich ausgebaut; neue statistische Methoden werden schnell implementiert. Mit RStudio existiert zudem eine komfortable Entwicklungsumgebung, die lokal oder in einer Client-Server-Installation über den Webbrowser genutzt werden kann. Auch wenn die Wurzeln von R im akademischen Umfeld liegen, so nimmt die Bedeutung von R im professionellen Unternehmenskontext ständig zu. R ist mittlerweile eines der führenden Werkzeuge von Data Scientists in Unternehmen.
Dieser Beitrag stellt keine Einführung in die Programmiersprache R dar und es sollen hier lediglich die für das Verständnis notwendigen Voraussetzungen geschaffen werden. Das folgende Hallo-Welt-Programm gibt einen Einblick in die grundlegende Syntax.

```
> myString <- "Hello, world!"
> print (myString)
[1] "Hello, world!"
```

Im ersten Statement wird mit dem Zuweisungsoperator `<-` der Variablen `myString` der Wert `Hello, world!` zugewiesen. Im nächsten Statement wird eine Funktion `print()` aufgerufen und es folgt die Ausgabe des zugewiesenen Wertes. Es gibt verschiedene Datentypen, standardmäßig `numeric`, `character` und `logical`. Darüber hinaus gibt es in R den Datentyp `factor`. Dieser Datentyp ist bei der ersten Verwendung nicht sehr intuitiv, erleichtert aber viele Auswertungen. Variablen vom Typ `factor` können nur bestimmte Ausprägungen eines Merkmals annehmen z.B. männ-

lich, weiblich oder divers bei der Geschlechtszugehörigkeit. Diese Ausprägungen werden als `levels` bezeichnet. Der R Data Frame, häufig mit `df` abgekürzt, entspricht dem SAS Dataset.
R überzeugt besonders in der Datenvisualisierung. Hervorzuheben ist hier das bekannte Grafik-Paket `ggplot2`. Es wurde von Hadley Wickham entwickelt und basiert auf der Grammatik von Grafiken. Mithilfe von `ggplot2` lassen sich elegante und sehr komplexe Grafiken erstellen. Die Funktion `ggplot()` startet die Grundebene einer Grafik.

```
> a <- ggplot(wdata, aes(x = weight))
```

Mit + können Ebenen oder Elemente eingefügt werden. So lassen sich komplexe Grafiken schrittweise aufbauen.

```
> a + geom_density(aes(fill = sex), alpha=0.4)
```

