Multivariate Datenvisualisierung

Jörg Sellmann
Julius Kühn-Institut
Stahnsdorfer Damm 81
14532 Kleinmachnow
joerg.sellmann@julius-kuehn.de

Zusammenfassung

Die bekannte Darstellung multivariater Daten in der Ebene der ersten beiden Hauptkomponenten wird erweitert durch die dritte Dimension. Diese Würfeldarstellung wird weiter geführt zur dynamischen Grafik, in der die Objekte im Raum umherfliegen. Alternative Darstellungsmöglichkeiten sind das gemeinsame Merkmals- und Objektdisplay, die Optimierung der zweidimensionalen Hauptkomponentendarstellung oder auch die Hinzunahme weiterer Achsen.

Schlüsselwörter: Hauptkomponentendisplay, G3D, Biplot, Non-Linear-Mapping, animierte Grafiken, Y3-Achse

1 Phase 0

Ende der 80er Jahre des letzten Jahrhunderts wurde an der Humboldt-Universität zu Berlin ein Programmpaket zu ausgewählten grafischen Verfahren der multivariaten Datenanalyse entwickelt und veröffentlicht [1]. Aufbauend auf die darin veröffentlichten BASIC-Quelltexte und die zuvor in [2] vorgestellten Verfahren wurde durch den Autor ein TURBO-PASCAL Programmpaket für den 386er Prozessor entwickelt. Highlight dieses Paketes war die Möglichkeit, multivariate Datensätze in einer pseudo 3-dimensionalen Darstellung rotieren zu lassen. Die dazu notwendigen mathematischen Transformationen wurden in Ermangelung von Google und Co. anhand mathematischer Lehrbücher in die Programmiersprache umgesetzt. So umfasst allein das Rotationsprogramm ca. 1500 Zeilen selbst entwickelten und getippten Code.

2 Phase 1

Genau 25 Jahre später, im Jahr 2013 am Rande eines SAS-Kurses zur Hauptkomponentenanalyse im Rahmen der Biometriekurse des BMEL gelang es mit Hilfe der Prozedur **`g3d`** und einem **`Annotate`**-Datensatz, **beschriftete** Objekte im Raum darzustellen:

```
proc princomp data=iris out=irispca outstat=score;
var Petal_length Petal_width Sepal_length Sepal_width;
id NR;
run;
```

```
data label;
length function style $8 text $26;
retain function 'label' xsys ysys zsys '2' position '2' size 1;
set irispca;
x=prin2; y=prin1; z=prin3; text=NR;
run;

proc g3d data=irispca;
 scatter prin1*prin2=prin3 /
 anno=label shape=symbol color='black' rotate=60 tilt=30 grid;
run;
```

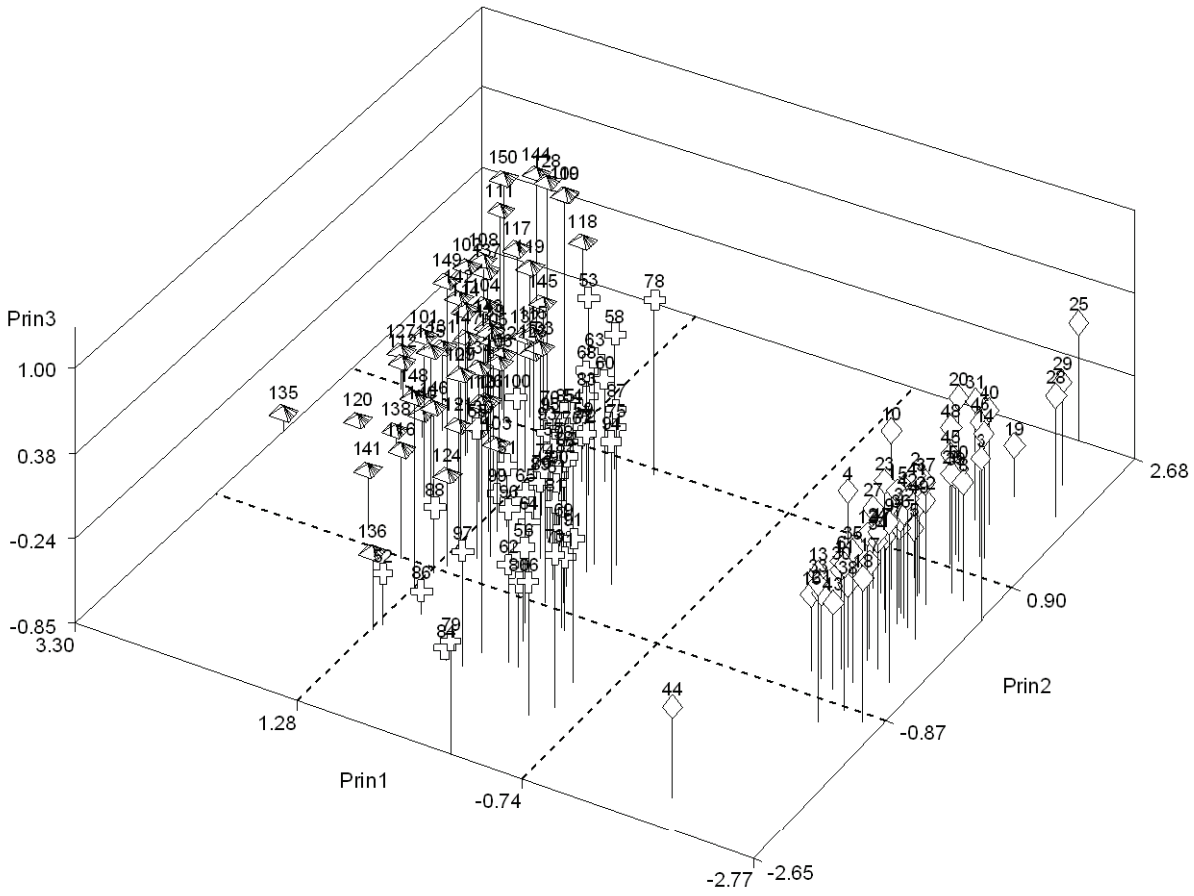

Abbildung 1: 3D-Darstellung des IRIS-Datensatzes

In Abbildung 1 ist die klare Trennung der 3 IRIS-Typen, gekennzeichnet durch unterschiedliche Symbole, zu erkennen. Offen bleiben die Fragen nach der Neigung `tilt` und dem Rotationswinkel `rotate`, um diesen besten Blick in den Datensatz zu bekommen. Hier ist Probieren angesagt. Von einer Dynamik wie 25 Jahre zuvor ist noch gar keine Rede.

Der Datastep ② ist für die weiteren Makros von zentraler Bedeutung. Er erstellt aus der Ausgabe von `proc princomp out=` den Annotate-Datensatz.

3 Phase 2

Motiviert durch den Beitrag „Animation using SGPLOT“ im Lieblings-Blog des Autors http://blogs.sas.com/content/graphicallyspeaking/2013/05/23/animation-using-sgplot/ konnte 2015 das Problem sich drehender Grafiken gelöst werden. Es entstanden mehrere Makros, die entweder den ganzen Würfel (3.1) oder nur dessen Inhalt drehen (3.2). Einmal begonnen, wurde gleich ein weiteres Uralt-Programm nach SAS portiert: Biplots (3.3). Der Umsetzung entgegen sieht das Non-Linear-Mapping (3.4). Völlig neu ist die Idee, mehr als zwei Y-Achsen in den Grafiken zu verwenden (3.5).

