Wetterkarten mit SAS

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

Zusammenfassung

Seit SAS 9.4 Release M5 gibt es die Prozedur sgmap. Mit dieser können geographische Karten analog zu gmap erstellt werden. An 2 Beispielen werden Einblicke in die Funktionsweise des sgmap-Statements choromap vorgestellt.
Im ersten Fall wird mittels eines ESRI-Shapefiles auf der Basis der bestehenden Boden-Klima-Räume und ausgewählter Klimadaten eine Gruppierung von Deutschland in neue klima- und bodenähnliche Regionen vorgenommen. Hier ist der wesentliche Punkt die Verwendung der Bezeichnungen in der Karte.
Im zweiten, wichtigeren Fall wird anhand der vom Deutschen Wetterdienst bereitgestellten Vorhersagewerte eine deutschland-weite Temperaturkarte erstellt. Im Vergleich zu sgplot/polygon sollen Möglichkeiten und Grenzen von sgmap aufgezeigt werden.

Schlüsselwörter: sgmap, shapefile, choromap, template, sgplot, polygon

1 Einleitung

Im Institut für Strategien und Folgenabschätzung des Julius Kühn-Instituts (https://www.julius-kuehn.de/sf) entwickeln und erproben wir Konzepte und Handlungsoptionen für die integrierte Pflanzenproduktion und für den Ökolandbau. Zu den Schwerpunkten gehören hierbei räumliche Analysen und Modellierung unter der Nutzung von Geoinformationssystemen. Ein weiterer Schwerpunkt ist die Erforschung der Klimawandel-Auswirkungen auf die Landwirtschaft, aber auch die Erhebung von Daten zu Bodenzustand und Fruchtfolgesystemen oder zur Artenvielfalt, welches in die Erstellung von Karten mündet (http://geoportal.julius-kuehn.de/index.htm). Dabei erhalten wir täglich Daten vom Deutschen Wetterdienst, die zur Modellierung vergangener Ereignisse, aber auch zur Vorhersage künftiger Phänomene genutzt werden.

In diesem Beitrag geht es aber nicht um die inhaltliche Bewertung, sondern um die neuen Möglichkeiten, die SAS 9.4 ab dem Release M5 mittels der Prozedur `sgmap` bietet, Karten zu erstellen, einzufärben, zu beschriften usw. Als Karten-Basis dienen in allen Fällen ESRI-Shapfiles, die auf der Basis des Datenbestandes des JKI erstellt worden sind. Das verwendete Koordinaten-System ist DHDN / 3-degree Gauss-Kruger zone 3 (EPSG:31467).

1.1 Bezeichnungen in der Karte

Im Jahr 2007 stellte das Julius Kühn-Institut (JKI, damals noch Biologische Bundesanstalt) die „Definition von Boden-Klima-Räumen für die Bundesrepublik Deutschland“ vor [1]. Dabei wurden 50 Regionen (BKR) gebildet, die z. T. aus mehreren, nicht zusammenhängenden Teilflächen bestehen. Aktuelle Anforderungen führten zu der Notwendigkeit, diese 50 BKR auf 5-10 zu reduzieren.
Für jede BKR stehen langjährige Klimadaten und aktuelle Bodendaten zur Verfügung, so dass mittels Clusteranalyse neue Regionen gebildet werden konnten. In der Ergebniskarte sollten die neuen Regionen auf der Basis der BKR inkl. deren Nummer dargestellt werden. Die Herausforderung ist, wie oben erwähnt, dass einige BKR aus mehreren Teilen bestehen. So kann kein Standarddatensatz zur Beschriftung verwendet werden, der pro BKR nur eine Beschriftung enthält.

Zur Erstellung der einzelnen Karten mittels `sgmap` werden in diesem Abschnitt die folgenden vier Datensätze genutzt:

- bkr2016: Geometrien der BKR, die aus den ESRI-Dateien (.shp, .dbf, .shx) eingelesen wurden
- cluster: die neuen Regionsnr. aus der hierarchischen Clusteranalyse
- bkr2016mid: berechnete „Schwerpunkte“ der BKR
- bkr2016own: die Bezeichnungen, die aus einer manuell erstellten Excel-Datei stammen, in der für jede BKR ein oder mehrere Koordinaten angegeben sind.

Die Geometrien werden wie folgt importiert, die Dateien `bkr_2016.dbf` und `bkr_2016.dbf` müssen im gleichen Ordner ebenfalls vorhanden sein:

```
proc mapimport datafile='bkr_2016.shp' out=bkr2016; run;
```

Der Datensatz *cluster* wird mittels `proc tree` im Anschluss an die Clusteranalyse aus den Daten `a.tree` erstellt, hier zum Beispiel mit 12 neuen Regionen:

```
proc tree data=a.tree nclusters=12 noprint out=cluster;
id bkr;
run;
```

Im ersten Versuch (Abb. 1) wird die Karte ohne Bezeichnungen (`bkr2016mid`, `bkr2016own`), aber mit Legende erstellt. Jetzt fehlt der Bezug zu den ursprünglichen BKR.

```
proc sgmap mapdata=bkr2016 maprespdata=cluster;
           choromap cluster / id=bkr mapid=bkr16_id name='A';
           keylegend 'A';
run;
```

