Zentrale Layout-Steuerung von ODS Grafiken mit ODS Styles

Johannes Lang
HMS Analytical Software
Rohrbacher Str.26
69115 Heidelberg
johannes.lang@analytical-software.de

Zusammenfassung

In diesem Beitrag wird gezeigt, wie die Vorteile von ODS Styles zur zentralen Layout-Verwaltung mit ODS Graphics kombiniert werden können, um hartcodierte Layout-Definitionen im Grafikprogramm zu vermeiden. Hierfür werden so genannte Attribute Maps genutzt, um eine Entkopplungsschicht zwischen ODS Style und Grafikprozedur zu erstellen. Dabei werden in einer (temporären) Steuertabelle jeder Variablenausprägung passende ODS Style-Elemente zugewiesen, und die Tabelle anschließend beim Aufruf der Grafikprozedur referenziert. Zentrale Farbwerte und Linienarten werden in permanenten Steuertabellen gepflegt, aus denen dynamisch ODS Styles erzeugt werden können. Generische Grafikmakros verbergen diese Komplexität für den Anwender, so dass diese Vorteile ohne detaillierte ODS Graphics Kenntnisse genutzt werden können.

Schlüsselwörter: ODS Styles, ODS Graphics, Attribute Maps

1 Einleitung: Warum ODS Graphics?

Viele SAS-Anwender haben in den letzten Jahren ihre Grafikprogramme modernisiert, indem sie von SAS/GRAPH auf den Nachfolger ODS Graphics umgestellt haben. Die Basis von ODS Graphics sind so genannte Templates, in denen Struktur und Layout einer Grafik definiert werden. Diese werden in der so genannten Graph Template Language (GTL) erstellt, die jedoch nur für spezielle Anforderungen manuell erstellt werden muss. Für viele Standard-Fälle genügt die Kenntnis der Grafikprozeduren wie z.B. SGPLOT, welche die GTL-Erzeugung im Hintergrund übernehmen.

ODS Styles sind schon lange für die Grafikerzeugung nutzbar, um mehrere Layout-Einstellungen zusammenhängend definieren zu können (z.B. Linienfarben und -breiten, sowie Schriftarten und -größen). SAS liefert darüber hinaus viele vorgefertigte Styles mit, die direkt genutzt oder auch als Vorlage für eigene Styles verwendet werden können.

In diesem Beitrag soll anhand ausgewählter Beispiele gezeigt werden, wie die Vorteile von ODS Styles und ODS Graphics kombiniert werden können, um Layoutdefinition und Grafikerzeugung sauber zu trennen. Das Schlüsselelement hierbei sind so genannte Attribute Maps, welche in ODS Graphics enthalten sind und als Entkopplungsschicht zwischen ODS Styles und der jeweiligen Grafikprozedur genutzt werden können.

2 Grundlagen zu ODS Styles

Ein ODS Style ist ein Template zur Steuerung von festgelegten Layout-Eigenschaften, welche entweder komplett definiert oder nur in Teilen überschrieben werden können. Die Layout-Eigenschaften sind in Form von benannten Style-Elementen definiert, welche festgelegte Attribute in Form von Name-Wert-Paaren enthalten (z.B. kann im Style-Element GraphDataDefault das Attribut color auf einen Farbwert gesetzt werden). Manche Layout-Elemente beeinflussen alle Grafiken (z.B. GraphFonts), manche nur bestimmte Grafiktypen (z.B. GraphBox für Boxplots). Eine Übersicht findet sich z.B. in [1], S.607ff.
ODS Styles werden standardmäßig in so genannten Template Stores gespeichert, die wie Standard-Datasets über einen Libref angesprochen werden. Über globale Optionen (ODS PATH) wird eine Suchreihenfolge festgelegt, um Template Stores zur Laufzeit für Lese- bzw. Schreiboperationen zu finden.
Die Anwendung der definierten Style-Elemente erfolgt standardmäßig implizit, d.h. ohne explizite Referenzierung im Grafikprogramm (vgl. **Listing 1**).

```
ODS HTML Style=analysis;
PROC SGPLOT data=sashelp.cars;
   vbar make;
RUN;
```

Listing 1: Beispiel für implizite Nutzung eines ODS Style ohne Referenz in der Grafikprozedur

Es ist aber auch möglich, explizit auf Style-Elemente zu verweisen, um alle zugehörigen Attribute für eine bestimmte Datenreihe bei der Grafikerzeugung anzuwenden. Dies wird im folgenden Abschnitt näher erläutert.

