

Jürgen Symanzik

Micromap Plots in R

*A Step-By-Step Approach to Visualize Spatial Data via Linked
Micromaps, Conditioned Micromaps, and Comparative
Micromaps in R*

To my family,
Natascha & Alex

Contents

List of Figures	ix
List of Tables	xv
Preface	xvii
About the Author	xix
1 An Introduction to Micromaps	1
JÜRGEN SYMANZIK	
1.1 Introduction	1
1.2 Linked Micromap Plots	2
1.2.1 Historical Development	2
1.2.2 Design, Features, and Interpretation	3
1.2.3 Software and Online Developments	8
1.2.4 Challenges and Open Research Questions	10
1.3 Conditioned Micromaps	10
1.3.1 Historical Development	10
1.3.2 Design, Features, and Interpretation	10
1.3.3 Software and Online Developments	10
1.3.4 Challenges and Open Research Questions	10
1.4 Comparative Micromaps	10
1.4.1 Historical Development	10
1.4.2 Design, Features, and Interpretation	11
1.4.3 Software and Online Developments	11
1.4.4 Challenges and Open Research Questions	11
1.5 Outlook on the Book Chapters	11
1.6 How to Use this Book and the Accompanying R Code, Webpage, and R Package	11
1.7 INSTRUCTIONS FOR CHAPTER AUTHORS: Use of Bookdown	11
1.8 Use of Bookdown	11
1.9 Creating Figures and Tables	12
1.10 Indexing	13
1.11 Citations and References	13
1.12 Micromap Examples	14
1.13 Alternative Creation and Inclusion of Figures	18
1.14 Translating the Book Chapters to pdf and html Output	22
1.15 External Data Files and Additional R Code	24
1.16 Open Bookdown Questions	25
1.17 Initial Chapter 2 Outline (Merge into Chapter 2 Introduction at the End - Mostly Done)	25

2 Linked Micromap Plots via the <code>micromap</code> R Package	31
JÜRGEN SYMANZIK, MARCUS W. BECK, MICHAEL G. McMANUS	
2.1 Introduction	31
2.2 Steps to Create a Linked Micromap Plot with the <code>micromap</code> R Package .	32
2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C.	34
2.3.1 Identifying and Geoprocessing of Spatial Boundary Data	34
2.3.2 Linking Spatial Boundary Data and Statistical Data	35
2.3.3 Creating a Draft Linked Micromap Plot	37
2.3.4 Refining the Linked Micromap Plot	38
2.4 Example 2: A Linked Micromap Plot for Watersheds in West Virginia	52
2.4.1 Identifying and Geoprocessing of Spatial Boundary Data	53
2.4.2 Linking Spatial Boundary Data and Statistical Data	56
2.4.3 Creating a Draft Linked Micromap Plot	58
2.4.4 Refining the Linked Micromap Plot	61
2.5 Summary and Further Reading	69
3 Linked Micromap Plots via the <code>micromapST</code> R Package	73
LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR., DANIEL B. CARR	
3.1 Introduction	73
3.2 Basic Plot Layout and Function Call	75
3.3 Display Options	81
3.4 Mapping beyond U.S. states	86
3.5 Summary and Further Reading	89
4 Modifying Shapefiles for Use in Micromaps	93
BRADEN PROBST	
4.1 Introduction	93
4.2 Outline	93
4.3 Main	94
4.4 Code Chunks China	94
4.5 Summary and Further Reading	106
5 Using Your Own Boundary Files in the <code>micromapST</code> R Package	109
JAMES BLACKWOOD PEARSON, JR., LINDA WILLIAMS PICKLE, DANIEL B. CARR	
5.1 Introduction	109
5.2 Boundary Files Included in <code>micromapST</code>	110
5.3 Creating Border Groups for Use in <code>micromapST</code>	111
5.3.1 Introduction	111
5.3.2 Challenges	111
5.3.3 Steps in the Process	112
5.4 Building a Border Group for <code>micromapST</code>	113
5.4.1 Build a U.S. Name Table	113
5.4.2 Process the U.S. Shapefile	118
5.4.3 Link the Name Table and Shapefile	119
5.4.4 Apply <code>BuildBorderGroup</code> to Create Internal Data Frames	120
5.4.5 Compare U.S. Results to Default Caricature Map	121
5.5 Linked Micromap Plots for a Large Number of Areas	122
5.5.1 Creating Long Linked Micromap Plots	123
5.5.2 Aggregating counties to Area Development Districts in Kentucky . .	126
5.5.3 Building the New Border Group for the Aggregated Kentucky ADDs	129

5.6	Summary and Further Reading	132
6	Adding New Plot Types to Linked Micromap Plots (such as Arrow Plots, Line Charts, and Scatterplots)	135
	SARAH SCHWARTZ, JÜRGEN SYMANZIK	
6.1	Introduction	135
6.2	Outline	135
6.3	Example 1: Adding Vignette Example	141
6.4	Example 2: Adding Bar Charts as a New Plot Type	154
6.5	Example 3: Adding Arrow Plots for Differences as a New Plot Type	160
6.6	Example 4: Adding Scatterplots as a New Plot Type	167
6.7	Main	192
6.8	Summary and Further Reading	192
7	Enhanced Glyphs Available in the micromapST R Package	193
	LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR., DANIEL B. CARR	
7.1	Introduction	193
7.2	Time Series Data	195
7.3	Scaled Response Data	200
7.4	Scatter Plots	206
7.5	Summary and Further Reading	207
8	Linked Micromap Plots for Point Locations	211
	MARTIN HOLDREGE, JÜRGEN SYMANZIK	
8.1	Introduction: The Challenge of Displaying Point Data Using Micromaps	211
8.2	Outline of Steps to Create Linked Micromap Plots for Point Locations	212
8.3	Plotting Large Cities in Asia	213
8.3.1	Loading Point Location Data	213
8.3.2	Generating Polygons around Points	214
8.3.3	Creating a Draft Linked Micromap Plot	216
8.3.4	Combining Point Locations with a Background Map	216
8.3.5	Refined Linked Micromap Plot Showing the Base Map	219
8.4	County Seats in Utah	221
8.4.1	Loading Point Locations and Base Map Data	221
8.4.2	Adjusting Point Locations	222
8.4.3	Preparing Points and Base Map	224
8.4.4	Creating a Linked Micromap Plot	225
8.5	Premier League Football Stadiums in England and Wales	227
8.5.1	Obtaining Point Locations	227
8.5.2	Acquiring the Underlying Map	229
8.5.3	Combining with Points	229
8.5.4	Creating a Draft Linked Micromap Plot	230
8.5.5	Refined Linked Micromap Plot with Adjusted Stadium Locations	230
8.6	Summary and Further Reading	234
9	Web-based Linked Micromap Plots Using Shiny	237
	MARCUS W. BECK	
9.1	The Need for Web Applications	237
9.2	Existing Examples of Micromap Web Applications	238
9.3	Shiny as an Open Source Tool for Web Applications	239
9.3.1	What is Shiny?	239

9.3.2 Minimal Micromap Example	244
9.3.3 Getting Shiny Online	248
9.4 Summary and Further Reading	250
10 Conditioned Micromaps	253
BRENT D. MAST	
10.1 Introduction	253
10.2 Main	253
10.3 WV Examples from Brent Mast	272
10.4 WV Conditioned Choropleth Maps with Factors	277
10.5 DC Examples from Brent Mast	284
10.6 NJ Examples from Brent Mast	290
10.7 Further Reading	305
11 Comparative Micromaps	309
BRENT D. MAST	
11.1 Introduction	309
11.2 Main	309
11.3 Further Reading	310
12 Back to the Roots: Non-traditional Micromaps (for Viewing Patterns of Scientific Posters)	313
CHUNYANG LI	
12.1 Introduction	313
12.2 Data Processing and Preparation	314
12.2.1 Eye Tracking Data	314
12.2.2 Shapefile Preparation: AOI Definition	314
12.2.3 Eye Tracking Data Feature Extractions	315
12.3 Linked Microposter Plot Creation	316
12.4 Summary and Further Reading	341
13 Applications for the Environmental Field	345
MICHAEL G. McMANUS, MARCUS W. BECK, JÜRGEN SYMANZIK	
13.1 Introduction	345
13.2 Recent Geovisualizations with Environmental Data	346
13.3 Spatial Surveys and Linked Micromaps	353
13.4 Summary and Further Reading	355
14 Application of Linked Micromaps in the Medical Field	359
LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR.	
14.1 Introduction	359
14.2 Background	359
14.2.1 Data	360
14.2.2 Preliminary U.S. Analysis	360
14.3 State-level Analysis	363
14.4 Applying Linked Micromap Plots to Sub-state Data	373
14.5 Summary and Further Reading	379
Appendix	383
A More to Say	383

<i>Contents</i>	vii
A.1 Micromap Reference Overview	385
Bibliography	395
Index	409
.	409