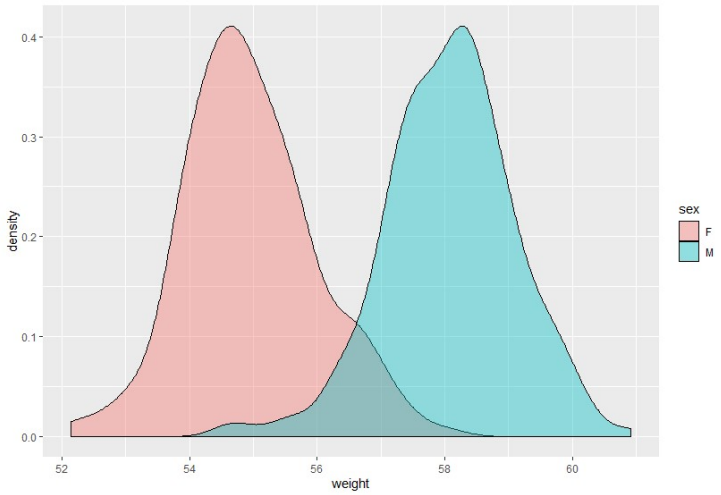

Abbildung 1: Aufbau einer Grafik mit dem R Paket `ggplot2`

3 Datentransfer von SAS nach R

Prinzipiell gibt es eine Vielzahl an Wegen, Daten zwischen den Systemen auszutauschen. Neben der generischen Lösung in Form einer gemeinsamen Datenbank gibt es weitere Methoden des Datenaustausches. Zunächst geht es um den Transfer von Daten aus SAS nach R. Dies kann dann sinnvoll sein, wenn Daten gemeinsam genutzt werden oder eine Vorverarbeitung der Daten in SAS stattfindet, bevor diese in R weiterverarbeitet werden. Im Folgenden werden verschiedene Wege vorgestellt, Daten von SAS nach R zu transferieren. Jede Methode hat ihre Vor- und Nachteile. Entscheidend ist vor allem der Kontext. Geht es darum, einen Weg zu finden, routinemäßig Daten zwischen beiden Systemen auszutauschen oder geht es um einen einmaligen Datentransfer von SAS nach R? Soll die von SAS nach R übertragene Datei letztendlich wieder nach SAS zurücktransferiert werden oder ist die Übertragung nur in eine Richtung notwendig? Zusätzlich spielen individuelle Faktoren wie die jeweilige Infrastruktur und der Kenntnisstand der Anwender eine Rolle. Es lässt sich demnach nicht allgemeingültig beantworten, welches die beste Methode ist.

3.1 Transfer über CSV-Dateien

Die erste Methode eignet sich nicht nur für SAS und R, sondern allgemein für den Datentransfer zwischen unterschiedlichen Systemen. In SAS werden die Daten mit `PROC EXPORT` exportiert

```
PROC EXPORT DATA=sashelp.cars
 OUTFILE="C:/Documents/KSFE 2020/cars.csv"
 DBMS=CSV;
RUN;
```

und in R mit der Funktion `read.csv()` (oder der Funktion `fread()` aus dem Paket `data.table` als Alternative für große Datenmengen) eingelesen.

```
> df_cars <- read.csv("C:/Documents/KSFE 2020/cars.csv")

> head(df_cars,n=5)

   Make           Model  Type Origin DriveTrain    MSRP Invoice
1 Acura             MDX   SUV   Asia        All $36,945 $33,337
2 Acura RSX Type S 2dr Sedan   Asia      Front $23,820 $21,761
3 Acura         TSX 4dr Sedan   Asia      Front $26,990 $24,647
4 Acura          TL 4dr Sedan   Asia      Front $33,195 $30,299
5 Acura      3.5 RL 4dr Sedan   Asia      Front $43,755 $39,014
  EngineSize Cylinders Horsepower MPG_City MPG_Highway Weight
1        3.5         6        265       17          23   4451
2        2.0         4        200       24          31   2778
3        2.4         4        200       22          29   3230
4        3.2         6        270       20          28   3575
5        3.5         6        225       18          24   3880
  Wheelbase Length
1       106    189
2       101    172
3       105    183
4       108    186
5       115    197
```

Die Funktion `head(data.frame, n=)` zeigt die obersten n Datensätze eines Data Frames an, offensichtlich wurde die aus SAS exportierte CSV-Datei korrekt in R eingelesen. Auch wenn die Methode auf den ersten Blick sehr simpel und komfortabel erscheint, und vor allen Dingen kein spezielles Wissen beim Anwender erfordert, so ist in der Realität häufig Detailarbeit, wie die sinnvolle und übereinstimmende Wahl der Trennzeichen, erforderlich, damit der Datenaustausch fehlerfrei funktioniert.

3.2 Transfer über SAS Transport Files

Eine weitere Möglichkeit der Datenübertragung von SAS nach R besteht im Transfer über SAS Transport Files. Diese verwenden einen umgebungsunabhängigen Standard für die Darstellung von numerischen und character Variablen. In SAS lassen sich Transport Files mit der XPORT Engine erzeugen

```
LIBNAME xportout XPORT "C:/Documents/KSFE 2020/class.xpt";

DATA xportout.class;
  SET sashelp.class;
RUN;
```