3 Referenzierung von Style-Elementen bei der Nutzung von PROC SGPLOT

3.1 Angabe von Style-Elementen beim Plot-Statement

Bei einem Aufruf von SGPLOT können mehrere Plot-Statements angegeben werden, um z.B. mehrere Datenreihen in einer Grafik darzustellen (zur Kombinierbarkeit unterschiedlicher Plots vgl. [2]). Für jedes Statement können Layout-Attribute definiert werden, wobei hier nicht nur explizite Angaben möglich sind (siehe **Listing 2**), sondern auch Verweise auf ODS Style-Elemente (siehe **Listing 3**).

```
proc sgplot data=sashelp.class;
   density height /
      lineattrs=(color=red)
   ;
run;
```

Listing 2: Beispiel für Plot-Statement mit hartcodierter Layout-Angabe

```
proc sgplot data=sashelp.class;
   density height /
      lineattrs=GraphData1
   ;
run;
```

Listing 3: Beispiel für Plot-Statement mit Angabe von ODS Style-Element

Der Vorteil der zweiten Variante besteht darin, dass im Grafikprogramm keine hartcodierten Layout-Details mehr stehen, die schwer zu pflegen sind. Nachteilig ist jedoch, dass diese Layout-Zuweisung in jedem Plot-Statement erforderlich ist, und keine komplett vom Grafikprogramm getrennte Layout-Verwaltung ermöglicht.

3.2 Angabe von Style-Elementen in einer Attribute Map

3.2.1 Grundlagen zu Attribute Maps

Um Layout-Einstellungen komplett getrennt vom Grafikprogramm verwalten zu können, gibt es seit SAS 9.3 das Konzept der Attribute Maps. Eine Attribute Map ist eine Sammlung von festgelegten Layout-Merkmalen, die normalerweise für unterschiedliche Ausprägungen einer Gruppierungsvariablen festgelegt werden können. Damit kann z.B. das ungewollte Verschieben von Farbzuordnungen bei fehlenden Ausprägungen in den Daten verhindert werden, da die Zuordnung dann nicht mehr zur Laufzeit erfolgt, sondern vorher festgelegt wird.

Seit SAS 9.4M1 können Attribute Maps sowohl mit Grafikprozeduren wie SGPLOT, als auch in GTL-Templates genutzt werden. Es gibt Attribute Maps für diskrete Werte (type = discrete) oder für Wertebereiche (type = range). Hier wird nur die Variante für diskrete Werte vorgestellt.

Ein Attribute Map Dataset kann mehrere Attribute Maps enthalten, die jeweils durch einen Schlüsselwert in der festgelegten Variablen ID unterschieden werden müssen (vgl. **Tabelle 2**). In einer weiteren festgelegten Variablen VALUE werden die Variablenausprägungen aufgelistet, für welche die jeweiligen Layout-Einstellungen gelten sollen. Je nachdem welche Einstellungen definiert werden, müssen diese in unterschiedlichen, festgelegten Variablen erfolgen.

3.2.2 Nutzung einer diskreten Attribute Map für gruppierte Grafiken

Wird zum Beispiel eine Auswertung nach Ländern grafisch dargestellt, so ist die Länderkennung typischerweise in einer Gruppierungsvariablen hinterlegt (vgl.). Um nun jedem Land ein bestimmtes Layout zuzuweisen, müssen alle möglichen Länderkennungen in einer Attribute Map hinterlegt werden (vgl. **Tabelle 2**).

Tabelle 1: Beispiel für Eingabedaten mit Gruppierungsvariablen (input)

Obs	X	Country	Var1
1	Jul08	AU	0,85
2	Jul08	CA	0,77
3	Jul08	DE	0,99
4	Jul08	IL	0,56
5	Aug08	CA	0,33
…	…	…	…

Tabelle 2: Beispiel für diskrete AttributeMap, welche auf ODS Style-Elemente verweist (dattrmap)

Obs	Id	Value	LineStyleElement	FillStyleElement
1	myid	AU	GraphData1	GraphData1
2	myid	CA	GraphData2	GraphData2
3	myid	DE	GraphData3	GraphData3
4	myid	IL	GraphData4	GraphData4

```
proc sgplot data=input dattrmap=dattrmap;
   xaxis
     […]
   ;
   yaxis
     […]
   ;
   series
      x = x
      y = var1 /
         group = country
         attrid = myid
   ;
run;
```