List of Figures

1.1	Mock-up linked micromap plot, based on the 21 counties of New Jersey. <code>StatVar1</code> is used as the sorting variable.	4
1.2	Mock-up linked micromap plot, based on the 21 counties of New Jersey. <code>StatVar2</code> is used as the sorting variable. Further changes are the use of two-ended cumulative maps, the reduced number of perceptual groups, and the introduction of a median row.	6
1.3	Mock-up linked micromap plot, based on the 21 counties of New Jersey. <code>StatVar4</code> is used as the sorting variable. The maps are placed on the right.	8
1.4	Mock-up scatterplot of the four statistical variables shown in the three previous linked micromap plots, based on the 21 counties of New Jersey.	9
1.5	A trivial scatterplot of the cars dataset.	12
1.6	Here is a first micromap example. Note that it occupies 90 percent of the page width. The rows are far too much condensed.	15
1.7	And here is a second micromap example. This uses the default output settings. This still looks very bad.	17
1.8	And now the third (revised) micromap example. Note that this specifies a width and height for the figure. This may be the best solution to scale the micromap plots.	19
1.9	And now the fourth (and final) micromap example.	21
2.1	Workflow to create a linked micromap plot with the micromap R package (Diagram created with the DiagrammeR R package (Iannone, 2022)).	33
2.2	Map representation of the <i>USstates</i> spatial boundary dataset for the United States that frequently is used as the basis for linked micromap plots that are created with the micromap R package.	35
2.3	Draft linked micromap plot, based on the <i>edPov</i> dataset.	39
2.4	First refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are the introduction of a median row instead of the eleventh perceptual group and the reverse ordering of the <i>pov</i> data in the first statistical graphics column.	41
2.5	Second refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are related to the order of the five columns in the plot. Most notable, the map column is shown on the left here.	42
2.6	Third refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are related to perceptual groups with different numbers of subregions and the vertical alignment of rows in the middle of the plot. Also, <code>labels</code> are aligned on the left.	43
2.7	Fourth refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are the use of a divergent brown-blue-green color scheme and the conversion back to the traditional grouping for 51 subregions.	45
2.8	Fifth refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are related to the panel specific attributes.	48

2.9	Sixth (and final) refined linked micromap plot, based on the <i>edPov</i> dataset. Main changes are related to labeling, the coloring of perceptual groups above and below the median row, and the column spacing.	51
2.10	Map representation of the <i>WV_Watershed</i> spatial boundary dataset for the 25 watersheds in West Virginia that is used as the basis for the linked micromap plots for option (i).	55
2.11	Map representation of the <i>WV_Watershed</i> spatial boundary dataset for the 25 watersheds in West Virginia that is used as the basis for the linked micromap plots for option (ii). It is identical to the one for option (i) shown in Figure 2.10.	57
2.12	Draft linked micromap plot for option (i), based on the previously created <i>wv_data</i> and <i>wv_polys_table</i> data frames for the West Virginia watershed dataset.	59
2.13	Draft linked micromap plot for option (ii), based on the West Virginia watershed dataset. In contrast to the previous figure, this figure has been created directly from the external shapefiles without using the <i>create_map_table()</i> function first.	60
2.14	First refined linked micromap plot for option (i), based on the <i>wv_data</i> data frame. Main changes are the introduction of abbreviated subregion names and initial boxplots and dotplots with confidence bounds in the two statistical graphics columns.	63
2.15	Second (and final) refined linked micromap plot for option (i), based on the <i>wv_data</i> data frame. Main changes are related to sorting, coloring, spacing, and labeling. An overall statistics (or criteria) line has been added to the second statistical graphics column.	65
2.16	Second (and final) refined linked micromap plot for option (ii), based on the West Virginia watershed dataset. In contrast to the previous figure, this figure has been created directly from the external shapefiles without using the <i>create_map_table()</i> function first. It is identical to the one for option (i) shown in Figure 2.15.	68
3.1	Cumulative number of covid-19 cases reported by U.S. states on Nov. 20, 2020.	77
3.2	Cumulative number of covid-19 cases reported by U.S. states on Nov. 20, 2020.	79
3.3	Cumulative number of covid-19 cases per 1000 by state on Nov. 20, 2020, and percent who usually wore a mask in Sep. 2020.	80
3.4	Cumulative number of covid-19 cases by state on Nov. 20, 2020, and percent who usually wore a mask in Sep 2020, with optional error bars, cumulative map shading and full state names.	83
3.5	Cumulative number of covid-19 cases by state on Nov. 20, 2020, and percent who usually wore a mask in Sep 2020, in gray scale with expanded labels and bar glyph.	85
3.6	Percent of Maryland residents with income < 150% of the federal poverty level and percent with 4+ years of college, by county.	88
3.7	Percent of Maryland residents with income < 150% of the federal poverty level and percent with 4+ years of college by county, displayed in smaller perceptual groups.	90
4.1	Here is a first micromap example for this chapter. Note that the figure is formatted in a slightly more meaningful way this time.	95

4.2	Chunk 1.	97
4.3	Chunk 1.	99
4.4	Chunk 1.	101
4.5	Chunk 1.	104
4.6	Chunk 1.	107
5.1	U.S. state boundaries plotted with latitude/longitude coordinates as provided by the U.S. Census Bureau (United States Census Bureau, 2021).	112
5.2	U.S. states maps: Simplified (A), after moving and scaling (B), final (C) and micromapST default map (D).	122
5.3	U.S. states maps smoothed to retain varying percents of vertices from the original shapefile: 100% (No smoothing) (A), 2.5% (B), 1% (C), 0.5% (D), 0.2% (E), 0.1% (F).	123
5.4	Kentucky counties, smoothed by <code>BuildBorderGroup</code> .	125
5.5	Invasive cervical cancer incidence rates in Kentucky Counties, 2014-2018, age-adjusted to 2000 standard million population.	127
5.6	Kentucky smoothed Area Development Districts.	130
5.7	Invasive cervical cancer incidence rates in Kentucky Area Development Districts, 2014-2018, age-adjusted to 2000 standard million population.	131
6.1	Linked micromap plot, based on the West Virginia watershed dataset, that shows all pre-defined plot types of the micromap R package (Payton & Olsen, 2015).	140
6.2	Arrow plot from vignette.	143
6.3	Arrow plot from vignette.	145
6.4	Arrow plot from vignette.	146
6.5	Arrow plot from vignette.	147
6.6	Arrow plot from vignette.	149
6.7	Arrow plot from vignette.	151
6.8	Arrow plot from vignette.	152
6.9	Arrow plot from vignette.	155
6.10	Bar chart.	158
6.11	Bar chart.	161
6.12	Arrow plot for differences.	164
6.13	Arrow plot for differences.	166
6.14	Scatterplot.	183
6.15	Scatterplot.	185
6.16	Scatterplot.	189
6.17	Scatterplot.	191
7.1	Time Series plots of cumulative covid case rates by month, Feb 2020 - Apr 2022, sorted by March 2021 (median date) rates.	198
7.2	Time Series arrow plots of cumulative covid case rates, Feb 2020 - Mar 2021 and Mar 2021 - Apr 2022, and county rate boxplots.	201
7.3	Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade, measured by average scores and achievement category.	203
7.4	Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade, centered at Not Proficient vs. At Least Proficient.	205
7.5	Scatter plot of percent of residents with income < 150% federal poverty level and percent with 4+ years of college, sorted by percent who attended college 4+ years.	208

8.1	Latitude and longitude of 10 large cities in Asia. Note, in this example the point locations are far apart, and therefore the final map will not have overplotting issues.	214
8.2	Latitude and longitude of 10 large cities in Asia, now plotted with circles around them.	215
8.3	Map showing the 10 largest cities in Asia. Note that the locations of the cities are shown, but there are no country polygons to indicate the location of the cities.	216
8.4	Map showing the 10 largest cities in Asia. Note that the locations of the cities are shown. In addition, country polygons are shown as well to indicate the location of the cities in Asia.	218
8.5	Draft linked micromap plot showing the 10 largest cities in Asia and their population. Note that the locations of the cities are shown. In addition, country polygons are shown as well to indicate the location of the cities in Asia.	219
8.6	Refined linked micromap plot showing the 10 largest cities in Asia and their population.	221
8.7	Locations of counties and county seats (points), in Utah.	223
8.8	Original locations of county seats in blue, and adjusted locations (county centroids) in red.	224
8.9	Circles drawn around the locations of county seats. Note that overlap between individual circles is fairly limited, because some locations have been adjusted.	225
8.10	Micromap showing the elevations and population at the last Census of county seats in Utah. Some locations have been adjusted to decrease overplotting.	228
8.11	Micromap showing stadium locations. Note that these maps are not very readable due to the overlap in locations	231
8.12	Micromap showing stadium locations. Here the stadium locations have been moved slightly so that complete overlap of points does not occur. Micromaps can contain two label columns, as seen here where both the stadium name and team name are provided.	235
9.1	A minimal working shiny application in RStudio showing code for the user interface and server. The application can be run in RStudio by selecting "Run App".	240
9.2	A minimal working shiny application with additional components in the server and user interface that produces a histogram with options to select the sample size.	242
9.3	A representation of reactivity each time the user input is changed to select a different sample size for the histogram.	242
9.4	A minimal working shiny application using the micromap package.	245
9.5	A screenshot of a shiny app with linked micromaps from the Virginia Department of Environmental Quality, available at https://evjones.shinyapps.io/FreshwaterProbMonEDA/	249
10.1	Trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.	269
10.2	First modified trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.	270
10.3	Second modified trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.	271
10.4	Trivial 2-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.	272