3.1 Drehung des Würfels

Die Drehung des ganzen Würfels ist ohne viel Aufwand zu realisieren. Als Basis nehmen wir den aus der Hauptkomponentenanalyse stammenden Datensatz, den in ② erstellten Annotate-Datensatz und führen die `proc g3d` in einem Makro mit einer vorgegebenen Winkelschrittweite `incr` aus:

```
%MACRO
 G3DCubeAll(.., giffile, gifsize, gifanim, start, end, incr, ..);
②     /* create annotate */
③     /* open gif */
  %do rotate=&start. %to &end. %by &incr.;
       proc g3d … ;
  %end;
④     /* close gif */
%MEND G3DCubeAll;
```

Die in der `%do` Schleife erstellten Grafiken werden mittels ③ und ④ in eine animierte gif-Datei geschrieben:

```
③  options papersize=&gifsize.
      printerpath=gif animation=start
      animduration=&gifanim.
      animloop=yes noanimoverlay;

   ods printer dpi=100 file=&giffile.;

④  options printerpath=gif animation=stop;
   ods printer close;
```

Je nach Geschwindigkeit `gifanim` und Auflösung `incr` erhält man eine schnelle oder langsame sowie gleichförmige oder ruckelnde Drehung. Mit `animloop=yes` erreicht man eine endlose Animation der gif-Datei. Höhe und Breite der gif-Datei werden mit dem Parameter `gifsize` definiert.

Ein Beispielaufruf für `G3DCubeAll` mit allen Parametern:

```
%G3DCubeAll(daten=ackerpca, ident=ident, giffile="ackerbau2a.gif",
    gifsize="('10 in', '8 in')", gifanim=0.2, shape=symbol,
    color="black", start=0, end=350, incr=10, tilt=30);
```

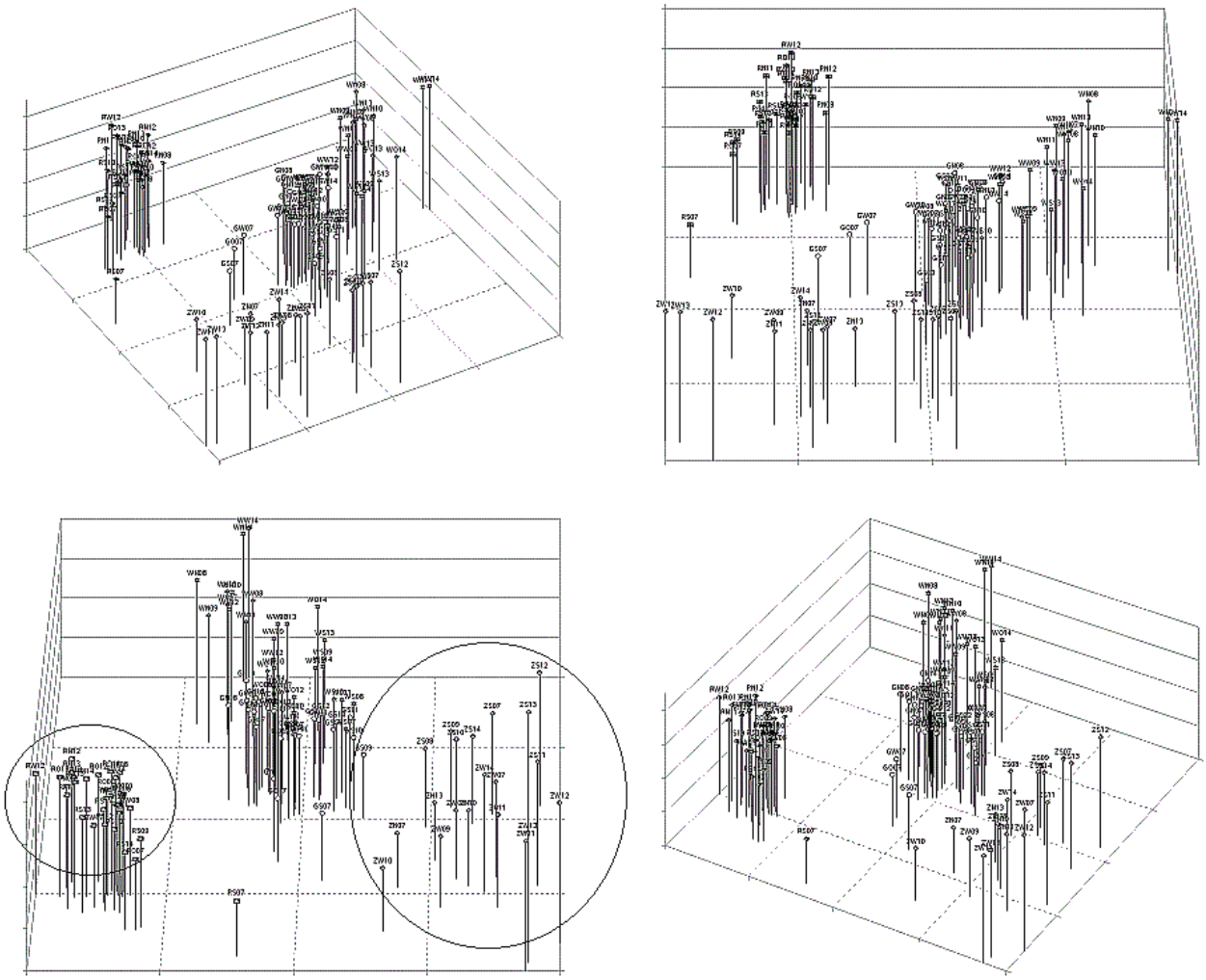

Abbildung 2: ausgewählte Phasen des Ackerbau-Datensatzes

In der (farbigen Bildschirm-)Darstellung gut zu erkennen ist die klare Trennung der Ackerbau-Kulturen Winterraps (linker Kreis), Zuckerrüben (rechter Kreis) und Winterweizen/Winterroggen (mittlere Gruppe). Das lässt den Schluss zu, dass sich diese Kulturen bzgl. der gemessenen Parameter stark unterscheiden bzw. ähneln.

3.2 Drehung im Würfel

Mehr Ruhe beim Betrachten verspricht der Ansatz, nicht den gesamten Würfel zu drehen, sondern nur dessen Inhalt. Hierzu ist etwas Mathematik gefordert, denn es müssen die neuen Koordinaten der Objekte bei jeder Drehung berechnet werden.