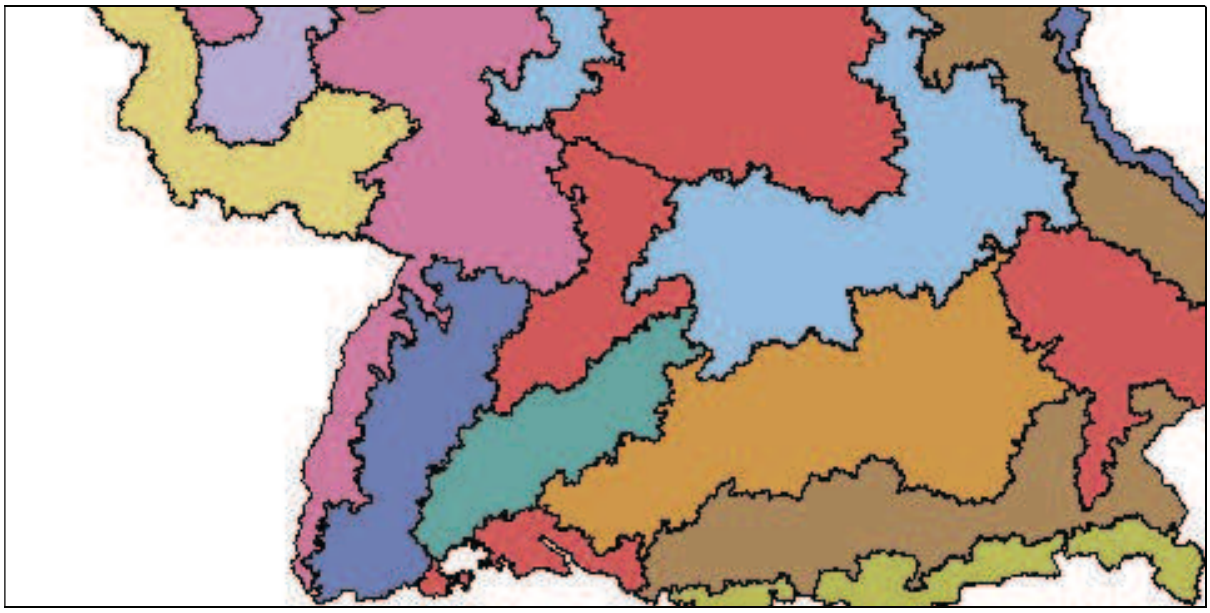

Abbildung 1: Region Süden mit Legende, aber ohne Bezeichnungen der BKR

Um eine Beschriftung der BKR zu erhalten, muss ein Datensatz `plotdata=` angegeben werden. Als Lösung bietet sich an, die arithmetischen „Schwerpunkte“ der BKR zu berechnen. Das funktioniert in 90 % der Fälle gut, bei entarteten Geometrien führt das zu schwer erkennbaren (199) oder auch direkt falschen Ergebnissen (114) (Abb. 2).

```
proc sgmap  mapdata=bkr2016        /* Geometrien der BKR */
            maprespdata=cluster    /* BKR → neue Clusternr */
            plotdata=bkr2016mid;   /* Koordinaten für Text */

            choromap cluster / id=bkr mapid=bkr16_id;
            text x=x y=y text=bkr16_id;
run;
```

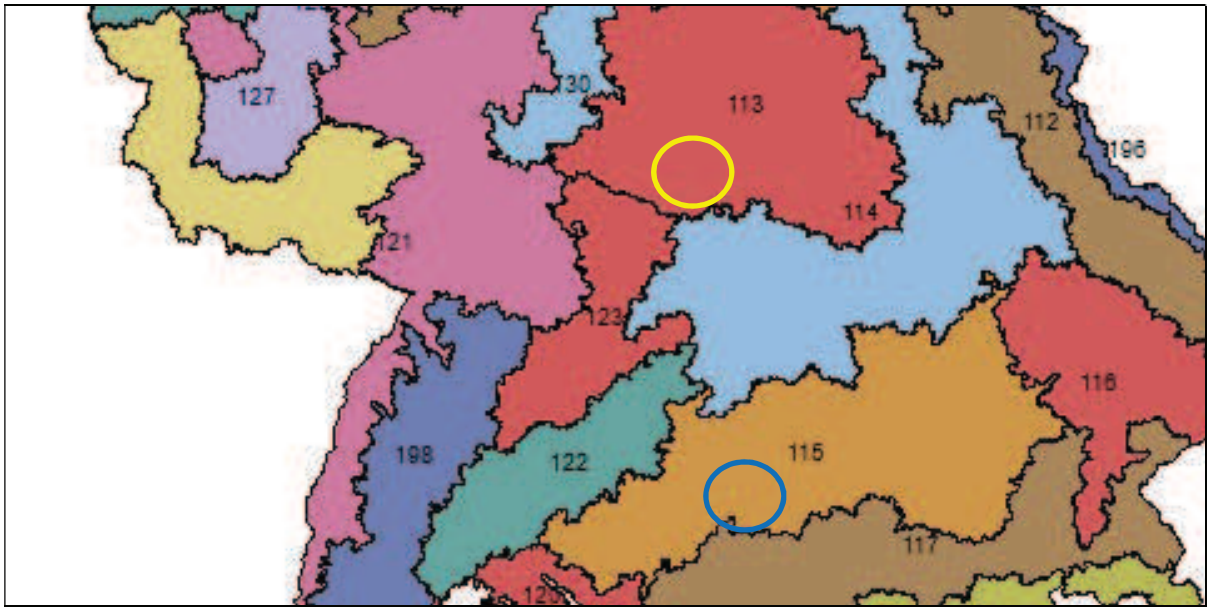

Abbildung 2: Region Süden mit Bezeichnungen auf Basis der „Schwerpunkte“

Für eine vernünftige Beschriftung bleibt als Ausweg nur übrig, eine eigene Datei mit den Koordinaten und Bezeichnungen zu erstellen.
Sieht man sich z. B. die Alpen-Region 199 an, so erkannt man deutlich die 3 Teilflächen. In der Excel-Datei (Tabelle 1) sind dafür 3 Datensätze wie folgt eingetragen:

Tabelle 1: bkr2016own.xlsx (Auszug)

bkr16_id	x	y
…		
199	3660804	5279075
199	3594427	5260498
199	3769590	5294815
…		

Mit diesen Koordinaten sieht der Süden wesentlich besser aus (Abb. 3). Die einzelnen Teile der Region 199 werden separat bezeichnet, die 114 steht jetzt in der richtigen Geometrie:

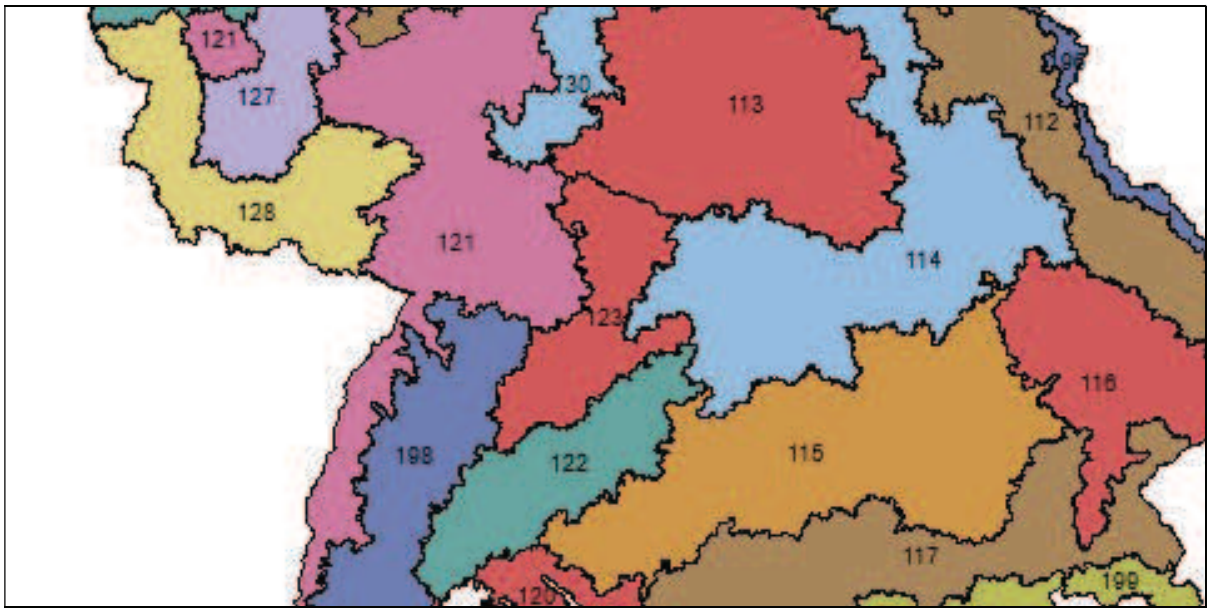

Abbildung 3: Region Süden mit Bezeichnungen auf Basis eigener Koordinaten

Nach dem Importieren der Excel-Datei `bkr2016own.xlsx` nach `bkr2016own` wird die endgültige Karte (Abb. 4) wie folgt gezeichnet:

```
proc sgmap  mapdata=bkr2016       /* Geometrien der BKR */
            maprespdata=cluster   /* BKR → neue Clusternr */
            plotdata=bkr2016own;  /* Koordinaten für Text */

            choromap cluster / id=bkr mapid=bkr16_id;
            text x=x y=y text=bkr16_id;
run;
```

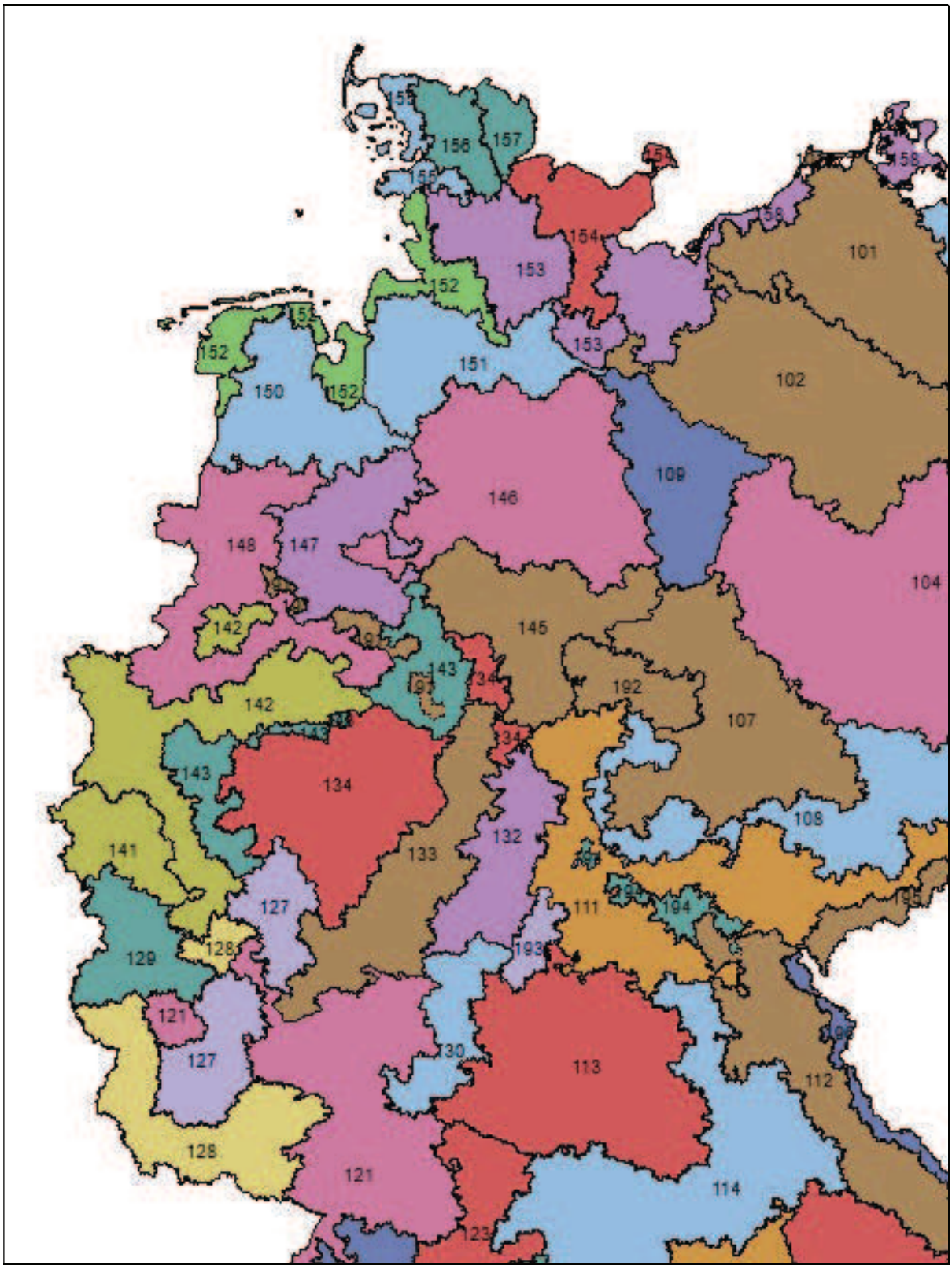

Abbildung 4: 12 „neue“ Regionen (gleiche Farbe) auf Basis der 50 „alten“ BKR

1.2 Farben in der Karte

Im Abschnitt 1.1 wurden die Farben verwendet, so wie sie mit der Prozedur `sgmap` bereitgestellt werden. Für die Wetterkarten benötigen wir allerdings Farbverläufe, so z.B.

von Kalt (Blau) nach Warm (Rot). Wie sehen hier die Möglichkeiten aus, analog zur Prozedur `sgplot`, die Farben selbst zu definieren?