10.5 Conditioned choropleth map 1.	274
10.6 Modified conditioned choropleth map 1.	275
10.7 Conditioned choropleth map 2.	277
10.8 Modified conditioned choropleth map 2.	278
10.9 Conditioned choropleth map 1 for 1 categorical variable.	279
10.10 Conditioned choropleth map 2 for 1 categorical and 1 quantitative variable.	280
10.11 Conditioned choropleth map 3 for 1 quantitative and 1 categorical variable (order switched).	281
10.12 Conditioned choropleth map 4 for 2 categorical variables.	282
10.13 Conditioned choropleth map 5 for 2 categorical variables (with more than 2 factor levels).	284
10.14 Conditioned choropleth map 6 for 2 categorical variables (with more than 2 factor levels). Also with externally provided text information.	285
10.15 Conditioned choropleth map 3.	287
10.16 Modified conditioned choropleth map 3.	288
10.17 Conditioned choropleth map 8.	290
10.18 Modified conditioned choropleth map 8.	291
10.19 Conditioned choropleth map 4.	294
10.20 Modified conditioned choropleth map 4.	295
10.21 Conditioned choropleth map 5.	297
10.22 Modified conditioned choropleth map 5.	298
10.23 Conditioned choropleth map 6.	303
10.24 Modified conditioned choropleth map 6.	304
10.25 Conditioned choropleth map 7.	306
10.26 Modified conditioned choropleth map 7.	307
11.1 Here is a first micromap example for this chapter. Note that the figure is formatted in a slightly more meaningful way this time.	311
12.1 Poster with twelve defined AOIs.	316
12.2 Microposters column of the linked microposter plot.	328
12.3 Similar to Figure 12.2, now with the color-coded legend and labels column added.	331
12.4 Similar to Figure 12.3, now with two dotplot columns added.	339
12.5 Similar to Figure 12.4, now with one boxplot column added.	341
12.6 Similar to Figure 12.5, now with the main title added.	342
13.1 A linked micromap of stream condition for 11 basins from spatially balanced probabilistic surveys done by Virginia Department of Environmental Quality. Higher scores for Virginia Stream Condition Index (VSCI) and Virginia Coastal Plains Macroinvertebrate Index (VCPMI) indicate better ecological condition of the streams. Sample size, n, is indicated for each basin and median estimates and their interquartile ranges are shown in the statistical graphics column. The dashed vertical line at 60 is the threshold for biological impairment.	347

13.2 A linked micromap summarizing results for the pesticide Imidacloprid for 29 Florida basins from spatially balanced probabilistic surveys of freshwater bodies by Florida Department of Environmental Protection. Data are sorted by percentage of samples in a basin having detectable concentrations of the pesticide. The second statistical graphics column shows percent of agricultural and urban land cover in a basin. The third statistical graphics column displays boxplots of concentrations of the pesticide, with single vertical bars indicating samples having concentrations below instrument detection limits.	350
13.3 Map of Oregon lakes, with lakes represented as their centroids. The simple features object of the 3190 lakes is the sample frame from which the Generalized Random Tessellation Stratified function draws an n = 50 lakes for a spatially balanced probabilistic survey.	354
13.4 Oregon lakes Sample Frame & Equal Probability Sites in Projected Coordinates. Red points are the GRTS sample and blue points are the sample frame.	355
13.5 A linked micromap of specific conductance, a measure of the ionic content of the sampled rivers and streams, for 25 basins from spatially balanced probabilistic stream surveys done by West Virginia Department of Environmental Protection. The first statistical panel displays the median and 95% confidence limits based on the default variance local neighborhood estimator. The second statistical panel displays the median and 95% confidence limits based on specifying the simple random sample (SRS) variance estimator.	356
14.1 U. S. Colorectal cancer mortality rates, 1990-2022 by age group <50 and 50+	361
14.2 U. S. Colorectal cancer mortality rates among white non-Hispanics ages <50, 1990-2022 for males, females and both	362
14.3 Time trends and recent rates of colorectal cancer mortality among whites ages <50 by state	365
14.4 Time trends of colorectal cancer mortality, sorted by percent change, among whites ages < 50, by state	367
14.5 Colorectal cancer mortality among whites ages < 50, and related lifestyle factors	368
14.6 Mortality rates and screening for colorectal cancer, among whites ages < 50	370
14.7 Colorectal cancer mortality among whites ages < 50 and Pct Poverty, Pct Obese, Pct No Exercise	371
14.8 Colorectal cancer mortality among whites ages < 50 and Pct Poverty, Pct Obese, Pct No Exerercise	372
14.9 Colorectal cancer mortality among whites ages < 50, scatter plots of rates and main risk factors	374
14.10 Rates and time trends of colorectal cancer mortality among whites ages <50 by Area Development Districts in Kentucky	377
14.11 Rates and risk factors of colorectal cancer mortality among whites ages <50 by Area Development Districts in Kentucky	378
14.12 Appalachian counties in Kentucky	379

List of Tables

1.1	Full symmetry partitionings with targeting groups of size 5. The left column (Number) contains the number of regions. The middle column (Partitioning 1) puts smallest counts in the middle. Full symmetry alternatives that avoid small counts appear in the right column (Partitioning 2). Abandoning full symmetry can lead to fewer panels. The table ends with 51 regions (the number of U.S. states plus Washington, D.C.), but it can be easily extended.	5
1.2	Table originally published in Symanzik and Carr (2008). A table of the iris data.	13
5.1	Mminimum information needed in the Name Table: First 15 U.S. states.	114
5.2	Contents of the Name Table	116
5.3	Final Name Table for the first 15 U.S. states.	117
7.1	Glyphs available in micromapST and the specification of input data by column name, which indexes the input data frame. NA indicates that this column parameter is not used for that glyph.	194
12.1	An example of a shapefile for a linked microposter plot that shows x and y coordinates for two AOIs (here the Logo and the Title of the poster).	315

Preface

This is the preface for the `Micromap Plots in R` book.

Why Read this Book

You will learn a lot about all types of micromap plots.

Structure of the Book

Chapter 1 provides a general overview on all types of micromap plots and provides an overview of the following chapters of the book.

Chapter 2 introduces the `micromap` R package. Chapter 3 introduces the `micromapST` R package.

Software Information and Conventions

I used the `knitr` R package (Xie, 2022b) and the `bookdown` R package (Xie, 2022a) to compile the book. Also see the (Xie, 2015) book.

My R session information is shown below:

```
xfun::session_info()

## R version 4.5.1 (2025-06-13 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 10 x64 (build 19045)
##
## Locale:
##   LC_COLLATE=English_United States.utf8
##   LC_CTYPE=English_United States.utf8
##   LC_MONETARY=English_United States.utf8
##   LC_NUMERIC=C
##   LC_TIME=English_United States.utf8
##
```

```
## Package version:  
##   base64enc_0.1.3   bookdown_0.43    bslib_0.9.0      cachem_1.1.0  
##   cli_3.6.5         compiler_4.5.1   digest_0.6.37    evaluate_1.0.4  
##   fastmap_1.2.0     fontawesome_0.5.3  fs_1.6.6       glue_1.8.0  
##   graphics_4.5.1    grDevices_4.5.1  highr_0.11      htmltools_0.5.8.1  
##   jquerylib_0.1.4   jsonlite_2.0.0   knitr_1.50      lifecycle_1.0.4  
##   memoise_2.0.1     methods_4.5.1   mime_0.13       R6_2.6.1  
##   rappdirs_0.3.3    rlang_1.1.6     rmarkdown_2.29  rstudioapi_0.17.1  
##   sass_0.4.10       stats_4.5.1     tinytex_0.57    tools_4.5.1  
##   utils_4.5.1       xfun_0.52      yaml_2.3.10
```

Package names are in bold text (e.g., **rmarkdown**), and inline code and filenames are formatted in a typewriter font (e.g., `knitr::knit('foo.Rmd')`). Function names are followed by parentheses (e.g., `bookdown::render_book()`).

Acknowledgments

A lot of people helped me when I was writing the book. More to come later.

Jürgen Symanzik
Logan, Utah, USA

About the Author

Jürgen Symanzik is a Professor in the Department of Mathematics and Statistics at Utah State University in Logan, Utah, USA.

I need to check with CRC Press how to introduce all the chapter authors, e.g., via a few lines of text or by listing them by name and indicating the affiliations (and e-mail addresses).

1

An Introduction to Micromaps

JÜRGEN SYMANZIK

Welcome to the *Micromap Plots in R Book*. In this chapter, we provide a general overview of micromap visualizations. We present the three major classes of micromaps and we introduce the nomenclature used throughout the book. For each of the three major classes of micromaps, we look at the historical developments, their design, features, and interpretation, and the primary software and online developments for their construction. Also provided are an outlook on the remaining chapters of this book and how to best use the accompanying R code, webpage, and newly developed **micromapExtra** R package.

1.1 Introduction

Micromap visualizations are an important tool for the visualization of statistical data in a geographic context, providing a rich context for interpretation. Micromap visualizations are based on numerous small panels that are arranged in specific ways. Apparently, the most important types of panels are those that contain one or multiple series of small maps, the *micromaps*, that are used to show spatial patterns. These maps are often used to link to panels that contain underlying statistical data and region identifiers. There exist three major classes of micromaps, i.e., linked micromap plots, introduced in Section 1.2, conditioned micromaps, introduced in Section 1.3, and comparative micromaps, introduced in Section 1.4. In each of these three sections, we will first look at the historical development of that class of micromaps, then discuss their design, features, and interpretation, and finish with an overview of past and current software and online developments.

Our nomenclature will closely follow the one from Symanzik, Carr, McManus, et al. (2017). Alternatives can sometimes be found elsewhere in the literature, but it should be easy to match alternative terms with the terminology used in this book. As an example, linked micromap plots were originally called *map row plots* (Olsen et al. (1996)). They often have been abbreviated as *LMplots*. Conditioned micromaps initially have been called *conditioned choropleth maps* (Carr, Wallin, et al. (2000)) and have been abbreviated as *CCmaps*.