Gegeben sei der Spalten-Vektor $\boldsymbol{v}$ eines Objekts in der Ebene. Dann gilt für den neuen Vektor $\boldsymbol{v_1}$ bei Drehung um den Winkel α:

$$v_1 = R_\alpha * v \qquad \text{mit} \qquad R_\alpha = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$$

Der Rest sieht dann fast genauso aus:

```
%MACRO G3DCubeInt(.., start, end, incr, ..);
③     /* open gif */
      %do grad=&start. %to &end. %by &incr.;
           %drehung(.., &grad., ..);
      %end;
④     /* close gif */

%MACRO drehung (.., grad, ..);
①     /* Berechnung der neuen Koordinaten */
②     /* create annotate */
      proc g3d data=datatmp; …; run;
%MEND drehung;
```

Neu ist hier lediglich der untere Abschnitt ①, die übrigen sind mit den vorhergehenden nahezu identisch.

```
data datatmp;
set &daten. end=last;
pi=constant('pi');

/* Rotationsmatrix --> Rotation um grad° */
prin1n=cos(pi*&grad./180)*prin1+sin(pi*&grad./180)*prin2;
prin2n=cos(pi*&grad./180)*prin2-sin(pi*&grad./180)*prin1;
prin3n=prin3;
shape=&shape.;
output;

if last;    /* Generiere zwei 'extreme' Beobachtungen */
prin1n=&y_min.;  prin2n=&x_min.;  prin3n=&z_min.;  shape='Point';
&color.='gray'; &ident.='20'x; output;

prin1n=&y_max.;  prin2n=&x_max.;  prin3n=&z_min.;  shape='Point';
&color.='gray'; &ident.='20'x; output;
run;
```

Zu beachten ist die Erstellung zweier „unsichtbarer“ extremer Beobachtungen. Diese bewirken, da in `proc g3d` keine min- und max-Werte festgelegt werden können, dass die Punktwolke immer in einem Würfel mit den gleichen Ausmaßen liegt. Ohne diese Dummy-Objekte würde man ein Zittern der Punkwolke erhalten.

Ein Beispielaufruf mit dem Ackerbaudatensatz für `G3DCubInt` ist:

```
%G3DCubeInt(daten=ackerpca, ident=ident, giffile="ackerbau1a.gif",
     gifsize="('10 in', '8 in')", gifanim=0.2, color="black",
     start=0, end=350, incr=10, rotate=170, tilt=45, shape=symbol
);
```

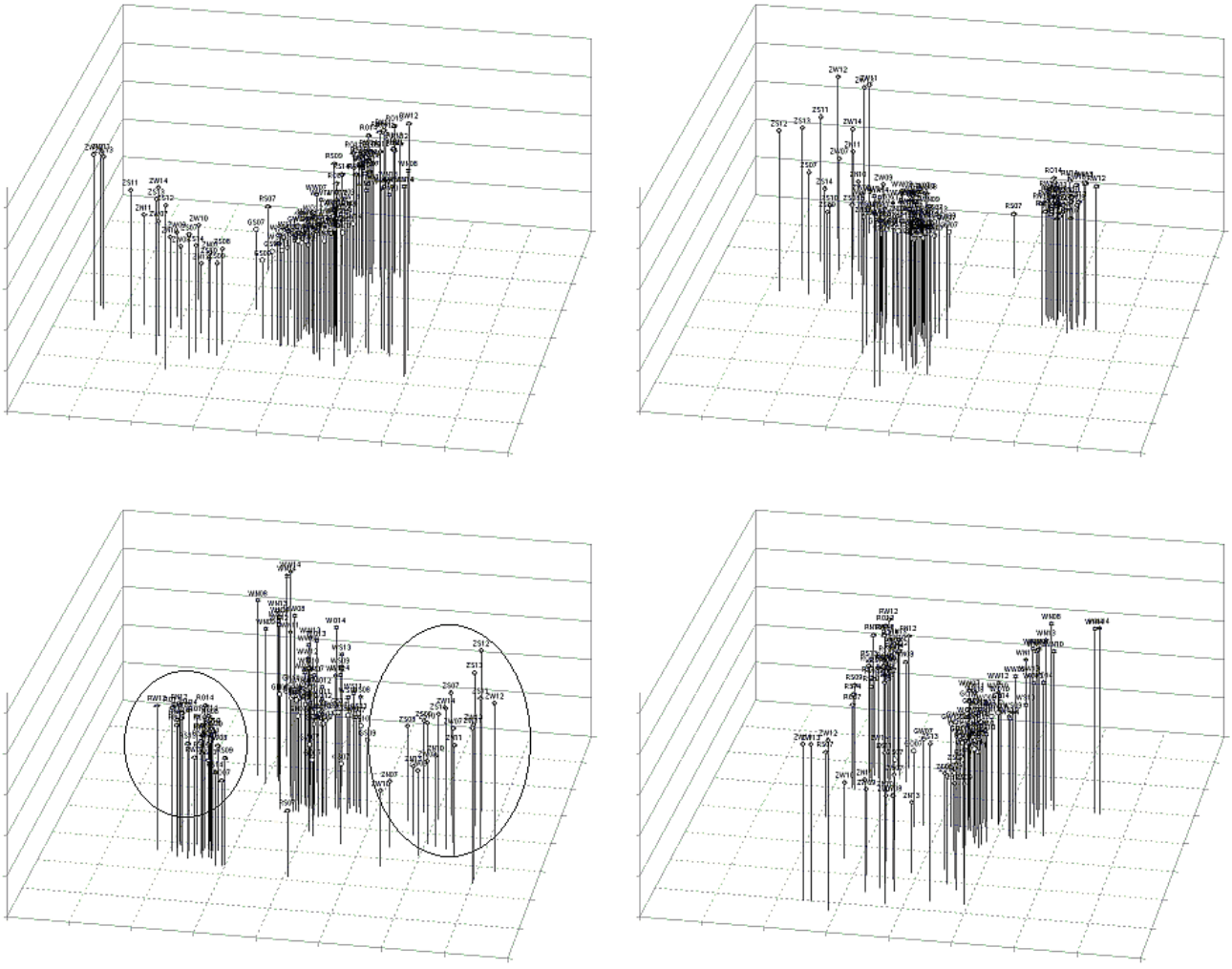

Abbildung 3: ausgewählte Phasen des Ackerbau-Datensatzes

Auch hier ist wieder die klare Trennung der Ackerbau-Kulturen Winterraps (linker Kreis), Zuckerrüben (rechter Kreis) und Winterweizen/Winterroggen (mittlere Gruppe) zu erkennen.

Beide Makros sind so gehalten, dass eine Kennzeichnung von Gruppen anhand von Farben und/oder Symbolen in den Grafiken visualisiert werden kann. Über den Aufrufparameter `color` kann entweder eine Farbe direkt (`color='black'`) oder eine Spalte,

die die Farbe für jedes Objekt enthält (`color=Farbe`), übergeben werden. Analog verhält es sich mit dem Parameter `symbol` für die Shapeauswahl.

Als Ausgang für die beiden Makros `G3DCubInt und G3DCubAll` dient ein vorher mittels Hauptkomponentenanalyse erstellter Datensatz, da in den Makros auf die Spalten `prin1`, `prin2` und `prin3` Bezug genommen wird.
Das **primäre** Ziel für beide Makros ist die Bildschirmdarstellung, um, visuell unterstützt durch die Drehung, Muster in den Daten zu erkennen. Eine Print-Darstellung kann ohnehin nur für Einzelbilder erfolgen. Hier im Script soll mit den bewusst kleinen schwarz-weiß Abbildungen 2 und 3 das Ergebnis nur angedeutet werden.

3.3 Biplots

In den Kapiteln 3.1 und 3.2 haben wir eine Möglichkeit gesehen, Muster in den Daten zu erkennen. Es drängt sich sofort die Frage nach der Ursache der Muster auf. Können wir eine Zuordnung von einzelnen Variablen zu bestimmten Gruppen finden? Eine nahe liegende Antwort heiß Faktoranalyse. Doch was ist, wenn man die Faktoren schlecht oder gar nicht interpretieren kann.