Listing 4: PROC SGPLOT Aufruf für Liniengrafik mit gruppierten Eingabedaten und Attribute Map

Für jede Ausprägung der Gruppierungsvariablen wird eine ODS Style Klasse referenziert (GraphData<n>), welche die entsprechenden Layout-Einstellungen enthält. Dabei empfiehlt es sich, nach Möglichkeit dieselbe Style-Klasse für alle Layout-Merkmale einer Ausprägung zu nutzen, um ein abgestimmtes Layout gewährleisten zu können.

3.2.3 Nutzung einer diskreten Attribute Map für ungruppierte Grafiken

Wenn eine Attribute Map für ungruppierte Daten genutzt werden soll, so ist ein Kunstgriff nötig: Die Daten müssen transponiert werden, so dass eine künstliche Gruppierungsvariable entsteht, welche die Namen der darzustellenden Variablen enthält (vgl. **Tabelle 3**,

Tabelle 4, sowie **Listing 6**). Damit können dann Layout-Einstellungen für ganze Datenreihen definiert werden, nicht nur für einzelne Ausprägungen (siehe **Tabelle 6**).

Tabelle 3: Beispiel für Eingabedaten ohne Gruppierungsvariablen (input)

Obs	X	Var1	Var2	Var3	Var4
1	Jul08	1,16	1,13	0,32	2,33
2	Aug08	1,14	1,18	0,31	2,38
…	…	…	…	…	…

```
proc transpose data=input out=input_transposed prefix=col;
   by x;
   var var1-var4;
run;

data input_transposed2 (drop = Col1);
   set input_transposed (keep = x _NAME_ Col1);
   label _NAME_ = '';
   attrib
      var1  length = 8
      var2  length = 8
      var2  length = 8
      var4  length = 8
   ;
   select (_NAME_);
      when ('VAR1') do;
         var1 = Col1;
      end;
      when ('VAR2') do;
         var2 = Col1;
      end;
      when ('VAR3') do;
         var3 = Col1;
      end;
      when ('VAR4') do;
         var4 = Col1;
      end;
      otherwise;
   end;
run;
```

Listing 5: Transponierung der Eingabedaten als Vorbereitung für PROC SGPLOT mit Attribute Map

Tabelle 4: Beispiel für transponierte Eingabedaten mit künstlicher Gruppierungsvariablen (input_transposed2)

Obs	X	_NAME_	Var1	Var2	Var3	Var4
1	Jul08	VAR1	1,16	.	.	.
2	Jul08	VAR2	.	1,13	.	.
3	Jul08	VAR3	.	.	0,32	.
4	Jul08	VAR4	.	.	.	2,33
5	Aug08	VAR1	1,14	.	.	.
6	Aug08	VAR2	.	1,18	.	.
7	Aug08	VAR3	.	.	0,31	.
8	Aug08	VAR4	.	.	.	2,38
…	…	…	…	…	…	…

Tabelle 5: Beispiel für diskrete Attribute Map mit Variablennamen statt -ausprägungen (dattrmap)

Obs	Id	Value	LineStyleElement	FillStyleElement
1	Linien	Var1	GraphData1	GraphData1
2	Linien	Var2	GraphData2	GraphData2
3	Linien	Var3	GraphData3	GraphData3
4	Linien	Var4	GraphData4	GraphData4

```
proc sgplot data=input_transposed2 dattrmap=dattrmap;
   xaxis
     […]
   ;
   yaxis
     […]
   ;
   series
      x = x
      y = var1 /
         group = _name_
         attrid = linien
   ;
   […]
   series
      x = x
      y = var4 /
         group = _name_
         attrid = linien
   ;
run;
```

Listing 6: PROC SGPLOT Aufruf für Liniengrafik mit transponierten Eingabedaten und Attribute Map

4 Bereitstellung für den Anwender

Damit die Vorteile der zentralen Layout-Verwaltung mit ODS Styles und die Nutzung von Attribute Maps als Entkopplungsschicht zu den Grafikprogrammen für alle SAS Grafikprogrammierer einfach nutzbar sind, empfiehlt es sich, ein paar zusätzliche Strukturen einzuführen. Diese sollen im Folgenden kurz vorgestellt werden.