What follows in Section 1.5 is an outlook on the 14 chapters in this book. Finally, Section 1.6 will provide some recommendations how to best use this book and the accompanying R code, webpage, and the newly developed **micromapExtra** R package.

1.2 Linked Micromap Plots

1.2.1 Historical Development

Linked micromap plots were introduced as the first of the three major classes of micromap visualizations. They were originally presented at the Joint Statistical Meetings (JSM) in Chicago, Illinois, in 1996 (Olsen et al., 1996). A series of applications was published soon thereafter, dealing with unemployment data of the United States and Washington, D.C. (Carr & Pierson, 1996), carbon dioxide (CO_2) emissions in the Organization for Economic Co-operation and Development (OECD) countries and wheat price and labor force statistics in the United States and Washington, D.C. (Carr, Olsen, Courbois, et al., 1998), and of various climate variables in Omernik ecoregions (that were named after James M. Omernik) (Carr, Olsen, Pierson, et al., 1998, 2000).

The early linked micromap plots development was based on ideas from several other visualization developments of the 1980s and 1990s. In particular, their development was heavily inspired by concepts developed for the conversion of statistical tables into graphical displays (Carr, 1994; Carr & Nusser, 1995), the small multiples principle that was made widely popular by Edward R. Tufte (Tufte, 1983, 1990, 1997), as well as early map caricatures, such as Mark S. Monmonier's state visibility map (Monmonier, 1993).

Linked micromap plots quickly gained popularity among researchers at United States (U.S.) Federal Agencies such as the U.S. Department of Agriculture – National Agricultural Statistics Service (USDA–NASS), various branches of the U.S. Environmental Protection Agency (USEPA), the National Cancer Institute (NCI), the U.S. Census Bureau, and the U.S. Bureau of Labor Statistics (BLS). In addition to static linked micromap plots, some of these agencies (in particular the NCI, the USEPA, and the USADA-NASS) developed early interactive and web-based versions of linked micromap plots as discussed in more detail in Symanzik and Carr (2008).

As indicated in Olsen et al. (1996), linked micromap plots “give equal consideration to presenting data in attribute [i.e., statistical] space and in geographic space.” In contrast, traditionally used choropleth maps emphasize the map visualization and put less emphasis on the visualization of the statistical data. According to Porta and Last (2018), a choropleth map is a “map in which regions with differing occurrence rates of conditions of interest (e.g., cancer) are visually distinguished by different color or shading corresponding to rates at which the designated conditions have occurred in each region during the period of observation.”

Symanzik and Carr (2008) pointed out three main advantages of linked micromap plots over choropleth maps. First, small map regions (such as Washington, D.C., in a map of the United States) may be hard to see in a choropleth map. In linked micromap plots, small map regions are often enlarged and sometimes also pulled to the outside of the main map area, resulting in some kind of a map caricature that helps to make the smaller map regions better visible. Further changes of the underlying maps often take place, such as moving far-away subregions closer to the main geographic region (such as Alaska and Hawaii for the United States) and simplifying the boundaries of the subregions (think of river boundaries or rugged coastlines) for faster plotting and obtaining a less dominant, i.e., thinner, borderline.

Second, when converting data from an ordered statistical variable into five to eight discrete colors for coloring the map area in a choropleth map, there is an immediate loss of information. The exact numerical values of the variable are lost, the ranking of the values is

lost, and the relative proximities of the values that are translated into a certain color class that represents an interval of values is lost. This problem is solved by using row-labeled statistical displays such as dotplots that are linked to the micromaps in linked micromap plots, rather than showing the statistical data directly on the map via a small set of discrete colors.

Third, it is difficult to show more than one statistical variable directly on a map. This becomes even more difficult when trying to display confidence bounds or the minimum, first quartile, median, third quartile, and maximum that make up the summary statistics that are visually displayed in a boxplot. This problem is resolved in linked micromap plots by linking the statistical graphic displays to certain spatial areas, rather than displaying the statistical information directly on a map. Also, multiple types of statistical graphic displays can be shown simultaneously in a linked micromap plot.

Readers who are interested to learn more about the background and history of linked micromap plots are encouraged to read the book chapters and encyclopedia entries by Symanzik and Carr (2008) and Symanzik, Carr, McManus, et al. (2017). Even more, for readers who want to learn the full background behind all three main types of micromap visualizations, we strongly recommend to read the excellent book by Carr and Pickle (2010).

1.2.2 Design, Features, and Interpretation

We will use three mock-up linked micromap plots, shown in Figures 1.1, 1.2, and 1.3 to introduce the main nomenclature for linked micromap plots and how to interpret such plots. These three figures are based on actual boundaries for New Jersey in the United States, obtained from United States Census Bureau (2013). The statistical data are entirely made up for demonstration purposes. In particular, the values for `StatVar4`, one of the four statistical variables, originate from a Normal distribution with mean 55 and standard deviation 10 and have been randomly assigned to the 21 counties in New Jersey.

In general, linked micromap plots consist of four to seven columns of panels. One to four of these columns are statistical graphics columns that can be used to display various statistical displays such as dotplots (with and without confidence bounds), boxplots, scatterplots, time series, and other graphical displays of observed or estimated statistical data. Sometimes, an additional global summary statistic is also shown in these columns. The three remaining columns are used for the maps, a color-coded legend, and a column with the subregion names, abbreviations, or some other meaningful subregion identifiers. The color-coded legend links the subregion names, the map regions, and the statistical data. There exists no fixed order how the columns have to be arranged. In particular, there is no strong recommendation where the column with the maps should be placed. This often depends on personal preferences of the authors and plot designers. There exist examples in the literature where the maps were placed on the left, e.g., in Mast (2013b) and Das Gupta and Wong (2021), on the right, e.g., in Wartenberg (2009), and even in the middle, e.g., in Tatalovich and Stinchcomb (2019).

Each row in a linked micromap plot is representing one subregion. The rows with the subregions are sorted according to a sorting variable that may or may not be one of the variables that are displayed in the statistical graphics columns. Moreover, the rows are arranged in perceptual groups. Ideally, each perceptual group should contain the same number of subregions, but that is not always possible. Table 1.1 provides suggestions for common partitionings of the number of subregions in each perceptual group, depending on the total number of subregions in the area of interest. In practice, having close to five subregions in most of the perceptual groups is ideal from a cognitive perspective. However, sometimes it

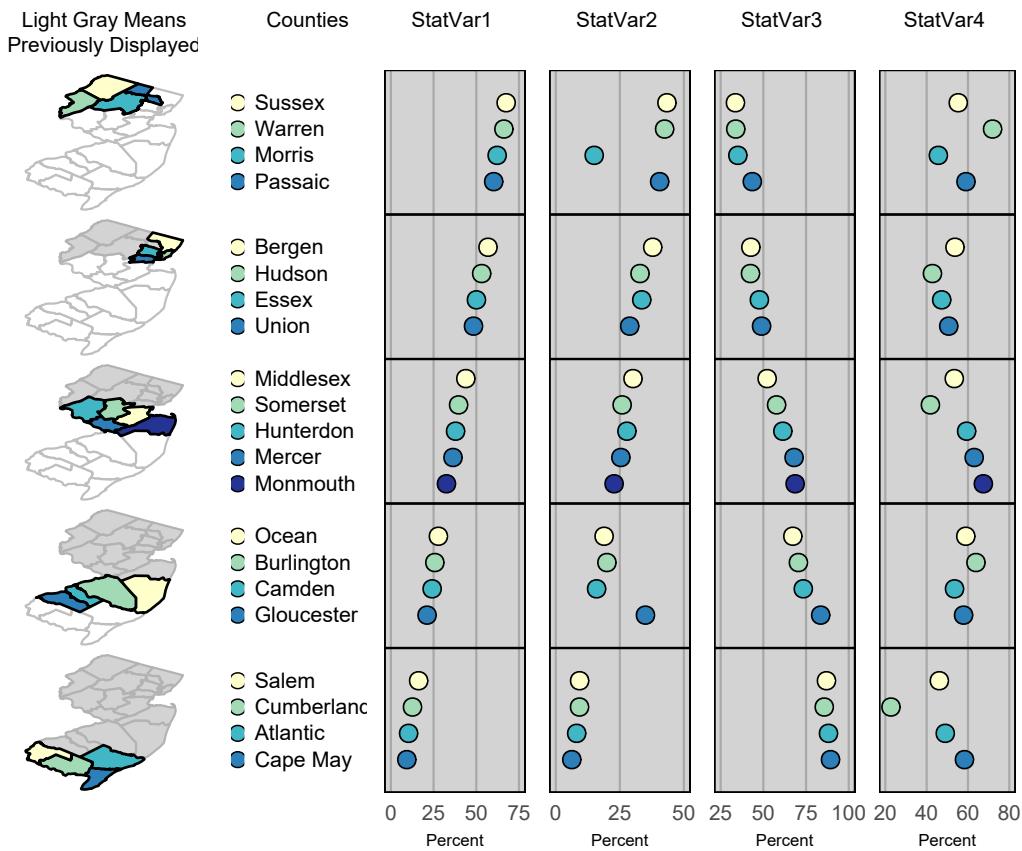


FIGURE 1.1 Mock-up linked micromap plot, based on the 21 counties of New Jersey. StatVar1 is used as the sorting variable.

is desirable to highlight one subregion in a median row and emphasize which subregions fall above and which fall below with respect to the sorting variable. Those partitionings that contain a 1 in Table 1.1 should be interpreted as partitionings with a median row.