Das JKI bekommt täglich vom Deutschen Wetterdienst Grid basierte Wetterdaten, wie z. B. Temperatur, Niederschlag und Luftgeschwindigkeit, in einem Raster von 1 x 1 km sowohl für den jeweils vorangegangenen Tag als auch die Vorhersagen für die nächsten 7 Tage. Das sind 358303 Kacheln bzw. Geometrien. Für diesen Beitrag wurde ein Raster von 10 x 10 km zugrunde gelegt, mit immer noch 3841 Kacheln. Auf der Basis der Temperatur-Prognose für den 05. Nov. 2018 soll eine farblich „sinnvolle“ Karte erstellt werden. Die Verwendung der Default-Farben erfüllt diesen Zweck jedoch nicht (Abb. 5).

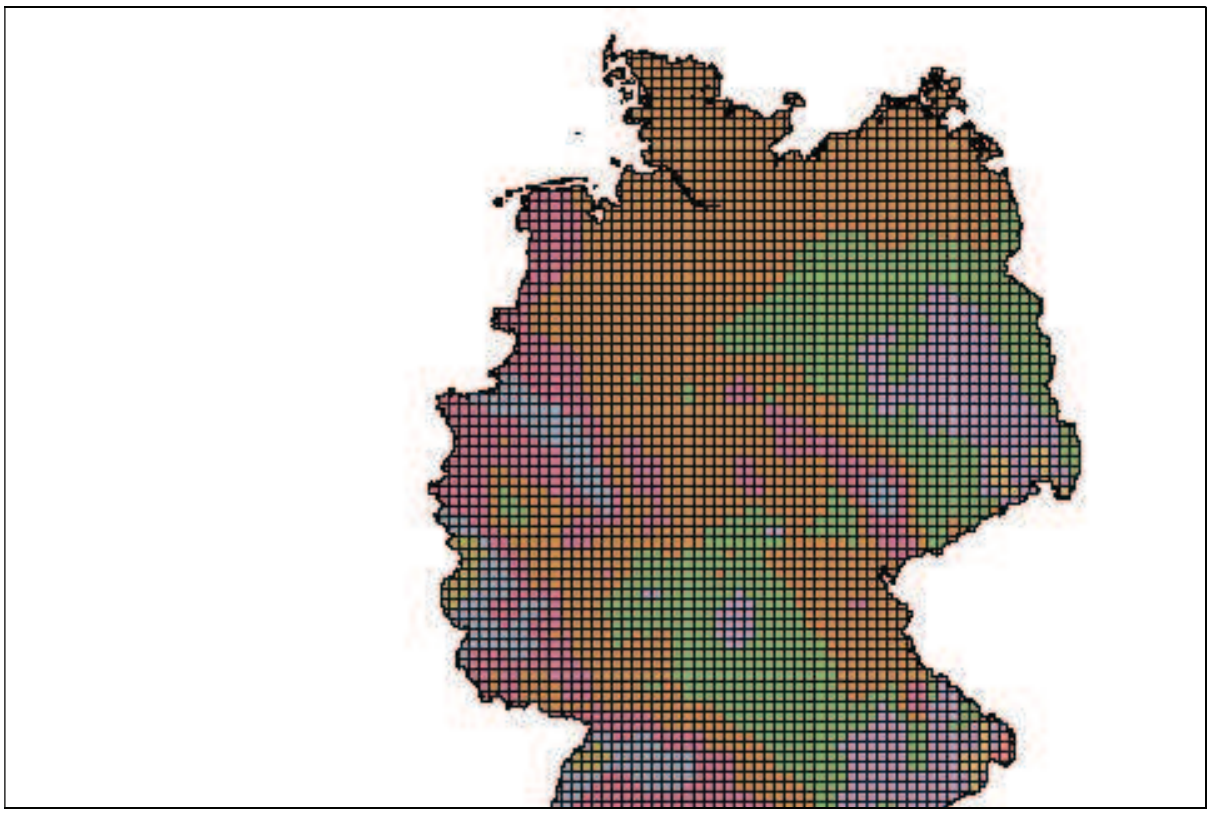

Abbildung 5: Prognosekarte mit Default-Farben

Interessant ist hier der Hinweis im log-Fenster auf den Status der Prozedur `choromap`:

```
NOTE: CHOROMAP statement is pre-production for this release.
```

Der Blick in die Syntax der Prozedur `sgmap` der SAS-Version 9.4 Release M5 gibt keinen Hinweis darüber, mit welchen Statements die Farben zugewiesen werden können. Es bleibt demnach nur übrig, sich vorher mittels `proc template` eigene 16 Farben zu definieren und dieses Template als Style der html-Ausgabe zuzuweisen.
Im Ergebnis (Abb. 6) erkennt man, dass wirklich die ersten 12 eigenen Farben verwendet werden, ab Position 13 allerdings wieder von vorn, mit leicht helleren Farben gebildet aus den neuen, weitergearbeitet wird.

```
/* Define 16 own colors */
proc template;
 define style MyStyle16Colors;
 parent=Styles.Default;
 style GraphData1 from GraphData1 /
            contrastcolor = #0F00FF color = #0F00FF;
 ...
 style GraphData16 from GraphData1 /
            contrastcolor = #FF000F color = #FF000F;
end;
run;

/* Own Colors with more than 12 colors */
ods html style=MyStyle16colors;

proc sgmap mapdata=test10geom maprespdata=prognose;
choromap LUFTTEMPERATUR1 / id=GRID10ID mapid=grid10id name='A';
keylegend 'A';
run;
```