Diese können in R eingelesen werden. Hierzu kann beispielsweise das Paket `Hmisc` verwendet werden. Das Paket enthält verschiedene Funktionen für den Datenimport, unter anderem die Funktion `SASxport.get()`. Mit dieser Funktion kann der Inhalt von SAS XPORT-Dateien gelesen werden.

```
> library(Hmisc)
> df_class <- sasxport.get("C:/Documents/KSFE 2020/class.xpt")

> head(df_class,n=5)

     name sex age height weight
1  Alfred   M  14   69.0  112.5
2   Alice   F  13   56.5   84.0
3 Barbara   F  13   65.3   98.0
4   Carol   F  14   62.8  102.5
5   Henry   M  14   63.5  102.5
```

Auch hier kann die aus SAS exportierte XPORT-Datei mithilfe der entsprechenden Funktion aus dem `Hmisc` Paket in R eingelesen werden. Während beim Transfer per CSV-Datei die Spalten Namen in ihrer Darstellung übernommen werden, sind die Spaltennamen bei der Übertragung per Transport File komplett in Kleinbuchstaben. Die Übertragung funktioniert, wobei auf R-Seite das zusätzliche Paket `Hmisc` notwendig ist. Allerdings ist das Verfahren mit SAS 6 kompatibel; Features von neueren SAS Versionen, wie zum Beispiel lange Variablennamen, werden nicht unterstützt. Um diese Methode fehlerfrei für die Datenübertragung nutzen zu können, müssen Daten in den meisten Fällen vorab angepasst werden. Dadurch entsteht ein zusätzlicher Aufwand, welcher zudem ein Fehlerrisiko birgt.

3.3 R-Paket SAS7BDAT

Neben der Möglichkeit SAS-Dateien vorab entsprechend umzuwandeln, damit diese in R eingelesen werden können, gibt es die Möglichkeit, Dateien direkt im SAS-Format SAS7BDAT zu übertragen.

```
DATA "C:/Documents/KSFE 2020/shoes";
  SET sashelp.shoes;
RUN;
```

Um Dateien im SAS-Format SAS7BDAT in R einzulesen, wird das gleichnamige R-Paket `sas7bdat` benötigt. Dieses wurde von Matt Shotwell programmiert.

```
> library(sas7bdat)
> df_shoes <- read.sas7bdat("C:/Documents/KSFE 2020/shoes.sas7bdat")

> head(df_shoes,n=5)

  Region      Product  Subsidiary Stores Sales Inventory Returns
1 Africa         Boot Addis Ababa     12 29761    191821     769
2 Africa Men's Casual Addis Ababa      4 67242    118036    2284
3 Africa  Men's Dress Addis Ababa      7 76793    136273    2433
4 Africa       Sandal Addis Ababa     10 62819    204284    1861
5 Africa      Slipper Addis Ababa     14 68641    279795    1771
```

Die Übertragung im SAS-Format funktioniert ebenfalls, Variablennamen werden hier in ihrer Darstellung übernommen. Zu beachten ist, dass das Paket `sas7bdat` mit komprimierten SAS-Dateien nicht zurechtkommt. Hier gibt es die Fehlermeldung, dass komprimierte Daten nicht gelesen werden können. Es ist demnach darauf zu achten, dass die SAS Option `COMPRESS = NO` gesetzt ist.

4 Datentransfer von R nach SAS

Auch für den umgekehrten Prozess, Daten aus R nach SAS zu transferieren, gibt es verschiedene Wege. Die Gründe für einen solchen Datenaustausch sind dieselben, wie für einen Transfer von SAS nach R. Um bei den im vorherigen Abschnitt skizzierten Szenarien zu bleiben, kann es notwendig sein, dass Daten, nachdem diese von SAS nach R übertragen und in R weiterverarbeitet wurden, zurück nach SAS transferiert werden müssen.