Within each perceptual group, the subregions shown in the map, the color-coded legend next to the subregion names, and the graphics in the statistical graphics columns are linked by the color from the color-coded legend. This allows to quickly link information from the statistical graphics columns to the underlying subregions in the maps and to the subregion names via the color-coded legend. Within each perceptual group, a particular color, shown in the color-coded legend, represents a single subregion in the map column and all statistical data for this subregion in the graphs in the statistical graphics columns. The same set of colors is used in all perceptual groups. It might be tempting to assume that subregions that appear in the same color in different perceptual groups are somehow related, but this is not the case. Rather, the colors could be used as an ordering criterion within each perceptual group. When using a sequential color scheme that goes from light to dark as in Figure 1.1, the lightest color (here yellow) always represents the largest value according to the variable that is used as the sorting variable in each perceptual group, the second lightest color (here green) always represents the second largest value in each perceptual group, and so on.

TABLE 1.1 Full symmetry partitionings with targeting groups of size 5. The left column (Number) contains the number of regions. The middle column (Partitioning 1) puts smallest counts in the middle. Full symmetry alternatives that avoid small counts appear in the right column (Partitioning 2). Abandoning full symmetry can lead to fewer panels. The table ends with 51 regions (the number of U.S. states plus Washington, D.C.), but it can be easily extended. Table originally published in Symanzik and Carr (2008).

Number	Partitioning 1	Partitioning 2
1	1	
2	2	
3	3	
4	4	
5	5	
6	3 3	
7	3 1 3	2 3 2
8	4 4	
9	4 1 4	3 3 3
10	5 5	
11	5 1 5	3 5 3
12	5 2 5	4 4 4
13	5 3 5	4 5 4
14	5 4 5	
15	5 5 5	
16	5 3 3 5	4 4 4 4
17	5 3 1 3 5	3 4 3 4 3
18	5 4 4 5	4 5 5 4
19	5 4 1 4 5	4 4 3 4 4
20	5 5 5 5	
21	5 5 1 5 5	4 4 5 4 4
22	5 5 2 5 5	5 4 4 4 5
23	5 5 3 5 5	
24	5 5 4 5 5	
25	5 5 5 5 5	
26	5 5 3 3 5 5	5 4 4 4 4 5
27	5 5 3 1 3 5 5	4 4 4 3 4 4 4
28	5 5 4 4 5 5	4 5 5 5 5 4
29	5 5 4 1 4 5 5	4 4 4 5 4 4 4
30	5 5 5 5 5 5	
31	5 5 5 1 5 5 5	4 4 5 5 5 4 4
32	5 5 5 2 5 5 5	5 5 4 4 4 5 5
33	5 5 5 3 5 5 5	4 5 5 5 5 5 4
34	5 5 5 4 5 5 5	
35	5 5 5 5 5 5 5	
36	5 5 5 3 3 5 5 5	4 4 5 5 5 5 4 4
37	5 5 5 3 1 3 5 5 5	4 4 4 4 5 4 4 4 4
38	5 5 5 4 4 5 5 5	4 5 5 5 5 5 5 4
39	5 5 5 4 1 4 5 5 5	4 4 4 5 5 5 4 4 4
40	5 5 5 5 5 5 5 5	
41	5 5 5 5 1 5 5 5 5	4 4 5 5 5 5 5 4 4
42	5 5 5 5 2 5 5 5 5	5 5 5 4 4 4 5 5 5
43	5 5 5 5 3 5 5 5 5	4 5 5 5 5 5 5 5 4
44	5 5 5 5 4 5 5 5 5	
45	5 5 5 5 5 5 5 5 5	
46	5 5 5 5 3 3 5 5 5 5	4 4 5 5 5 5 5 4 4
47	5 5 5 5 3 1 3 5 5 5 5	4 4 4 4 5 5 5 4 4 4 4
48	5 5 5 5 4 4 5 5 5 5	4 5 5 5 5 5 5 5 5 4
49	5 5 5 5 4 1 4 5 5 5 5	4 4 4 5 5 5 5 5 4 4 4
50	5 5 5 5 5 5 5 5 5 5	
51	5 5 5 5 5 1 5 5 5 5 5	4 4 5 5 5 5 5 5 5 4 4

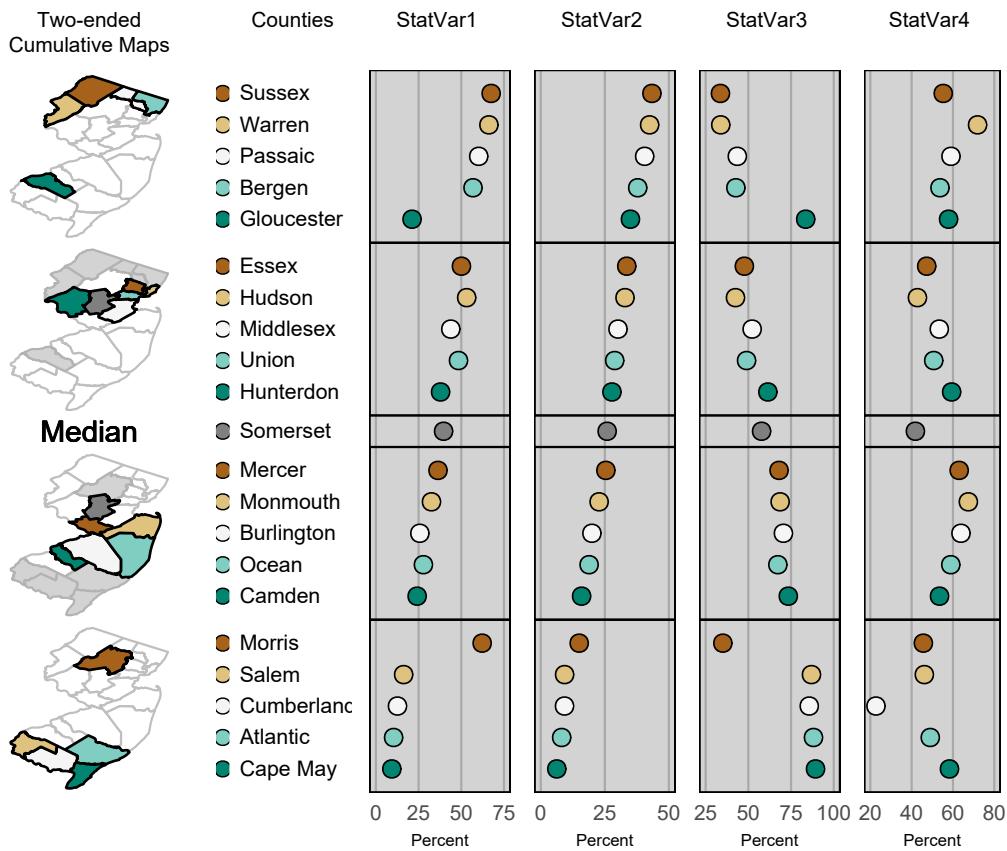


FIGURE 1.2 Mock-up linked micromap plot, based on the 21 counties of New Jersey. StatVar2 is used as the sorting variable. Further changes are the use of two-ended cumulative maps, the reduced number of perceptual groups, and the introduction of a median row.

Figures 1.1 and 1.2 each show a seven-column linked micromap plot for the 21 counties of New Jersey. The first column shows the maps, the second column shows the color-coded legend, and the third column shows the subregion identifiers, here the county names in New Jersey. Columns four, five, six, and seven are the statistical graphics columns that show dotplots that are by far the most frequently used graph types in linked micromap plots.

In Figure 1.1, the rows are sorted from highest (at the top) to lowest (at the bottom) according to StatVar1. This layout makes use of Partitioning 2 from Table 1.1 and uses five perceptual groups and no median row. Each consecutive map shows in light gray which subregions have been colored previously in the map(s) above, resulting in some aggregated maps. In the map at the bottom, eventually all subregions have been colored. With this sorting, the maps show some strong spatial pattern. The subregions with the largest values of StatVar1 can be found (by construction) in the northern part of New Jersey while the subregions with the smallest values of StatVar1 can be found (by construction) in the southern part of New Jersey.

In Figure 1.2, the rows are sorted from highest (at the top) to lowest (at the bottom) according to StatVar2. This layout makes use of Partitioning 1 from Table 1.1 and uses four

perceptual groups and a median row. A median row is often a good solution if the number of subregions is odd such as for the 21 counties of New Jersey. Here, Somerset is the county that appears (by construction) in the median row. This implies that it has the 11th highest and 11th lowest `StatVar2` value. Somerset does not appear in a map by itself, but rather is added to the maps in the panels above and below the median row in a neutral (gray) color, thus increasing the number of subregions shown in each of these two maps by one (i.e., six here). This plot makes use of a two-ended aggregate coloring of the maps. The subregions from all previous perceptual groups again are filled in the subsequent perceptual groups. But, this filling proceeds from the top perceptual group to the median row only and also from the bottom perceptual group to the median row by sequentially filling the subregions that have already been displayed on the more extreme ends. This allows to better distinguish between the top-50% and bottom-50% of the data according to the sorting variable. Here, it becomes obvious that Morris (missing in the northern part of New Jersey) and Gloucester (missing in the southern part of New Jersey) are potential spatial outliers with respect to `StatVar2`. Morris has a much lower value of `StatVar2` than its geographic neighbors and Gloucester has a much higher value of `StatVar2` than its geographic neighbors.