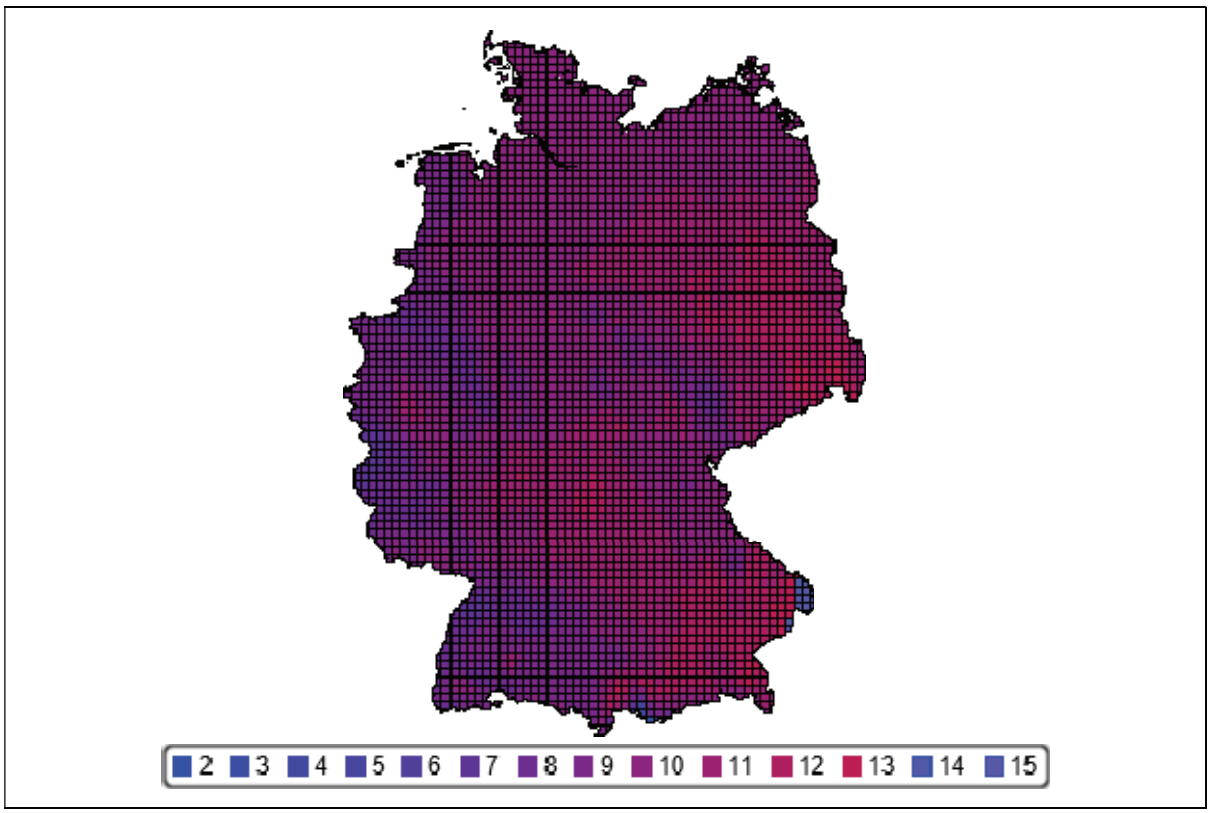

Abbildung 6: Prognosekarte mit 16 eigenen Farben

Es lag die Vermutung nahe, dass SAS nicht mit mehr als 12 selbst definierten Farben umgehen kann. In leichter Abwandlung des Beitrages aus [2] wurde zum Test eine Grafik mit den 16 eigenen Farben erstellt (Abb. 7). Die jeweils ersten 12 Farben aus Abb. 6 (Werte 2-13) und 7 (GraphData1-GraphData12) stimmen überein, danach trennen sich die Vorgehensweisen der Prozeduren `sgmap` und `sgplot`. Während `sgplot` sehr wohl

mit mehr als 12 selbst definierten Farben umgehen kann, ist das in dem pre-production Status der Prozedur `choromap` offensichtlich (noch) nicht der Fall.

Abbildung 7: `sgplot`-Testgrafik mit 16 eigenen Farben (MyStyle16colors)

Eine letzte Chance, eine `choromap` mit mehr als 12 kontinuierlichen Farben zu erhalten, war der Vorgriff auf das Release M6, was allerdings nur mit der University Edition zur Verfügung stand. Der Versuch, das Template mit den 16 eigenen Farben zu nutzen, wurde wie folgt abgewiesen:

```
ERROR: Unzureichende Berechtigungen für Zugriff auf
/opt/sasinside/SASConfig/Lev1/SASApp/sashtml12.htm.
 ERROR: No body file. HTML output will not be created.
```

Damit konnte nur das Default-Verhalten genutzt werden. Das Statement `gradlegend` führte zur Hoffnung auf eine bessere Karte. Aber ob mit `gradlegend` oder ohne, ob mit diskreten Temperaturwerten oder stetigen, ein Einfluss auf die Farbgebung war in keinem Fall möglich. Lediglich die Farbkontinuität war gewährleistet (Abb. 8).

```
proc sgmap mapdata=test10geom maprespdata=prognose;
 choromap LUFTTEMPERATUR1 / id=GRID10ID mapid=grid10id;
 gradlegend / integer;
run;
```