4.1 Transfer über CSV-Dateien

Eine Möglichkeit des Datenaustausches stellt hier ebenfalls der Transfer über CSV-Dateien dar. Mit der Funktion `write.csv()` lässt sich ein R Data Frame als CSV-Datei exportieren

```
> write.csv(df_shoes, "C:/Documents/KSFE 2020/shoes.csv")
```

und mit `PROC IMPORT` lässt sich die CSV-Datei auf gewohntem Wege in SAS importieren.

```
PROC IMPORT DATAFILE = "C:/Documents/KSFE 2020/shoes.csv";
 OUT = work.shoes
 DBMS = CSV;
RUN;

PROC PRINT DATA = shoes;
RUN;
```

Beob.	VAR1	Region	Product	Subsidiary	Stores	Sales	Inventory	Returns
1	1	Africa	Boot	Addis Ababa	12	29761	191821	769
2	2	Africa	Men's Casual	Addis Ababa	4	67242	118036	2284
3	3	Africa	Men's Dress	Addis Ababa	7	76793	136273	2433
4	4	Africa	Sandal	Addis Ababa	10	62819	204284	1861
5	5	Africa	Slipper	Addis Ababa	14	68641	279795	1771

Abbildung 2: `PROC PRINT` Ausgabe der aus R exportierten und in SAS importierten CSV-Datei shoes.csv

Der Transfer per CSV-Datei funktioniert, allerdings übernimmt die R-Funktion `write.csv()` in ihrer Default-Einstellung die Observation-Number als erste Spalte. Diese wird in SAS entsprechend als zusätzliche Spalte interpretiert. Dies lässt sich mit der Option `row.names = FALSE` verhindern.

```
> write.csv(df_shoes, "C:/Documents/KSFE 2020/shoes_02.csv",
row.names= FALSE)

PROC IMPORT DATAFILE = "C:/Documents/KSFE 2020/shoes_02.csv";
 OUT = work.shoes_02
 DBMS = CSV;
RUN;

PROC PRINT DATA = shoes_02;
RUN;
```

Beob.	Region	Product	Subsidiary	Stores	Sales	Inventory	Returns
1	Africa	Boot	Addis Ababa	12	29761	191821	769
2	Africa	Men's Casual	Addis Ababa	4	67242	118036	2284
3	Africa	Men's Dress	Addis Ababa	7	76793	136273	2433
4	Africa	Sandal	Addis Ababa	10	62819	204284	1861
5	Africa	Slipper	Addis Ababa	14	68641	279795	1771

Abbildung 3: `PROC PRINT` Ausgabe der aus R exportierten und in SAS importierten CSV-Datei shoes_02.csv

Die Methode ist insgesamt unkompliziert. Der Vorteil besteht hier (wie bei der Übertragung von SAS nach R per CSV-Datei) insbesondere darin, dass beim Anwender kein

spezielles Wissen erforderlich ist. Allerdings müssen Details wie die Default-Einstellungen der verwendeten R-Funktion oder auch die Wahl des Trennzeichens beachtet werden. Somit kann der Datentransfer in der Praxis entsprechend aufwendig werden.

4.2 Transfer über SAS Transport Files

Die Datenübertragung von R nach SAS mithilfe von SAS Transport Files ist nicht möglich. Mit der Funktion `write.xport()` aus dem R-Paket `SASxport` lassen sich zwar XPORT-Dateien schreiben,

```
> library(SASxport)
> write.xport(df_class, file="C:/Documents/KSFE 2020/class_02.xpt")
```

allerdings erkennt SAS die XPORT-Datei nicht und diese lässt sich nicht ohne Weiteres in SAS importieren.