Die Variablendarstellung der Hauptkomponentenanalyse ermöglicht einen alternativen multivariaten Zugang. Denn wie sich die Objekte als n Punkte im p-dimensionalen (Variablen-)Raum auffassen lassen, stellen die Variablen p Punkte im n-dimensionalen (Objekt-)Raum dar. Werden die Daten standardisiert und anschließend außerdem durch $\sqrt{n-1}$ geteilt, so gelten folgende Aussagen:

- die Variablen haben im $\mathbf{R}^n$ den Abstand 1 vom Nullpunkt,
- der Kosinus des Winkels, den die Verbindungsstrecken zweier Variablen zum Nullpunkt miteinander bilden, entspricht gerade dem Korrelationskoeffizienten dieser Variablen.

Um die Verhältnisse des $\mathbf{R}^n$ sichtbar zu machen, wird man wiederum eine Darstellung in der von den beiden ersten Hauptkomponenten aufgespannten Ebene anstreben. Die Koordinate der *i*-ten Variable ($i = 1, \ldots, p$) bezüglich der *j*-ten Hauptkomponente ($j = 1, 2$) des Merkmals *m* ist

$$m_{ij} = e_{ij}\sqrt{\lambda_j}$$

wobei e_{ij} die *i*-te Komponente des *j*-ten Eigenvektors und λ_j der *j*-te Eigenwert ist.

Stellt man die Objekte und die transformierten Variablen gemeinsam in der Ebene dar, erhält man die genannten Biplots.

In **R** sind die Biplots in der Standardinstallation enthalten:

```
> biplot(princomp(ackerbau[,c('H','I','F','W')],cor=TRUE))
```

Für SAS gibt es mehrere Lösungen. Die „eingebaute“ Lösung heißt **M**ultidimensional **P**reference **A**nalysis. Bei eingeschalteter ODS-Graphics erhält man als Nebenprodukt die gewünschte Grafik:

```
ods graphics on;
proc prinqual plots=MDP MDPREF=1.5 maxiter=100;
 transform identity(H I F W);
 id ident kultur;
run;
ods graphics off;
```

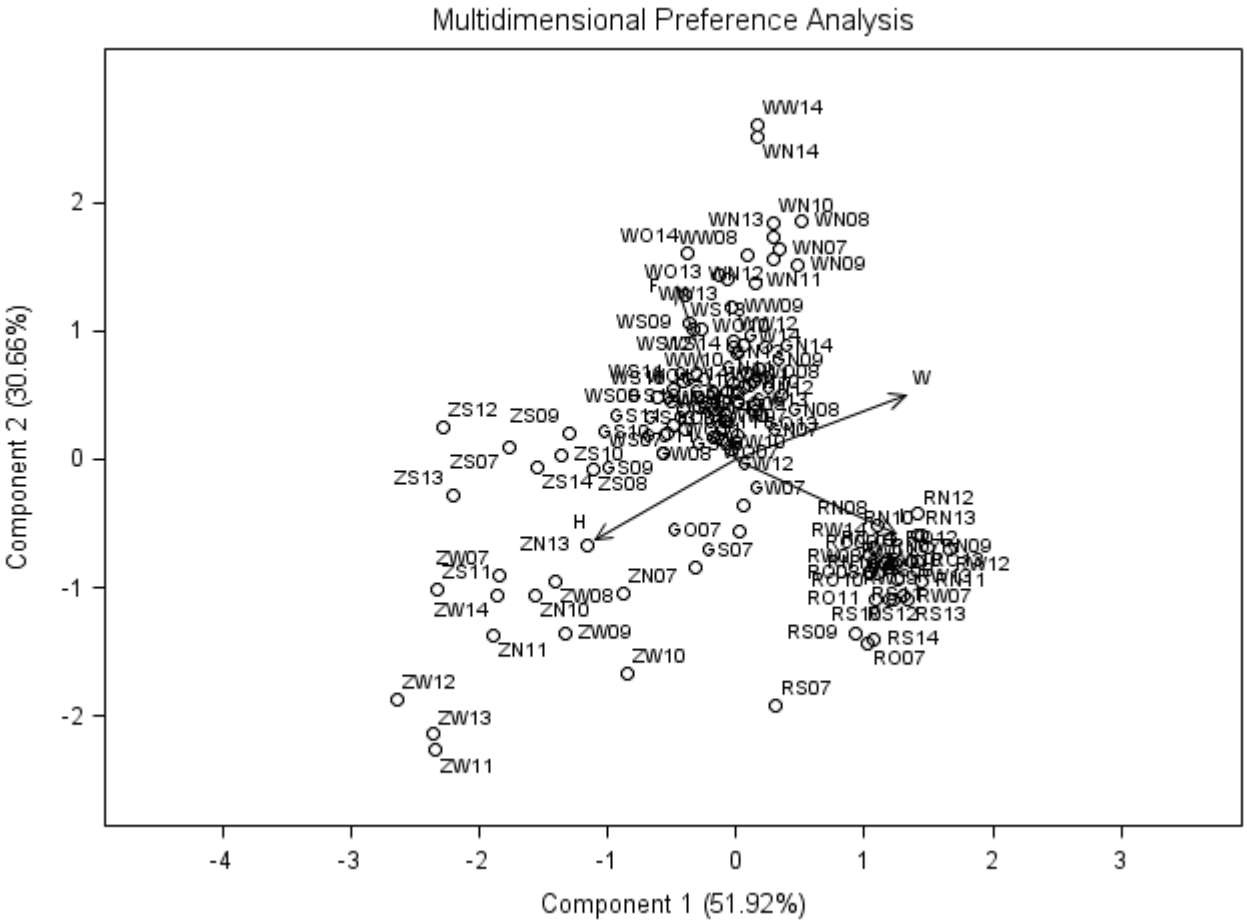

Abbildung 4: BIPLOT auf der Basis von MDP

Mehrere „schöne Dinge“ sind zu sehen. Erstens wieder die saubere Trennung in Zuckerrüben (Z*), Winterraps (R*) und Winterweizen(W*)/Winterroggen (G*). Der zweite Buchstabe kennzeichnet jeweils die Region **N**ord, **S**üd, **O**st bzw. **W**est. Die Buchstaben 3 und 4 stehen dann für das Erntejahr 2007-2014.
Zweitens sehen wir die mittels Vektor gekennzeichneten Variablen I, H, F und W. Diese vier Variablen stellen die Pflanzenschutzkategorien **I**nsektizide, **H**erbizide, **F**ungizide

und **W**achstumsregler dar. Die Daten der Jahre 2007-2013 sind [3] entnommen, die Daten für das Jahr 2014 stammen aus dem sich in Arbeit befindenden nächsten Bericht.
Die Grafik lässt jetzt den Schluss zu, dass die Variabilität der Pflanzenschutzintensität in den Zuckerrüben vorrangig durch die Herbizide, beim Winterraps durch die Insektizide und bei Winterweizen/Winterroggen durch die Fungizide bestimmt wird. Der Landwirt wird an dieser Stelle wohlwollend mit dem Kopf nicken, weil er die bedeutendsten Schaderreger in seinen Kulturen kennt.
Da wir dieses Bild als Nebenprodukt erhalten, sind auch Nachteile vorhanden: wir können kaum Einfluss auf die Grafik nehmen. Vor allem können wir keine Gruppen kennzeichnen. Hierzu gibt es wenigstens zwei Lösungen.