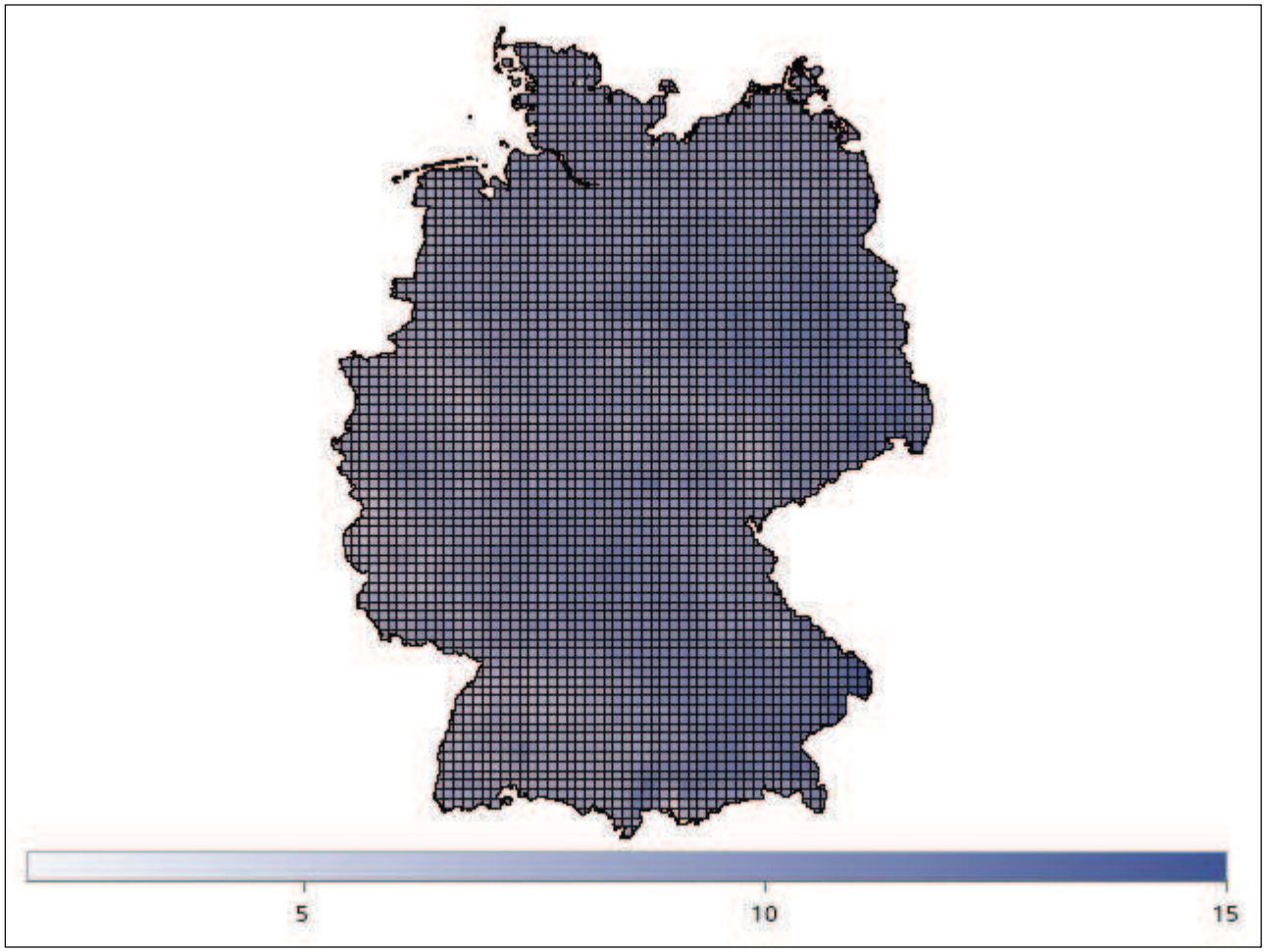

Abbildung 8: `sgmap` im Release M6 (University Edition)

1.3 Mehr als 12 Farben in der Karte

Nachdem der Autor im Abschnitt 1.2 an die Grenzen der Prozedur `choromap` hinsichtlich der Farbgestaltung gestoßen war, blieb als letzter Ausweg noch der Versuch, mittels `sgplot/polygon` Einfluss auf die Farben der Karte zu nehmen. Um das Ergebnis vorwegzunehmen, es gelingt. Darüber hinaus gibt es sogar mehrere Möglichkeiten, die Farben selbst zu bestimmen.
Es müssen bei der Verwendung von `sgplot` zwei wesentliche Voraussetzungen erfüllt sein:

- Damit die Farben in der richtigen Reihenfolge verwendet werden, muss der Datensatz vernünftig sortiert sein, unter Beibehaltung der Geometrien.
- Zum zweiten müssen im Datensatz alle Informationen vorliegen, die Geometrien und die Fakten. Bei `sgmap` kann das in bis zu 3 Datensätzen verteilt vorliegen.

Mit dem richtigen Verhältnis von Breite zu Höhe, entsprechend den Ausmaßen von Deutschland (640 x 876 km), kann die erste Grafik erstellt werden (Abb. 9):

```
ods graphics / width=330px height=450px noborder;

proc sgplot data=test10geom noborder;
```

```
polygon id=grid10id x=x y=y / fill outline;
xaxis display=None;
yaxis display=None;
run;
```

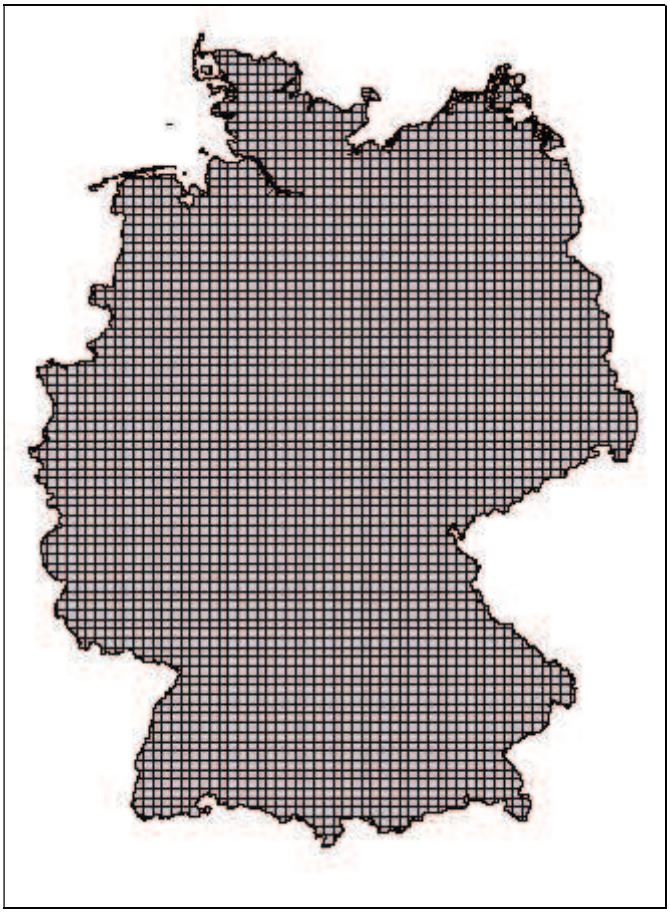

Abbildung 9: `sgplot/polygon` bei einer Auflösung von 330 x 450 (px)