4.3 R-Paket Foreign

Eine weitere Möglichkeit des Datentransfers bringt das R-Paket `foreign` mit. Dieses beinhaltet den Export in Datenformate anderer Statistikprogramme, unter anderem SAS. Die Funktion `write.foreign()` funktioniert so, dass jeweils zwei Dateien erzeugt werden, zum einen der zu exportierende Data Frame als Textdatei ohne Formatierung, und zum anderen eine Programmdatei, welche in diesem Fall SAS-Code enthält.

```
> library(foreign)
> write.foreign(df_class, "C:/Documents/KSFE 2020/class.txt",
"C:/Documents/KSFE 2020/class.sas", package="SAS")
```

Mit der Einstellung `package="SAS"` erzeugt R die beiden Dateien class.txt und class.sas. Die erzeugte SAS-Programmdatei class.sas hat folgenden Aufbau:

```
* Written by R;
*  write.foreign(df_class, "C:/Documents/KSFE 2020/class.txt",  ;

PROC FORMAT;
value Name
     1 = "Alfred"
     2 = "Alice"
     3 = "Barbara"
     4 = "Carol"
     5 = "Henry"
     6 = "James"
     7 = "Jane"
     8 = "Janet"
     9 = "Jeffrey"
     10 = "John"
```

```
      11 = "Joyce"
      12 = "Judy"
      13 = "Louise"
      14 = "Mary"
      15 = "Philip"
      16 = "Robert"
      17 = "Ronald"
      18 = "Thomas"
      19 = "William"
;

value Sex
      1 = "F"
      2 = "M"
;

DATA  rdata ;
INFILE  "C:/Documents/KSFE 2020/class.txt"
     DSD
     LRECL= 21 ;
INPUT
 Name
 Sex
 Age
 Height
 Weight
;
FORMAT Name Name. ;
FORMAT Sex. ;
RUN;
```

Bei Ausführung des SAS-Programms wird das Dataset rdata erzeugt, die Datenübertragung mit der Funktion `write.foreign()` funktioniert.

```
PROC PRINT DATA = rdata;
RUN;
```

Beob.	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5

Abbildung 4: `PROC PRINT` Ausgabe des im Programm class.sas erzeugten SAS-Dataset rdata

5 SAS/IML

Seit der SAS Version 9.22 gibt es mit SAS/IML eine umfassende Schnittstelle zu R. IML steht für Interactive Programming Language. Neben der Möglichkeit, R-Code direkt aus SAS auszuführen, bietet die Schnittstelle eine bequeme Möglichkeit des Datentransfers. Mit der Prozedur `PROC IML` wird SAS/IML aufgerufen und mit `QUIT` wird die Session beendet. Dazwischen folgen die entsprechenden IML-Statements.

```
PROC IML;
  IML-Statements
;
QUIT;
```

Damit die Zusammenarbeit von SAS und R mit SAS/IML funktioniert, sind gegebenenfalls vorbereitende Schritte notwendig. Zunächst muss R installiert sein. Die aktuelle Systemeinstellung lässt sich mit

```
PROC OPTIONS OPTION=RLANG;
RUN;
```

überprüfen. Im SAS Log findet sich daraufhin entweder die Option `RLANG` mit dem Hinweis, dass R unterstützt wird, oder die Option `NORLANG`, und dass das System R nicht unterstützt. Für den Fall, dass R nicht unterstützt wird, muss im SAS Configuration File die Option –RLANG entsprechend ergänzt werden. Zudem muss die R_HOME Variable im Configuration File mit dem korrekten Pfad hinterlegt sein, damit SAS auf R zugreifen kann.

5.1 Transfer über SAS/IML

Neben den vorgestellten Alternativen zur Datenübertragung lassen sich mit SAS/IML Daten von SAS nach R und ebenfalls von R nach SAS transferieren. Die Funktion `ExportDataSetToR("libref.member", "R Data Frame")` erzeugt aus einem SAS Dataset einen R Data Frame. Als Parameter werden der Funktion das entsprechende SAS Dataset als String sowie der Name des R Data Frames, in welchen die Daten exportiert werden sollen (ebenfalls als String), übergeben.

```
PROC IML;
  RUN ExportDataSetToR("sashelp.cars", "df_cars");
…
```

Mit `ImportDataSetFromR("libref.member", "R Expression")` kann ein R Data Frame in ein SAS Dataset umgewandelt werden. Hierzu wird der Funktion sowohl der Name des SAS Datasets, welches erzeugt werden soll, als auch ein R Ausdruck als String übergeben. Der R Ausdruck wird evaluiert und in einen R Data Frame verwandelt, aus diesem erzeugt die Funktion ein SAS Dataset.

```
PROC IML;
  RUN ImportDataSetFromR("work.cars_02", "df_cars_02");
…
```

In SAS gibt es Konventionen, welcher bei der Benennung von Variablen beachtet werden müssen. Falls Variablen im zugrundeliegenden R Data Frame diesen Konventionen nicht gehorchen, finden folgende Abwandlungen der ursprünglichen Variablennamen statt.