Finally, we want to describe and interpret the patterns from the statistical graphics columns. In Figures 1.1 and 1.2, the dots for `StatVar1` and `StatVar2` run almost in parallel, i.e., high values of `StatVar1` are associated with high values of `StatVar2` and low values of `StatVar1` are associated with low values of `StatVar2`. There are only a few exceptions – the previously mentioned Morris and Gloucester that do not fit in the overall pattern. Overall, such a pattern is an indicator of a strong positive (linear) association between these two variables. In contrast, the dots for `StatVar1` and `StatVar3` (and `StatVar2` and `StatVar3`) diverge, forming some crude caret shape (resembling an upside down V-shape), i.e., high values of `StatVar1` (and `StatVar2`) are associated with low values of `StatVar3` and low values of `StatVar1` (and `StatVar2`) are associated with high values of `StatVar3`. Overall, such a pattern is an indicator of a strong negative (linear) association between these two variables. This can be confirmed numerically. The correlation coefficients r are about 0.8 for `StatVar1` and `StatVar2`, about -0.99 for `StatVar1` and `StatVar3`, and about -0.75 for `StatVar2` and `StatVar3`.

In Figure 1.3, the rows are sorted from highest (at the top) to lowest (at the bottom) according to `StatVar4`. This layout also makes use of Partitioning 1 from Table 1.1 and uses four perceptual groups and a median row. Moreover, it places the maps on the right. There is no strong noticeable pattern when comparing `StatVar1`, `StatVar2`, and `StatVar3` with `StatVar4`. While this can also be seen in the previously created Figures 1.1 and 1.2, this becomes most obvious in Figure 1.3. In fact, the correlation coefficients r are about 0.2 for `StatVar1` and `StatVar4`, about 0.36 for `StatVar2` and `StatVar4`, and about -0.15 for `StatVar3` and `StatVar4`. The reader should not be tempted to overinterpret the moderate correlation coefficient r of about 0.36 for `StatVar2` and `StatVar4` as this is entirely due to chance: The values for `StatVar4` originate from a Normal distribution with mean 55 and standard deviation 10 and have been randomly assigned to the 21 counties in New Jersey. Therefore, the band of five connected subregions in the top perceptual group in Figure 1.3 is entirely due to chance. It is easy to be misguided by an apparent pattern in a statistical graphic or a map visualization. Some considerable work has been done over the past 15 years, trying to answer the question “Is what we see really there?” (Wickham et al. (2010)). Readers are encouraged to assess their visual capabilities to detect true patterns using the examples from Wickham et al. (2010) and Beecham et al. (2017) where line-ups of one data-based graphic (or map) have been compared with several graphics (or maps) that have been created under the null hypothesis of no pattern (or no spatial dependence).

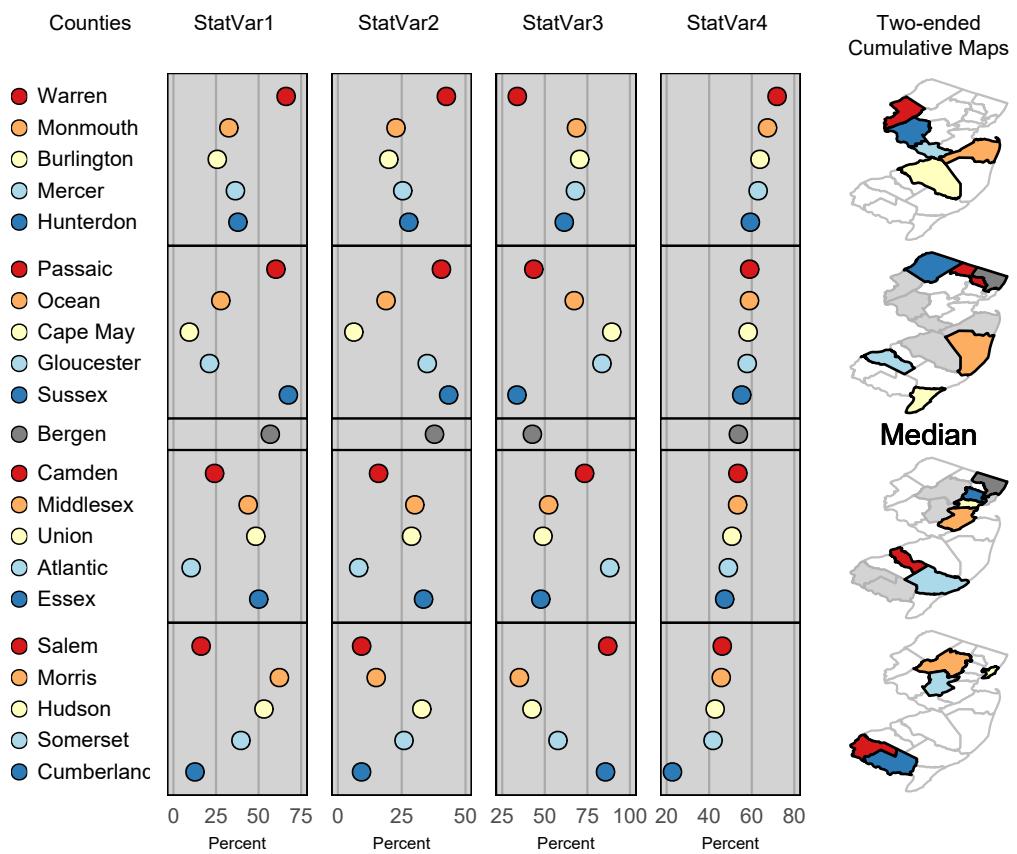


FIGURE 1.3 Mock-up linked micromap plot, based on the 21 counties of New Jersey. StatVar4 is used as the sorting variable. The maps are placed on the right.

For readers who cannot immediately interpret the patterns from parallel dotplots, it is sometimes helpful to visualize (or mentally visualize) these patterns via scatterplots or scatterplot matrices as shown in Figure 1.4. Doing such a visual translation from parallel dotplots to scatterplots or scatterplot matrices a few times should help to correctly interpret the patterns from parallel dotplots in the future.

1.2.3 Software and Online Developments

As mentioned in Section 1.2.1, linked micromap plots were first presented at the JSM in Chicago, Illinois, in 1996 (Olsen et al., 1996). Lead contributors were Daniel (Dan) B. Carr from George Mason University and Anthony R. Olsen and Susanne M. Pierson from the U.S. Environmental Protection Agency in Corvallis, Oregon. During the first five years of their existence, almost all linked micromap plots were created in S-Plus. The basis for these was a set of panel function that eventually were translated into R functions and later formed the basis for the R-based linked micromap plots in Carr and Pickle (2010). These panel functions are further used and described in Chapter 12 for non-traditional linked micromap plots. Other experimental approaches to construct linked micromap plots in R have been described in Symanzik and Carr (2013).

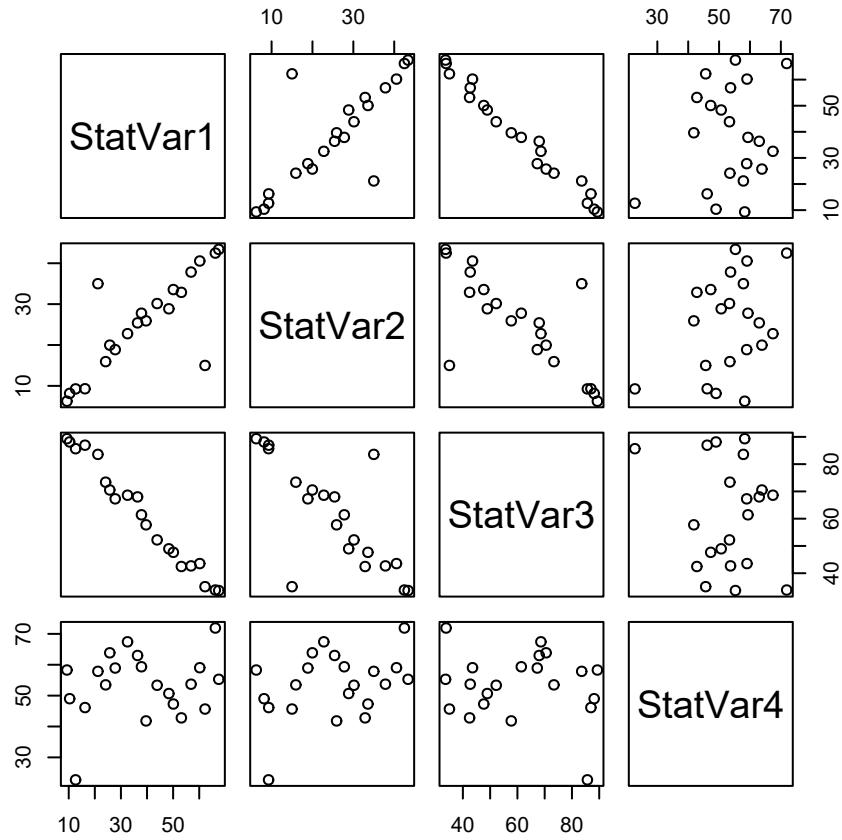


FIGURE 1.4 Mock-up scatterplot of the four statistical variables shown in the three previous linked micromap plots, based on the 21 counties of New Jersey.

Linda W. Pickle and B. Sue Bell from the National Cancer Institute, Jim X. Chen from George Mason University, and a few others soon developed an interest in the use of interactive linked micromap plots for cancer data (Bell et al., 2006; Carr et al., 2003; Carr, Chen, et al., 2002; Chen et al., 2006; Wang et al., 2002). This resulted in the creation of stand-alone Java code and the Java-based National Cancer Institute's State Cancer Profiles web site in 2002. This web site allowed the user to easily compare graphs of national or state trends for Whites, Blacks, Hispanics, Asian Pacific Islanders, and American Indian Alaskan Natives for both genders for a variety of cancer types. While the State Cancer Profiles web site still exists, the interactive linked micromap plots component was retired after about 12 years of use around 2014/2015.