Mit `proc sql` fügen wir die Geometrien und die Temperaturwerte, richtig sortiert, in einen neuen Datensatz zusammen:

```
proc sql;
create table prognose_polygon as
select a.x, a.y, a.grid10id, b.LUFTTEMPERATUR, b.LUFTTEMPERATUR1
from test10geom a, prognose b
where a.grid10id=b.grid10id
order by b.LUFTTEMPERATUR1, a.grid10id, a.nr asc;
```

Für die Färbung verwenden wir das `group`-Statement auf der Grundlage der diskreten Temperaturen `Lufttemperatur1` und erhalten Abb. 10:

```
proc sgplot data=prognose_polygon noborder;
polygon id=grid10id x=x y=y / group=Lufttemperatur1 fill outline;
keylegend / title='00'x location=outside position=bottom;
xaxis display=None;
yaxis display=None;
run;
```

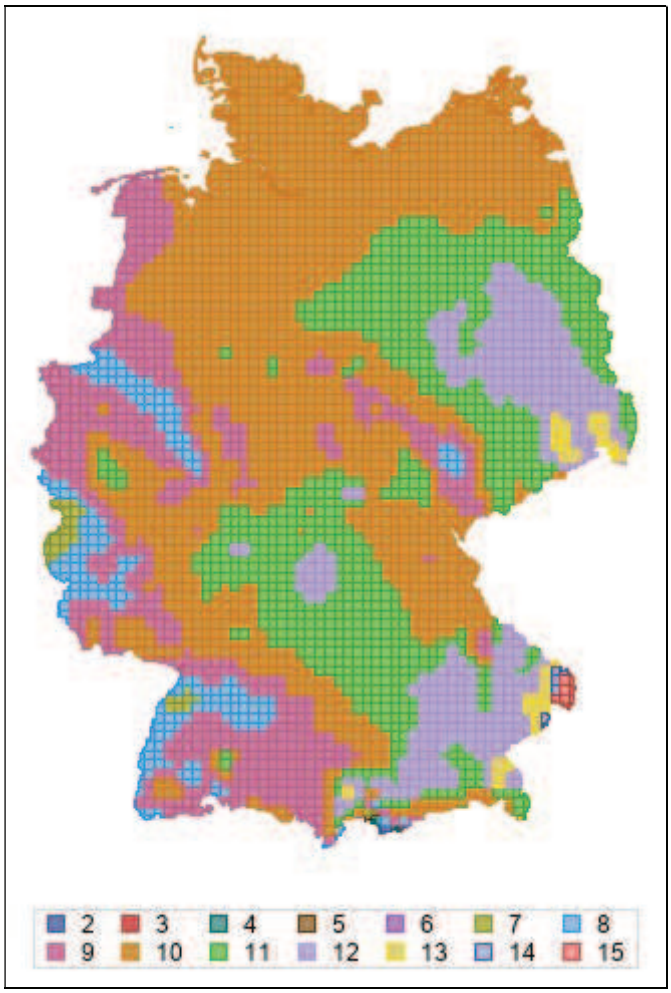

Abbildung 10: `sgplot/polygon` mit `group=Lufttemperatur1`

Definieren wir im nächsten Schritt unsere kontinuierlichen Farben, die erste Farbe immer +16, die zweite Farbe konstant Null, die dritte Farbe immer -16, und weisen diese mittels `styleattrs` zu, erhalten wir erstmalig unsere gewünschte Karte mit den Farben von Kalt zu Warm (Abb. 11):

```
%LET COLORS = %rgbhex(  0, 0, 255) %rgbhex( 16, 0, 239) …
              %rgbhex(208, 0,  47) %rgbhex(224, 0,  31)
;

proc sgplot data=prognose_polygon noborder;
styleattrs datacolors= (&COLORS.) datacontrastcolors=('black');
polygon id=grid10id x=x y=y / group=LUFTTEMPERATUR1 fill outline;
keylegend / title='00'x location=outside position=bottom;
xaxis display=None;
yaxis display=None;
run;
```