1. Variablennamen dürfen in SAS nicht mehr als 32 Zeichen lang sein, längere Namen werden abgeschnitten
2. Variablennamen dürfen in SAS nur mit A-Z, a-z oder einem Unterstrich beginnen, beginnt ein Name mit einem anderen Zeichen, so wird ein Unterstrich vor den ursprünglichen Namen gesetzt
3. Variablennamen dürfen in SAS nur die Zeichen A-Z, a-z, 0-9 und den Unterstrich enthalten, enthält ein Name ein unzulässiges Zeichen, so wird dieses mit einem Unterstrich ersetzt.

5.2 SUBMIT/R

Mit SAS/IML lässt sich außerdem R-Funktionalität innerhalb von SAS nutzen. Zwischen dem Startbefehl `SUBMIT/R` und dem Endbefehl `ENDSUBMIT` kann R-Code platziert werden. R ist im Gegensatz zu SAS case-sensitiv, dies gilt auch für R-Code, welcher innerhalb von SAS ausgeführt wird. Das folgende Beispiel demonstriert die prinzipielle Funktionsweise von `SUBMIT/R`.

```
PROC IML;
  RUN ExportDataSetToR("sashelp.class", "df_class");
  SUBMIT/R;
   df_class$neu <- "neu"
  ENDSUBMIT;
  RUN ImportDataSetFromR("class_neu", "df_class");
QUIT;

PROC PRINT DATA = class_neu;
RUN;
```

Beob.	Name	Sex	Age	Height	Weight	neu
1	Alfred	M	14	69	112.5	neu
2	Alice	F	13	56.5	84	neu
3	Barbara	F	13	65.3	98	neu
4	Carol	F	14	62.8	102.5	neu
5	Henry	M	14	63.5	102.5	neu

Abbildung 5: Ausgabe des mit `SUBMIT/R` modifizierten Datasets class_neu

Zunächst wird das SAS Dataset sashelp.class als R Data Frame df_class exportiert. Mit `SUBMIT/R` wird die R-Session gestartet. Es wird dem Data Frame df_class die Spalte

neu hinzugefügt. Mithilfe des Zuweisungsoperators wird diese Spalte mit dem character Wert „neu“ befüllt. Anschließend wird der Data Frame df_class in das SAS Dataset class_neu transferiert. Die anschließende Ausgabe per `PROC PRINT` zeigt, dass das Dataset korrekt hin- und wieder zurücktransferiert wurde und außerdem die mit R hinzugefügte Spalte *neu* vorhanden ist.

Nun lässt sich eine neue Spalte genauso mit SAS hinzufügen und der Umweg über `SUBMIT/R` ist nicht gerechtfertigt. Die meiste Funktionalität von R lässt sich auch mit SAS umsetzen (Gleiches gilt umgekehrt). Ein Bereich, in welchem R heraussticht, ist Datenvisualisierung. Am bekanntesten und umfangreichsten ist hier das bereits genannte R-Paket `ggplot2`. Hiermit lassen sich sehr komplexe und ästhetische Grafiken erstellen. Es ist denkbar, dass ein SAS Anwender die Datenvisualisierung in R nutzen möchte, umso komfortabler ist hier die Möglichkeit, den entsprechenden R-Code direkt aus SAS heraus auszuführen. Dies soll im Folgenden gezeigt werden. Verwendet wird hierzu das sashelp-Dataset cars. Erzeugt wird ein Streudiagramm (scatter plot), welches die Abhängigkeit zwischen *Horsepower* und *Invoice* abbilden soll.

```
PROC IML;
  RUN ExportDataSetToR("sashelp.cars", "df_cars");
  SUBMIT/R;
   install.packages("ggplot2")
   library(ggplot2)
   ggplot(df_cars, aes(x=Invoice,y=Horsepower)) + geom_point()
  ENDSUBMIT;
QUIT;
```