Jürgen Symanzik (first at George Mason University and later at Utah State University) and his colleagues and students experimented with a variety of other software approaches to creating linked micromap plots. These included linked micromap plots based on the Graphics Production Library (GPL) (Carr et al., 1996; L. Wilkinson et al., 2000), developed at the Bureau of Labor Statistics, for the USEPA's Cumulative Exposure Project (CEP) (Symanzik, Axelrad, et al., 1999; Symanzik, Carr, Axelrad, et al., 1999; Symanzik et al., 2000). The

GPL, extended and renamed to nViZn (read envision) (L. Wilkinson et al., 2001) and distributed by Illumitek, Inc./SPSS, was also used in Jones and Symanzik (2001), Symanzik and Jones (2001), Symanzik et al. (2002), and Hurst et al. (2003) for the construction of interactive linked micromap plots.

Add historical and online developments. Also introduce both main R packages that are used throughout the book. Summarize their main features, differences, and limitations.

1.2.4 Challenges and Open Research Questions

One of the main challenges regarding linked micromap plots relates to geographic regions with many subareas. The challenge is that more than twelve perceptual groups with five subregions each typically do not fit on a single print page. Possible solutions have been discussed for U.S. states with many counties such as Iowa and Tennessee that have between 60 and 120 counties (Carr, 2001). However, those were specially tailored solutions for these two states. Neither of the two current R packages for linked micromap plots supports similar solutions. Both current R packages only allow to stitch together multiple linked micromap plots, e.g., for the 255 counties of Texas (Payton et al., 2015) or for the 55 sub-boroughs of New York City (Medri et al., 2019; Medri Cobos, 2021), which are less sophisticated than the designs in Carr (2001).

Other conceptual challenges for linked micromap plots are imposed by the underlying geographic regions. No good solutions have been developed so far for long and narrow countries such as Chile where many of the provinces are so narrow that they can be barely colored without using a considerable distortion of the map. Problems exist if the geographic region consists of many small polygons, such as the numerous islands of the Philippines and Indonesia. Problems also exist when there are numerous big and numerous small subregions that cannot be resized in any obvious way as is the case for the 89 federal subjects of Russia where the largest subregions are about three orders of magnitude bigger than the smallest ones.

1.3 Conditioned Micromaps

1.3.1 Historical Development

1.3.2 Design, Features, and Interpretation

1.3.3 Software and Online Developments

Baulier (August 18, 2011) created conditioned micromaps in JMP.

See this ccmap macro from Friendly for CCmaps in SAS: <http://euclid.psych.yorku.ca/datavis/sasmac/ccmap.html>

1.3.4 Challenges and Open Research Questions

1.4 Comparative Micromaps

1.4.1 Historical Development

1.4.2 Design, Features, and Interpretation

1.4.3 Software and Online Developments

Baulier ([August 18, 2011](#)) created comparative micromaps in JMP.

1.4.4 Challenges and Open Research Questions

1.5 Outlook on the Book Chapters

1.6 How to Use this Book and the Accompanying R Code, Web-page, and R Package

1.7 INSTRUCTIONS FOR CHAPTER AUTHORS: Use of Bookdown

Please look at the source code for this chapter (i.e., the file `01-introduction.Rmd`) carefully. It provides an overview how to label and reference other chapters, sections, figures, and tables in your chapter. It also outlines how and what to index and how to include citations in your chapter.

Eventually, this chapter will become the real introduction for our *Micromap Plots in R* book. Minimal templates for actual chapters can be found in the files `02-micromap.Rmd`, `03-micromapST.Rmd`, etc.

The ultimate summary for this chapter will be based on the following text:

This chapter will provide a brief overview of the history of micromap plots, main application areas, and existing software for the creation of micromaps. A summary of the following eleven chapters of this book will also be provided.

1.8 Use of Bookdown

This section contains some basic information related to bookdown. For further details, see <https://bookdown.org/> and specifically <https://bookdown.org/yihui/bookdown/>.

For an overview how to use bookdown for CRC Press / Taylor & Francis books, see <https://yihui.org/en/2018/08/bookdown-crc/> and <https://www.routledge.com/bookdown-Authoring-Books-and-Technical-Documents-with-R-Markdown/Xie/p/book/9781138700109>.

1.9 Creating Figures and Tables

Here are some basic examples how to create figures and tables in bookdown. We have a figure in Figure 1.5 that makes use of the *cars* dataset and also a table in Table 1.2 that makes use of the *iris* dataset. See these examples how to create automatic figure and table numbers in your R code chunks and how to reference them in the main text. Also, please cite all datasets that are used in your chapter.

Use meaningful identifiers that start with the letters **Ch**, followed by the number of your chapter and a dash (such as Ch1-, Ch2-, etc.) so that we can eventually cross-reference figures across chapters (and also avoid that the same identifier is used more than once in different chapters).

```
par(mar = c(4, 4, 1, 0.1))
plot(cars, pch = 19)
```

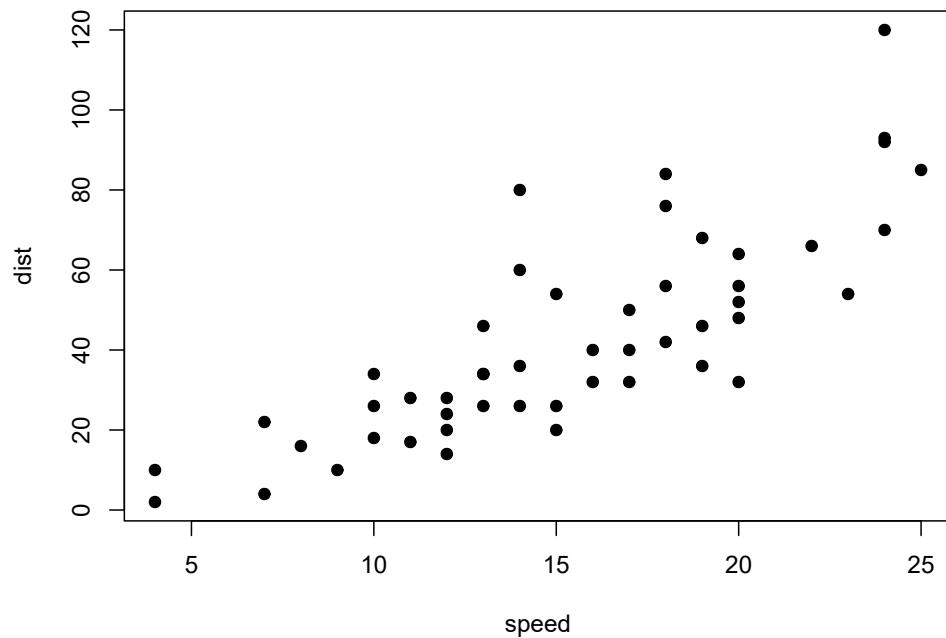


FIGURE 1.5 A trivial scatterplot of the cars dataset.

```
knitr::kable(
  head(iris),
  caption = "A table of the iris data.",
  booktabs = TRUE
)
```

TABLE 1.2 A table of the iris data.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

1.10 Indexing

As already done in the previous sections, index all R packages such as **ggplot2** and **ggmap** R packages.

Also provide an index entry for all datasets, such as the *cars* and the *iris* datasets.

For R packages and datasets, use the actual R spelling. Do not change the capitalization.

One final word on indexing: Please capitalize the first word of the index entry in a sequence of words, e.g., perceptual group, color blindness,, and quantile-quantile plot.

In case you want to use abbreviations, please introduce them first, e.g., linked micromap plots (LMplots) and conditioned choropleth maps (CCmaps).

Introduce a cross-reference index entry for the abbreviation (see the examples above), but always list the full length-index entry for the index.

See <https://en.wikibooks.org/wiki/LaTeX/Indexing> for other indexing options.

1.11 Citations and References

Here are some examples for citations: Xie (2022b) and Xie (2022a) are references from the preface. These citations appear as nouns in the text.

These are some micromap articles, book chapters, and books (Carr, 2001; Carr & Pickle, 2010; Symanzik & Carr, 2008). These are references for the **micromap** (Payton & Olsen, 2015) and **micromapST** (Carr & Pearson Jr., 2015) R packages. All of these citations appear in parentheses. **Note the use of the semicolon in the first set of articles, book chapters, and books.** Also note that three different bib files have been used here to create the final bibliography.

See here for further details on citations in bookdown: <https://bookdown.org/yihui/bookdown/citations.html>.

I have provided an updated bibtex file with a large number of micromap-related references, called **referencesMicromaps.bib** - see <https://github.com/symanzik/MicromapPlotsInR/blob/master/referencesMicromaps.bib>. See Appendix

A.1 for the references that are already listed in the bibtex file. Most of the underlying articles, book chapters, posters, etc. have been made available via a Box folder now. You should have received an e-mail invitation to this Box folder on 3/25/2022. If you did not receive such an invitation or cannot access the files, please let me know.

1.12 Micromap Examples

While Section 1.9 discussed general figure and table creation, this section focuses on micromaps.

The following examples have been taken from the `lmpplot()` help page of the **micromap** R package. Figure 1.6 shows the first basic linked micromap plot that makes use of the `USstates` and `edPov` datasets.