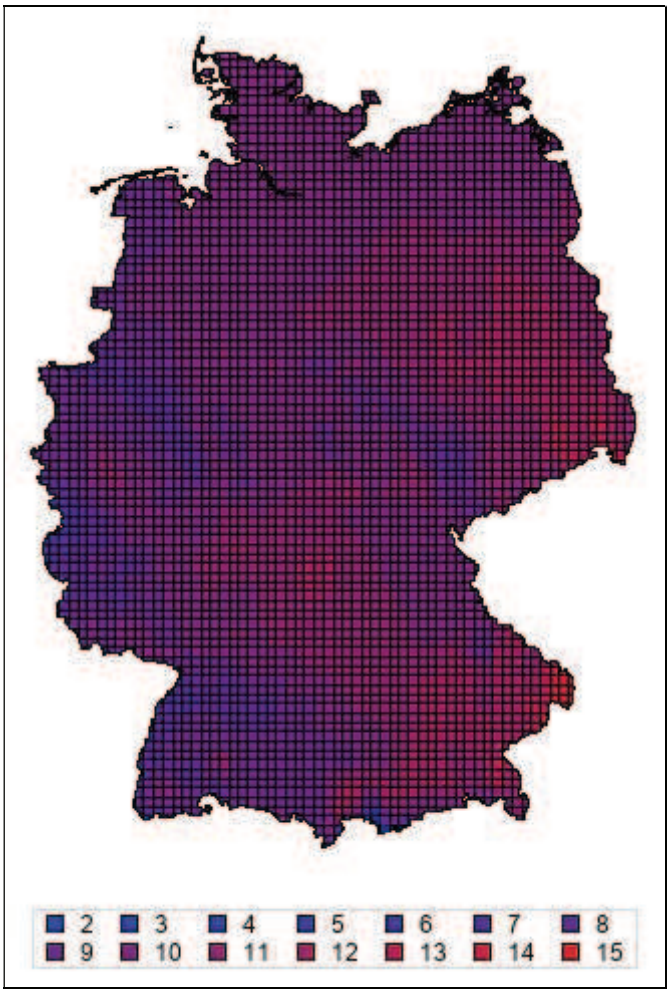

Abbildung 11: `sgplot/polygon` mit `styleattrs datacolors=(&COLORS.)`

Anstelle von `group=` steht als Alternative auch `colorresponse=` zur Verfügung. Dabei können wir ein eigenes Farbschema mittels `colormodel=(&COLORS.)` vorgeben (Abb. 12), nur Start- und Endwert oder Start-, Mittel- und Endwert mit `colormodel=(blue yellow red)` (Abb. 13). In beiden Fällen verwenden wir die stetigen Temperaturen `LUFTTEMPERATUR` als Response-Variable.
Ein Farbschema mittels Start- und Endwert vereinfacht auf jeden Fall die kontinuierliche Färbung und ist bei vielen Farben der manuellen Definition vorzuziehen.

```
proc sgplot data=prognose_polygon noborder;
  polygon id=grid10id x=x y=y / fill outline
     colorresponse=LUFTTEMPERATUR
     /* Abb. 12:      colormodel=(&COLORS.)            */
     /* Abb. 13: */   colormodel=(blue yellow red);
  xaxis display=None;
  yaxis display=None;
  gradlegend / notitle position=bottom;
run;
```

Abbildung 12: `colormodel=(&COLORS.)`

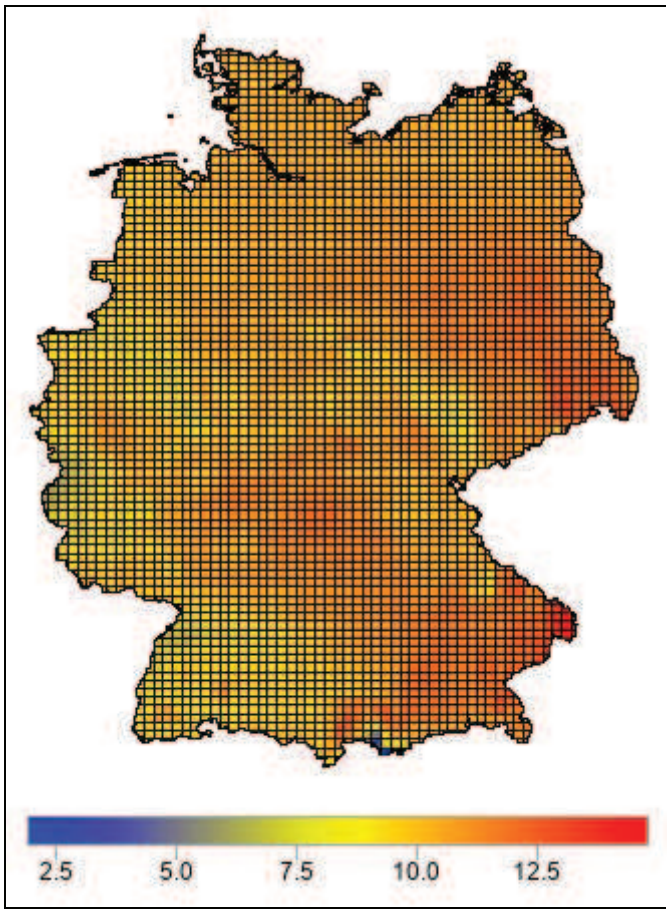

Abbildung 13: `colormodel=(blue yellow red)`

1.4 Ein (kurzer) Vergleich sgmap/choromap vs. sgplot/polygon

Welches der beiden Varianten, eine Karte zu erstellen, zu färben und zu beschriften, ist nun die bessere?

Für die Kombination `sgplot` mit `polygon` spricht die aktuell wesentlich bessere Möglichkeit, Farben zu verwenden. Der große Nachteil ist die Nicht-Einhaltung der Proportionen. Die Polygone orientieren sich an der Grafikfläche. Demzufolge wird die Karte u. U. unproportional skaliert. Hat man einmal manuell die richtige Proportion mittel `ods graphics` vorgegeben, wird diese wiederum durch Legenden unten oder rechts, ein- oder mehrzeilig/-spaltig verändert. Hier ist eine ständige Anpassung erforderlich.

Für `sgmap/choromap` spricht auf jeden Fall die richtige Projektion, unabhängig von der Grafikgröße. Auch die Möglichkeit, weltweit verfügbare ESRI-Kartendienste einzubinden, ist ein großer Vorteil von `sgmap`. Der Nachteil der Farbgestaltung bei mehr als 12 Farben sollte, so die Hoffnung des Autors, demnächst korrigiert werden.

1.5 Programme

Alle in diesem Beitrag verwendeten Programme und Beispiel-Daten stehen zum Download bereit unter

http://sf.julius-kuehn.de/sas/2019

Die aktuell gültigen BKR sind über den BKR-WFS-Dienst des JKI nutzbar:

https://geoservices.julius-kuehn.de/geoserver/bkr/wfs

Die Ausgansdaten `a.tree` für die Prozedur `tree` wurden per Randomisation aus der tatsächlich durchgeführten hierarchischen Clusteranalyse gebildet. Sie dienen hier nur zur Demonstration. Auch die Zahl der neuen Regionen ist fiktiv.

Literatur

[1] Dietmar Roßberg, Volker Michel, Rudolf Graf und Ralf Neukampf (2007): Definition von Boden-Klima-Räumen für die Bundesrepublik Deutschland. Nachrichtenbl. Deut. Pflanzenschutzd., **59** (7), S. 155–161

[2] Warren F. Kuhfeld: Displaying all of the output styles.
URL: https://blogs.sas.com/content/graphicallyspeaking/2018/08/17/displaying-all-of-the-output-styles/