Zunächst wird das Dataset in einen R Data Frame umgewandelt. Das Paket `ggplot2` muss bei erstmaliger Nutzung installiert und geladen werden. Die Funktion `ggplot()` startet den Plot `geom_point()` erzeugt das simpelste Streudiagramm.

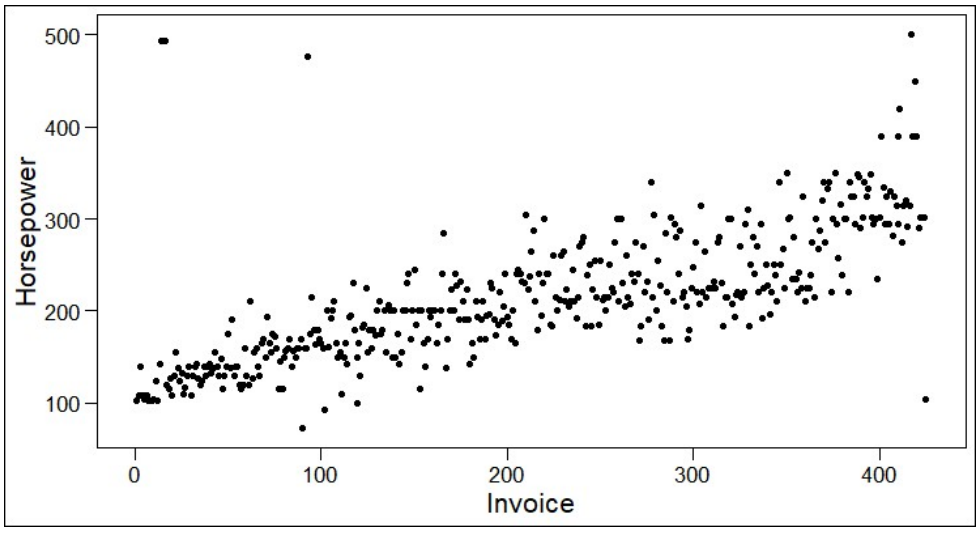

Abbildung 6: Mit `ggplot2` innerhalb von SAS erzeugtes Streudiagramm, dargestellt ist Horsepower in Abhängigkeit von Invoice

Dieser simple Plot lässt sich nun beliebig erweitern. So kann die Größe, Form und Farbe der Punkte variiert werden, es können Titel und Bezeichnungen hinzugefügt werden und es können bestimmte Bereiche hervorgehoben und Regressionslinien eingezeichnet werden. Jedes weitere Element wird der Grundfunktion mit + hinzugefügt.

```
PROC IML;
  RUN ExportDataSetToR("sashelp.cars", "df_cars");
  SUBMIT/R;
   install.packages("ggplot2")
   library(ggplot2)
   ggplot(df_cars, aes(x=Invoice,y=Horsepower,
       shape=Horsepower > 300, color=Horsepower > 300))+
   geom_point()+
   stat_ellipse()+
   labs(title="Horsepower according to the Invoice")
  ENDSUBMIT;
QUIT;
```