Wir speichern erstens das Ergebnis mittels `ods output MDPrefPlot=BiPlot` und geben dieses mittels `sgplot data=biplot / group=` aus. Wichtig ist hierbei die Anweisung `id ident kultur`, damit wir `idLab1` und `idLab2` zur Verfügung haben.

```
ods graphics on;
proc prinqual data=ackerbau plots=MDP MDPREF=1.5 maxiter=100;
ods output MDPrefPlot=BiPlot;
 transform identity(h i f w);
 id ident kultur;
run;
ods graphics off;

proc sgplot data=BiPlot noautolegend;
scatter x=prin1 y=prin2 / datalabel=idlab1 group=idLab2;
vector x=vec1 y=vec2 / datalabel=label2var;
run;
```

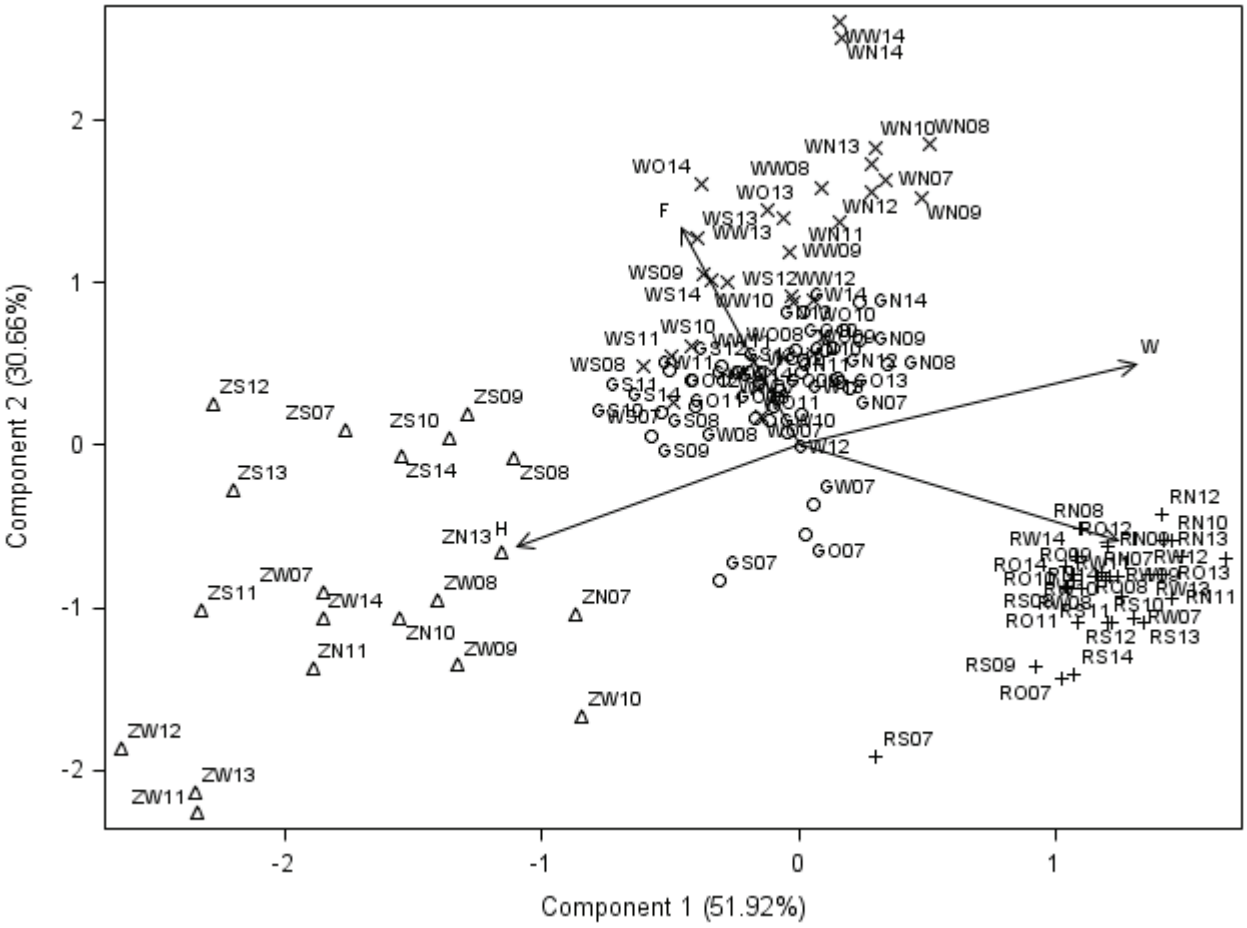

Abbildung 5: BIPLOT auf der Basis von MDP und SGPLOT

Der andere Weg ist die Nutzung des eigenen Makros `biplotmatrix`, das der Autor vor Kenntnis der SAS eigenen Variante bereits fertig gestellt hatte:

```
/* Hauptkomponenten-Analyse */
proc princomp data=ackerbau out=ackerpca outstat=score;
var h i f w;
id ident;
run;

/* Bildung der Matrix mit den Objekt- und Variablen-Koord. */
%biplotmatrix(ackerpca, score, ident);

/* Grafik */
proc sgplot data=PCA_OBJ_VARS noautolegend;
scatter x=prin1 y=prin2 /
  group=kultur datalabel=ident markerattrs=(size=0)
  datalabelattrs=(size=8) datalabelpos=center;
vector x=mprin1 y=mprin2 /
  xorigin=0 yorigin=0 datalabel=_name_
  datalabelattrs=(size=10 weight=bold) lineattrs=(color=black);
xaxis display=none;
yaxis display=none;
run;
```

Das Ergebnis ist mit dem vorhergehenden identisch. Beiden Lösungen zu eigen ist, dass wir die Daten in der Hand haben und mittels `sgplot` die Grafik nach eigenen Wünschen oder den Anforderungen des Verlags selbst gestalten können.

3.4 Non-Linear-Mapping

Das Hauptkomponenten-Display ist ein wichtiges Hilfsmittel bei der Mustererkennung, egal ob 2- oder 3-dimensional.

Dennoch gibt es einen gravierenden Nachteil: Im p-dimensionalen Raum weit entfernt voneinander liegende Objekte können durch die Projektion in die Ebene oder in den Würfel dicht zusammen rücken, vor allem dann, wenn die ersten Hauptkomponenten nur wenig Anteil an der Gesamtvarianz besitzen. Mögliche Fehlinterpretationen sind die Folge. Dieser Mangel lässt sich mittels linearer Transformationen nicht beheben.

Als (eine) Alternative existiert die oben benannte nichtlineare Methode, die die Nachteile mit Methoden der nichtlinearen Optimierung auf der Basis von Gradienten Verfahren iterativ aufheben möchte [4].