4.1 Bereitstellen mehrerer ODS Styles für unterschiedliche Anforderungen

Für Liniengrafiken werden teilweise andere Werte benötigt als für Balkengrafiken, beispielsweise um die Linienbreiten zu steuern. Daher muss für Balkengrafiken unter Umständen ein anderer ODS Style verwendet werden. Dabei kann der Vererbungsmechanismus von ODS Styles genutzt werden, um doppelten Code zu vermeiden: Wenn z.B. für Balkengrafiken nur eine Style-Klasse modifiziert werden muss, dann muss auch nur diese im neuen Style aufgeführt werden.

4.2 Einführung von Grafiktypen

Bei Standard-Grafiken, die z.B. für turnusmäßige Berichte erzeugt werden, empfiehlt sich die Einführung einer künstlichen Typkennung (z.B. SERIESPLOT, BARCHART oder BOXPLOT), anhand derer spezielle Grundeinstellungen automatisch gesetzt werden. So kann z.B. der passende ODS Style aktiviert und eine passende Größeneinstellung voreingestellt werden.

4.3 Bereitstellen von Steuertabellen und Einlesemakros

Für die Definition von Linienarten und Farbwerten können jeweils Steuertabellen bereitgestellt werden (siehe **Tabelle 6** und **Tabelle 7**). Im Beispiel werden zehn Farben definiert, die jeweils mit durchgezogener Linie (LinePattern = solid) oder mit gepunkteter Linie (LinePattern = dot) genutzt werden können. Für jede dieser 20 Kombinationen wird ein eindeutiger Layout-Schlüssel vergeben.

Layout-Änderungen bzw. -Erweiterungen sind nun einfach möglich, indem entsprechende ODS Styles dynamisch daraus erzeugt werden. Hierfür können z.B. mit Hilfe von SAS Makroprogrammierung pro Tabellenzeile entsprechende GraphData<n> Style-Elemente erzeugt werden. Die Farbdefinitionen können entsprechend nummerierten Style-Attributen (gcdata<n>) zugewiesen und in den GraphData<n> Style-Elementen referenziert werden (vgl. **Listing 8**).

Pro ODS Style-Erzeugung sollte ein eigenes Makro (z.B. %createCustomStyle_<n>) erstellt werden, sowie ein allgemeines Hüllen-Makro (z.B. %createAllCustomStyles), welches alle benutzerdefinierten Styles erzeugt und permanent ablegt. Dieses Makro muss dann nur bei Änderungen an den Styles bzw. den Steuertabellen aufgerufen werden.

Es empfiehlt sich, Beispielgrafiken mit den unterstützten Linienarten und Farbwerten zentral abzulegen, sodass sie als Referenz genutzt werden können.

Tabelle 6: Beispiel für Steuertabelle mit Linienarten

SortIndex	LineColorIndex	LinePattern
1	1	Solid
2	2	Solid
3	3	Solid
…	…	…
11	1	Dot
12	2	Dot
13	3	Dot
…	…	…

Tabelle 7: Beispiel für Steuertabelle mit Farbdefinitionen

SortIndex	Label	Value
1	CUSTOM BLAU	Cx5A9BBE
2	CUSTOM GRÜN	Cx506E5F
3	CUSTOM GRAU	CxBEB9B4
…	…	…

```
proc template;
   define style MyStyles.Customer_01;
      parent = Styles.Listing;

      class GraphColors /
         'CUSTOM BLAU' = Cx5A9BBE
         'gcdata1' = GraphColors('CUSTOM BLAU')
         'CUSTOM GRÜN' = Cx506E5F
         'gcdata2' = GraphColors('CUSTOM GRÜN')
         'CUSTOM GRAU' = CxBEB9B4
         'gcdata3' = GraphColors('CUSTOM GRAU')
         […]
       ;

      […]

      class GraphData1 /
         linestyle = 1
         contrastcolor = GraphColors('gcdata1')
         color = GraphColors('gcdata1')
       ;
      class GraphData2 /
         linestyle = 1
         contrastcolor = GraphColors('gcdata2')
         color = GraphColors('gcdata2')
       ;
```

```
      class GraphData3 /
         linestyle = 1
         contrastcolor = GraphColors('gcdata3')
         color = GraphColors('gcdata3')
       ;
      […]
      class GraphData11 /
         linestyle = 2
         contrastcolor = GraphColors('gcdata1')
         color = GraphColors('gcdata1')
       ;
      class GraphData12 /
         linestyle = 2
         contrastcolor = GraphColors('gcdata2')
         color = GraphColors('gcdata2')
       ;
      class GraphData13 /
         linestyle = 2
         contrastcolor = GraphColors('gcdata3')
         color = GraphColors('gcdata3')
       ;
       […]
   end;
run;
```