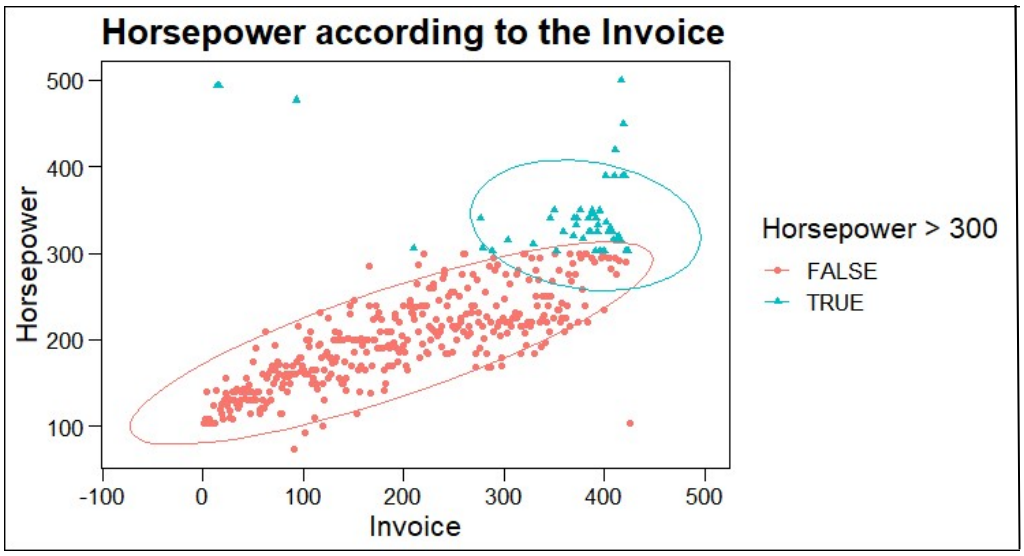

Abbildung 7: Mit `ggplot2` innerhalb von SAS erzeugtes Streudiagramm, dargestellt ist Horsepower in Abhängigkeit von Invoice

6 Diskussion

SAS und R lassen sich gut miteinander kombinieren. Es gibt verschiedene Wege, Daten zwischen den Systemen auszutauschen. Mit SAS/IML ist es sogar möglich, R-Code innerhalb von SAS aufzurufen. Daraus ergibt sich die Chance und Möglichkeit, von den Stärken beider Sprachen zu profitieren. Die vorgestellten Beispiele sind vereinfacht, in der Praxis sind weitere Details zu beachten. So haben SAS und R signifikante Unterschiede im Umgang mit Labeln, Formaten, Datentypen und Missing Values. Grundsätzlich lassen sich die allermeisten Tasks sowohl mit SAS als auch mit R lösen. Wann die Kombination von SAS und R sinnvoll ist und einen tatsächlichen Mehrwert darstellt, ist abhängig vom jeweiligen Kontext. Wie bereits eingangs erwähnt, spielen hier neben

technischen Fragen und tatsächlicher Funktionalität individuelle Faktoren, wie Erfahrung und Kenntnisse der Anwender, eine große Rolle. Bei den vorgestellten Methoden geht es deshalb lediglich um die grundsätzlichen Vorgänge *Datenaustausch zwischen SAS und R* und *Nutzen von R Funktionalität innerhalb von SAS*. Mithilfe dieser wesentlichen Bausteine lassen sich SAS und R quasi beliebig miteinander kombinieren. Es wird keine Wertung vorgenommen, wann welche Sprache generell die bessere Alternative darstellt. Auch gibt es weitere als die hier vorgestellten Wege, SAS und R gemeinsam zu nutzen.

Literatur

[1] STHDA Statistical tools for high-throughput data analysis, http://www.sthda.com/english/wiki/be-awesome-in-ggplot2-a-practical-guide-to-be-highly-effective-r-software-and-data-visualization

[2] SAS Documentation, XPORT Engine with DATA Step or PROC Copy https://documentation.sas.com/?docsetId=movefile&docsetTarget=p0ospq7g4cae73n1ucdg2x6i14li.htm&docsetVersion=9.4&locale=en

[3] SAS Documentation, Calling Functions in the R Language https://documentation.sas.com/?docsetId=imlug&docsetTarget=imlug_r_toc.htm&docsetVersion=15.1&locale=en