Der mathematische Grundgedanke hierbei ist die Minimierung von

$$E = \frac{1}{\sum_{i<j} d_{ij}^*} \sum \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*}$$

d_{ij}^* = p-dimensionale Abstand der Objekte i, j
d_{ij}= 2-dimensionale Abstand der Objekte i, j (in der Grafik-Ebene)

Das Ergebnis ist ein mehr oder weniger starkes Auseinanderschieben der Objekte nach der Hauptkomponentenanalyse (links) in der Ebene (rechts).

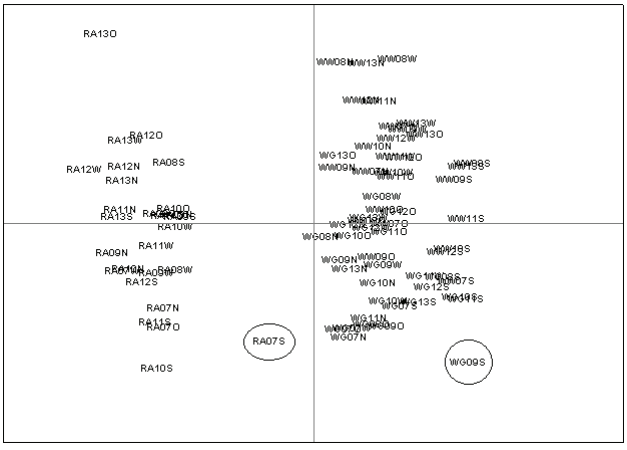

Abbildung 6: Hauptkomponentendisplay

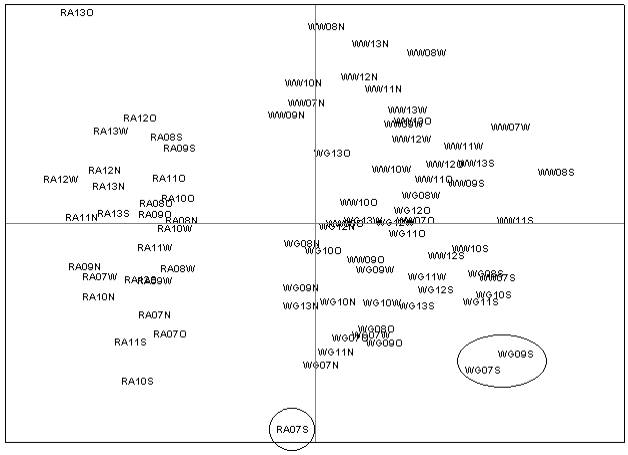

Abbildung 7: Non-Linear-Mapping

Neben dem Auseinanderdriften der Objekte aus Abb. 6 nach Abb. 7 ist auch die Verschiebung einzelner Objekte zu beobachten (siehe Markierungen). Aber auch hier soll das Gesamtbild des Verfahrens mehr interessieren als die Details in den kleinen Grafiken.

Ein Wermutstropfen gibt es allerdings. Die Abb. 7 ist entstanden auf Basis der mittels Non-Linear-Mapping erstellten neuen Koordinaten der Objekte – in der DOS-Box unter Nutzung des alten TURBO-PASCAL Programms. Die Daten wurden per `proc export` im `dlm`-Format exportiert, mit dem alten Programm „non-linear-gemapped“ und dann über `infile` in einem Datastep eingelesen und mittels `merge` in einem zweiten Datastep mit den ursprünglichen Bezeichnungen wieder zusammen geführt. Die vollständige Umsetzung in die SAS Welt ist noch offen. Als Basis können hier die original gedruckten BASIC- oder die vorliegenden PASCAL-Quelltexte dienen.

3.5 Mehr als zwei Y-Achsen

Herausgefordert durch die Frage eines Kollegen, ob denn SAS wie SigmaPlot™ auch drei Y-Achsen könne, dann würde er endgültig zu SAS wechseln, begann die Suche nach einer Lösung. Da die Recherche +SAS +Y3 in Google nichts ergab, wurde das Thema ad acta gelegt. Einige Zeit später stolperte der Autor mehr oder weniger zufällig über die Option `pad` im `sgplot`-Statement und das Thema wurde neu belebt.

Die Daten sind:

Tabelle 1: Beispieldaten für 3+1 Y-Achsen

Schlag	Ackerzahl	Dekade	Flaeche	BI
1	40	30	140	6
2	60	29	140	8
3	45	30	140	7
4	45	28	458	6
5	50	30	458	6
6	40	26	458	6
7	33	30	84	4
8	50	28	1463	4
9	67	30	1463	3
10	44	28	1463	5

Gesucht ist ein Statement der Form

```
proc sgplot … ;
VBAR Schlag / response=(Dekade Flaeche Ackerzahl);
Yaxis … ;   * Dekade ;
Y2axis … ;  * Flaeche ;
Y3axis … ;  * Ackerzahl ;
run;
```

Schon bei der Eingabe des Statements wird `Y3axis` rot eingefärbt, auch die Angabe `response=(Dekade Flaeche Ackerzahl)` ist so nicht möglich.
Im Folgenden wird eine Lösung in drei Schritten angegeben, die sich problemlos auf mehr als drei Y-Achsen erweitern lässt.

Schritt 1 (Datenaufbereitung):
Wir fügen zwei Datenzeilen hinzu, die sinnvolle Min- und Max-Werte für die jeweilige Variable enthalten (Datensatz `temp`). Die Variablen `Schlag` und `BI` sind bewusst leer.

Tabelle 2: zugefügte Dummy-Daten

Schlag	Ackerzahl	Dekade	Flaeche	BI
	0	25	0	
	100	30	1500	

Anschließend speichern wir die Min- und Max-Werte in Makrovariablen:

```
proc sql noprint;
select min(dekade), max(dekade), min(flaeche),
  max(flaeche), min(ackerzahl), max(ackerzahl)
into :y1min,:y1max,:y2min,:y2max,:y3min,:y3max
from temp;
```

Schritt 2 (Mathematische Grundlagen):
Für jede Variable rechnen wir die Schrittweite aus. Zudem wird Y3 anhand der Werte von Y1 transformiert. An dieser Stelle ist anzumerken, dass es unbedingt Y1 sein muss.

$$Y_{3i}^{*} = Y_{1min} + (Y_{3i} - Y_{3min}) * \frac{Y_{1max} - Y_{1min}}{Y_{3max} - Y_{3min}}$$

In der SAS Sprache sieht es dann wie folgt aus:

```
/* Schrittweiten und Faktor f definieren */
%LET step1=%sysevalf((&y1max.-&y1min)/5); /* Y1-Achse */
%LET step2=%sysevalf((&y2max.-&y2min)/5); /* Y2-Achse */
%LET step3=%sysevalf((&y3max.-&y3min)/5); /* Y3-Achse */
%LET f=%sysevalf((&y1max.-&y1min.)/(&y3max.-&y3min.));

/* Y3 entsprechend Y1 skalieren */
proc sql;
create table temp2 as
select
  schlag, klasse, dekade, flaeche,
  &y1min. + (ackerzahl-&y3min.) * &f. as ACKERZAHL
from temp;
```

Schritt 3 (Achsenbeschriftung):
Nachdem die Daten jetzt passen, muss noch die Beschriftung für die Y3-Achse erstellt werden. Das geschieht als `annotate`-Datensatz:

```
/* Jetzt den Annotate-Datensatz bilden */
data anno;
length y1space $ 11 textcolor $ 5;
retain
  function 'text'
  x1 95     x1space 'graphpercent'
  y1space 'datavalue'
  textcolor 'red' textsize 9;

/* Bauen der Y3-Achse */
do i = 0 to 5;
y1 = &y1min. + i*&step1.;
label = PUT (SYMGET('y3min') + i*SYMGET('step3'), best.);
output;
end;

/* Überschriften für 2. und 3. Achse */
  y1space = 'wallpercent'; width = 20;
  anchor = 'top'; textweight='normal';

  y1 = 108; /* Position oberhalb */
  x1=90; label="Kulturfläche"; textcolor="blue";
  output;

  y1 = 105; /* Position oberhalb, aber etwas tiefer */
  x1=95; label="Ackerzahl"; textcolor="red";
  output;
  drop i;
run;
```

In der `%do`-Schleife wird die Skalierung erstellt. `Y1` enthält die Position, `Label` den Wert. Anschließend werden noch die Beschriftungen oberhalb von `Y2` und `Y3` erstellt.