Listing 7: Beispiel für resultierenden ODS Style mit Farbdefinitionen aus Steuertabelle (Auszug)

4.4 Bereitstellen von Grafikmakros für bekannte Grafiktypen

Falls oft dieselben oder strukturell ähnliche Grafiken erzeugt werden, ist es sinnvoll hierfür generische Grafikmakros anzubieten, welche die Details der Style-Element-Referenzierung kapseln. Beim Aufruf können dann die darzustellenden Variablen und die entsprechenden Linienarten als Makroparameter-Listen übergeben werden (siehe **Listing 8**). Die Details der Attribute Map Erzeugung werden im Makro gekapselt und müssen dem Aufrufer nicht bekannt sein. Dadurch werden SAS Programmierer in die Lage versetzt, die Vorteile von ODS Styles und ODS Graphics zu nutzen, ohne in diesen Technologien Experten sein zu müssen[1].

```
%genericLinePlot (p_sTitle = Trefferquote von Algorithmen
  ,p_sInDSRef           = work.data
  ,p_sInDSXVar          = category
  ,p_sInDSYVars         = var1 var2 var3 var4
  ,p_sInDSYVarPatterns  = 1 6 11 11
  ,p_sXAxisLabel        = Kategorie
  ,p_sYAxisLabel        = Trefferquote je nach Algorithmus (%)
  ,p_nYAxisFrom         = 0
  ,p_nYAxisTo           = 1000
```

[1] Zur Automatisierung der Attribute Map Erzeugung siehe auch [5].

```
  ,p_nYAxisBy           = 100
  ,p_sXAxisType         = DISCRETE
  ,p_sXAxisValueList    = 'Kat1' 'Kat2' 'Kat3' 'Kat4'
  ,p_sOutPlotDir        = C:/temp
  ,p_sOutPlotFileName   = lineplot
  ,p_sOutPlotFiletype   = png
  ,p_sVarReturnCode     = g_bRC_genericLinePlot
 );
```

Listing 8: Aufruf-Beispiel für ein generisches Liniengrafikmakro (Variablen und Linienarten als Parameter)

5 Fazit

Wie in diesem Beitrag gezeigt wurde, kann mit ODS Styles und ODS Graphics eine saubere Trennung von Layoutdefinition und Grafikerstellung erreicht werden. Der Schlüssel hierfür sind so genannte Attribute Maps, in denen ODS Style-Elemente referenziert werden können. Dadurch müssen keine hartcodierten Layout-Informationen mehr im Grafikprogramm stehen, sondern es wird nur ein Verweis auf ein Attribute Map Dataset eingefügt. Dies wurde am Beispiel von PROC SGPLOT gezeigt.

Eine zentrale Definition von Farbwerten und Linienarten kann über Steuertabellen realisiert werden, aus denen dynamisch ODS Styles bzw. SAS Formate erzeugt werden. Für wiederkehrende Grafiktypen sollten generische Makros bereitgestellt werden, denen beim Aufruf die darzustellenden Variablen mit ihrem gewünschten Layout-Schlüssel übergeben werden können. Dadurch können die Vorteile von ODS Styles und ODS Graphics genutzt werden, ohne die Details der Implementierung kennen zu müssen.

Literatur

[1] SAS Institute Inc. (2016): SAS 9.4 Graph Template Language: User's Guide, Fifth Edition. Cary, NC.

[2] SAS Institute Inc. (2016): SAS 9.4 ODS Graphics: Procedures Guide, Sixth Edition. Cary, NC.

[3] SAS Institute Inc. (2016): SAS 9.4 Output Delivery System: Procedures Guide, Third Edition. Cary, NC.

[4] SAS Institute Inc. (2016): SAS 9.4 Output Delivery System: User's Guide, Fifth Edition. Cary, NC.

[5] R. Wicklin (2017): Automate the creation of a discrete attribute map, https://blogs.sas.com/content/iml/2017/01/30/auto-discrete-attr-map.html [04.03.2019]