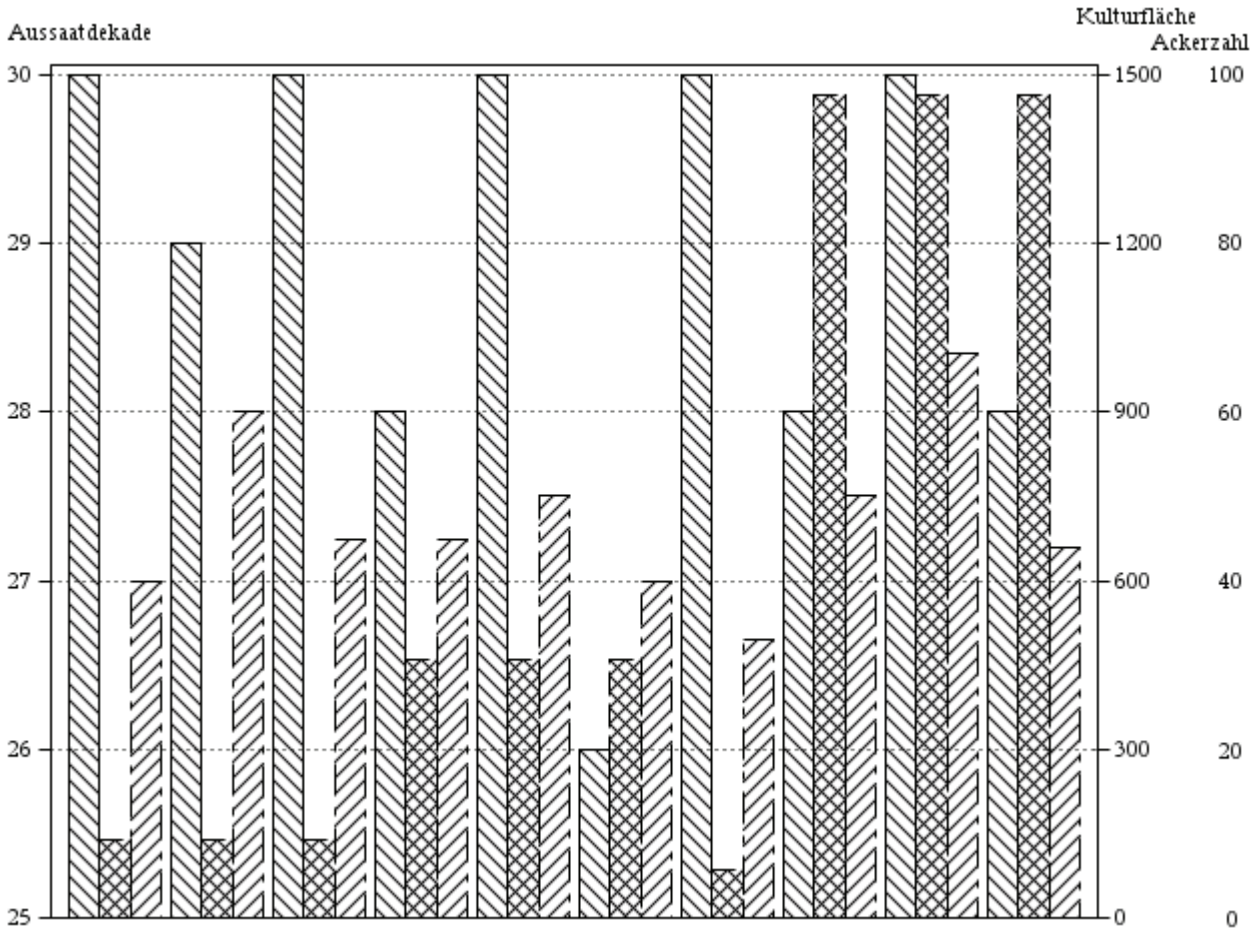

Abbildung 8: Beispiel für drei Y-Achsen, die Säulen entsprechen den Achsen von links nach rechts

Das Statement für Abbildung 8 lautet:

```
proc sgplot data=temp2 sganno=anno pad=(right=10%) noautolegend;
styleattrs datacontrastcolors=(black);
vbar schlag / response=dekade fillattrs=(color=gray)
     discreteoffset=-0.3 barwidth=0.3 nostatlabel;
vbar schlag / response=flaeche fillattrs=(color=blue) Y2axis
     barwidth=0.3 nostatlabel;
vbar schlag / response=ackerzahl fillattrs=(color=red)
     discreteoffset=0.3 barwidth=0.3 nostatlabel;
xaxis display=(novalues nolabel noticks);
yaxis labelpos=top label="Aussaatdekade" labelattrs=(size=9)
     valueattrs=(color=black)
     values=(&y1min. to &y1max. by &step1.);
y2axis display=(nolabel) valueattrs=(color=blue)
     values=(&y2min. to &y2max. by &step2.) grid;
run;
```

In der ersten Zeile weisen wir mit `sganno=anno` den annotate-Datensatz zu und mit `pad=(right=10%)` schaffen wir rechts Platz für die 3. Achse. Die erste bzw. dritte `vbar` verschieben wir mit `discreteoffset=-0.3/0.3` nach links bzw. nach rechts.

Stellen wir zusätzlich hinter jede vbar noch die folgende `groupdef`-Definition:

```
%LET groupdef = group=bi attrid=bi grouporder=ascending;
```

und verwenden eine aufsteigende Farbsättigung, so kann man u. U. erkennen, welche Kombination von Fläche, Ackerzahl und Aussaatdekade zu welcher BI-Klasse führt:

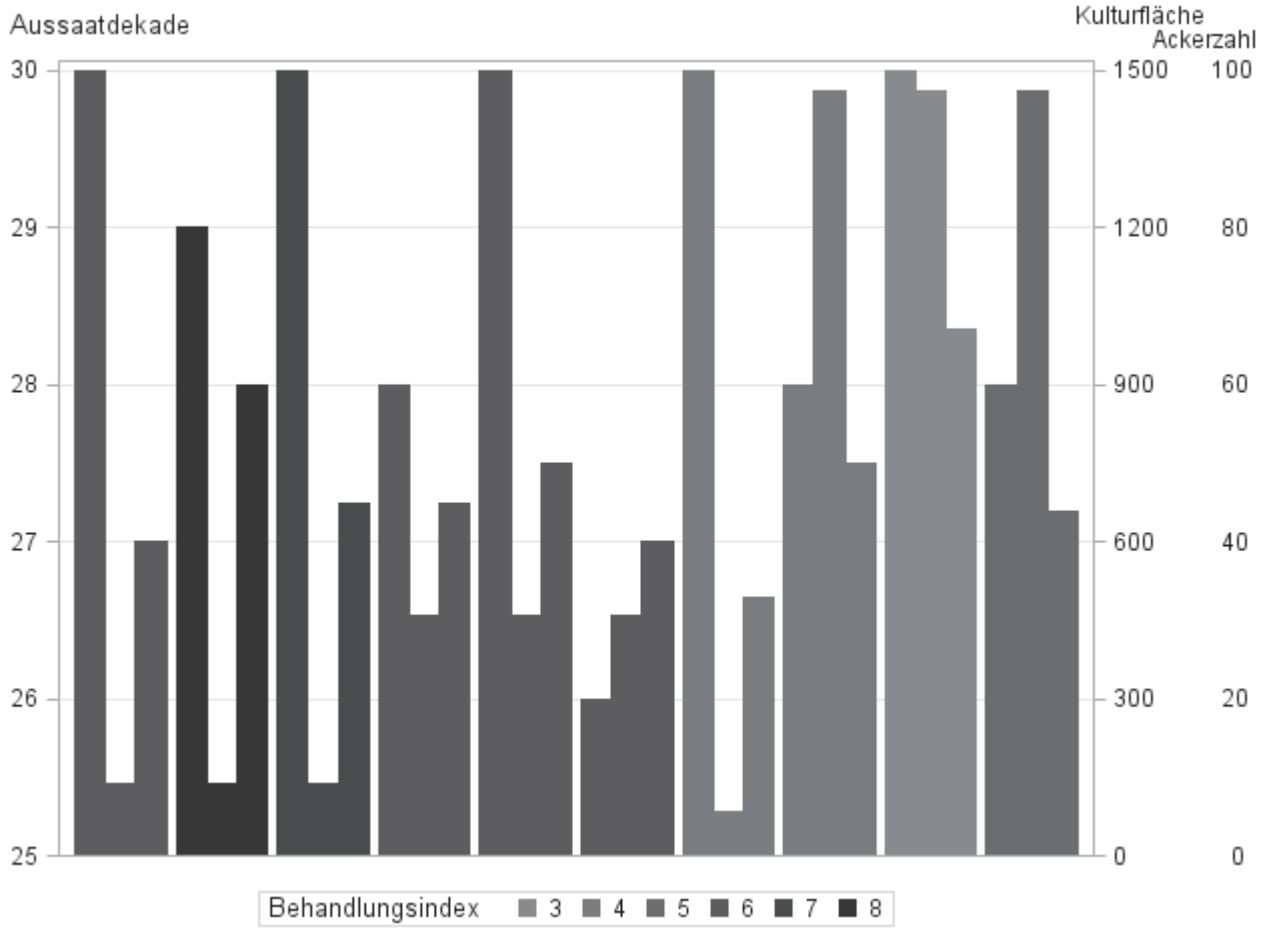

Abbildung 9: Zusammenhang zwischen BI -Klasse und Aussaatdekade / Kulturfläche / Ackerzahl

4 Ausblick

Wie in 3.4 angesprochen, ist die Umsetzung des Non-Linear-Mapping noch offen. Unter Nutzung der freien Programmiermöglichkeit mittels `proc fcmp` sollte diese Umsetzung gelingen [5].

Die Würfel-Drehung setzt die `proc g3d` voraus, die aber in der SAS University Edition nicht unterstützt wird. An anderer Stelle des bereits oben angesprochenen Blogs (http://blogs.sas.com/content/graphicallyspeaking/2015/03/10/a-3d-scatter-plot-macro/) wird jedoch eine 3D-Würfel-Darstellung auf reiner `sgplot`-Basis demonstriert:

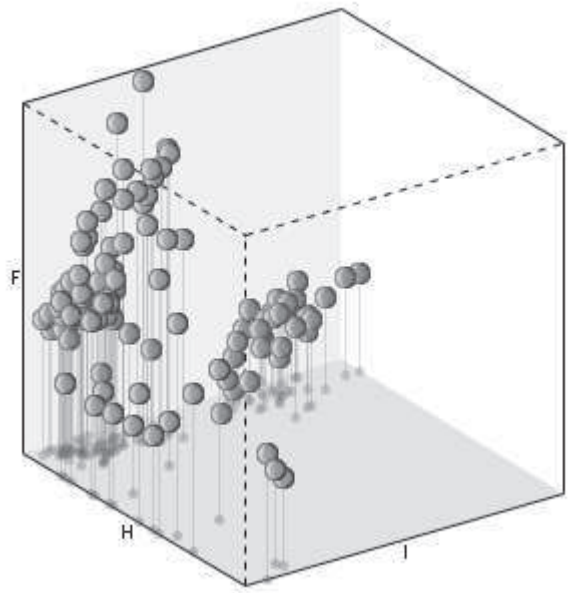

Abbildung 10: beispielhafte 3D-Darstellung mittels Ortho3D_Macro

Das hierbei verwendete `Ortho3D_Macro` soll in modifizierter Form die traditionelle `proc g3d` ablösen, um so die Möglichkeit sich drehender Würfel für alle SAS Plattformen zu ermöglichen.

Ob sich der Aufwand lohnt, für eine höhere Anzahl von Y-Achsen ein universelles Makro zu schreiben, muss die Zukunft zeigen. Solange es bei wenigen Anforderungen bleibt, ist die Übernahme aus der Quelle bzw. hier aus dem Manuskript sicherlich die einfachste Variante. Die hier vorgestellten drei Achsen und die Farbwahl als vierte Dimension bieten schon viele Möglichkeiten für die Zusammenhangsanalyse.

Die im Beitrag verwendeten Makros, Programme und IRIS-Beispieldaten sind auf http://sf.jki.bund.de/sas/ abgelegt. Als SAS-Version wurde die Version 9.4 M2 unter Windows 10 verwendet. Bei der Nutzung unter Linux müssen die Pfadtrenner angepasst werden.

Literatur

[1] G. Henrion, A. Henrion, R. Henrion: Beispiele zur Datenanalyse mit BASIC-Programmen, Deutscher Verlag der Wissenschaften, Berlin 1988.

[2] L. Lebart, A. Morineau, J.-P. Fénelon: Statistische Datenanalyse. Methoden und Programme. Akademie-Verlag, Berlin 1984.

[3] B. Freier, J. Sellmann, J. Strassemeyer, J. Schwarz, B. Klocke, H. Kehlenbeck, W. Zornbach: Netz Vergleichsbetriebe Pflanzenschutz. Jahresbericht 2013. Berichte aus dem Julius Kühn-Institut 178. Saphir-Verlag, Ribbesbüttel 2015.

[4] J. W. Sammon, Jr, A nonlinear mapping for data structure analysis, IEEE Transactions on Computers, vol. C-18, no. 5, pp. 401–409, 1969.

[5] A. Menrath: Schöne neue Welt - So können Sie fehlende SAS-Funktionalitäten mit PROC FCMP nachrüsten. Proceedings der 17. KSFE Ulm, S. 301-310. Shaker-Verlag, Aachen 2013.