Overall, write and format your R code according to the tidyverse R style, summarized at <https://style.tidyverse.org/index.html>. As many of our function calls for micromaps require a large number arguments, see Section 2.5 called ‘Long Lines’ (<https://style.tidyverse.org/syntax.html#long-lines>) how to format such function calls.

```
library(micromap)

# initial example

data(USstates)
head(USstates@data)

##   ST    ST_NAME AREA_KM PERIM_KM
## 0 AK    Alaska 1506038   60261
## 1 AL  Alabama 133761    2355
## 2 AR  Arkansas 137734    2172
## 3 AZ  Arizona 295267    2395
## 4 CA California 409603   5682
## 5 CO Colorado 269600    2100

statePolys <- create_map_table(USstates, "ST")
head(statePolys)

##   ID region poly coordsx coordsy hole plotorder plug
## 1 AK      1    1      2      5    0      1      0
## 2 AK      1    1      7     10    0      1      0
## 3 AK      1    1      4     12    0      1      0
## 4 AK      1    1      7     15    0      1      0
## 5 AK      1    1      4     15    0      1      0
## 6 AK      1    1      4     17    0      1      0
```

```
data(edPov)

# basic figure 1
lmplot(
  stat.data = edPov,
  map.data = statePolys,
  panel.types = c("labels", "dot", "dot", "map"),
  panel.data = list("state", "pov", "ed", NA),
  ord.by = "pov",
  grouping = 5,
  median.row = TRUE,
  plot.width = 2,
  plot.height = 6,
  map.link = c("StateAb", "ID")
)
```

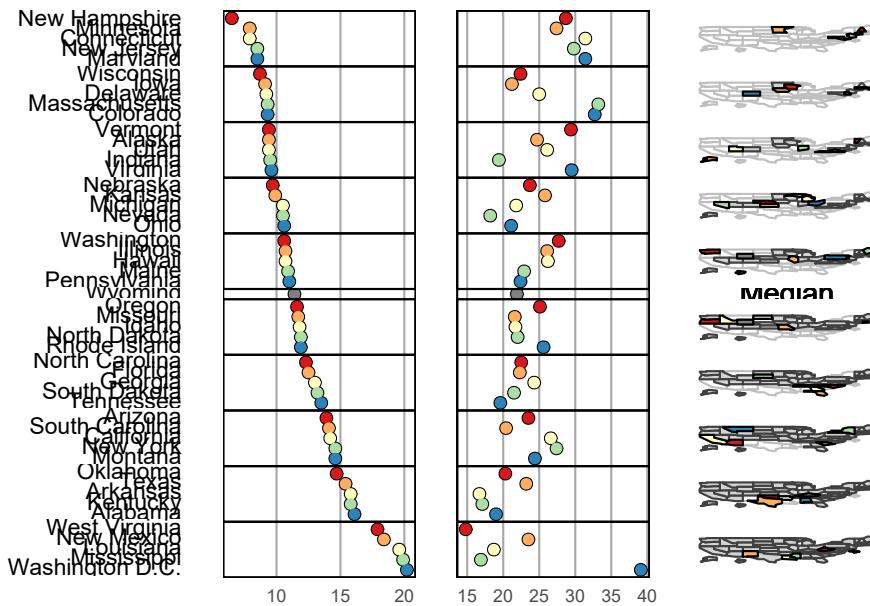


FIGURE 1.6 Here is a first micromap example. Note that it occupies 90 percent of the page width. The rows are far too much condensed.

This gets further refined now. Figure 1.7 shows the resulting second micromap plot.

```
# publication figure 1a
lmplot(
  stat.data = edPov,
  map.data = statePolys,
  panel.types = c("labels", "dot", "dot", "map"),
  panel.data = list("state", "pov", "ed", NA),
```

```

ord.by = "pov",
grouping = 5,
median.row = TRUE,
map.link = c("StateAb", "ID"),
plot.height = 9,
colors = c("red", "orange", "green", "blue", "purple"),
map.color2 = "lightgray",
panel.att = list(
  list(1,
    header = "States",
    panel.width = 0.8,
    align = "left",
    text.size = 0.9
  ),
  list(2,
    header = "Percent Living Below\nPoverty Level",
    graph.bgcolor = "lightgray",
    point.size = 1.5,
    xaxis.ticks = list(10, 15, 20),
    xaxis.labels = list(10, 15, 20),
    xaxis.title = "Percent"
  ),
  list(3,
    header = "Percent Adults With\n4+ Years of College",
    graph.bgcolor = "lightgray",
    point.size = 1.5,
    xaxis.ticks = list(10, 20, 30, 40),
    xaxis.labels = list(10, 20, 30, 40),
    xaxis.title = "Percent"
  ),
  list(4,
    header = "Light Gray Means\nHighlighted Above",
    inactive.border.color = gray(0.7),
    inactive.border.size = 2,
    panel.width = 0.8
  )
)
)

```

```
Some more refinements, resulting in Figure 1.8.  
  
edPov$points <- 0  
  
# publication figure 1b  
lmpplot(  
  stat.data = edPov,  
  map.data = statePolys,  
  panel.types = c("dot", "labels", "dot", "dot", "map"),  
  panel.data = list("points", "state", "pov", "ed", NA),
```

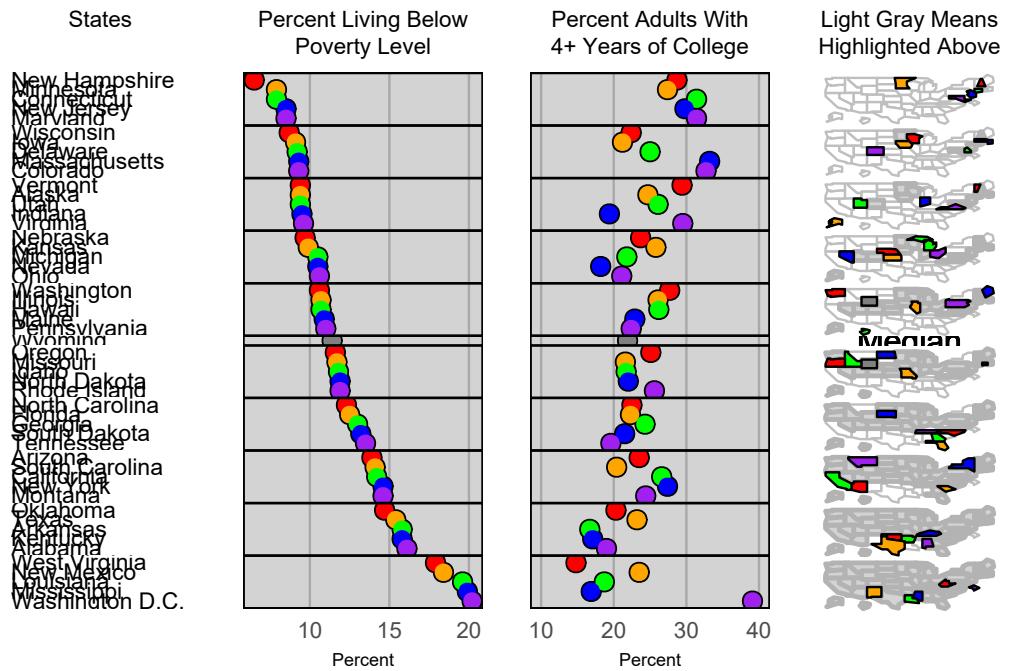


FIGURE 1.7 And here is a second micromap example. This uses the default output settings. This still looks very bad.

```
map.link = c("StateAb", "ID"),
ord.by = "pov",
grouping = 5,
median.row = TRUE,
plot.height = 9,
colors = c("red", "orange", "green", "blue", "purple"),
map.color2 = "lightgray",
panel.att = list(
  list(1,
    panel.width = 0.15,
    point.type = 20,
    graph.border.color = "white",
    xaxis.text.display = FALSE,
    xaxis.line.display = FALSE,
    graph.grid.major = FALSE
  ),
  list(2,
    header = "States",
    panel.width = 0.8,
    align = "left",
    text.size = 0.9
  )
),
```

```

list(3,
  header = "Percent Living Below\nPoverty Level",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  xaxis.ticks = list(10, 15, 20),
  xaxis.labels = list(10, 15, 20),
  xaxis.title = "Percent"
),
list(4,
  header = "Percent Adults With\n4+ Years of College",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  xaxis.ticks = list(20, 30, 40),
  xaxis.labels = list(20, 30, 40),
  xaxis.title = "Percent"
),
list(5,
  header = "Light Gray Means\nHighlighted Above",
  inactive.border.color = gray(0.7),
  inactive.border.size = 2,
  panel.width = 0.8
)
)
)

```

1.13 Alternative Creation and Inclusion of Figures

Final refinements. Here, the code is run separately. The figure is not shown directly. Rather, an external figure (jpeg or pdf) is created. Eventually, in Figure 1.9, this externally created figure is included into the text.

Even though this works, I would suggest to always use the approach from the previous section, i.e., Section 1.12, whenever possible. Exceptions are externally created files or screenshots, e.g., from a Shiny app, that could be included this way. Note that the external figures should be placed in the `img` folder and should be jpeg format.

```
# publication figure 1c
lmpplot(
  stat.data = edPov,
  map.data = statePolys,
  panel.types = c("map", "dot", "labels", "dot", "dot"),
  panel.data = list(NA, "points", "state", "pov", "ed"),
  map.link = c("StateAb", "ID"),
  ord.by = "pov",
  grouping = 5,
  median.row = TRUE,
```

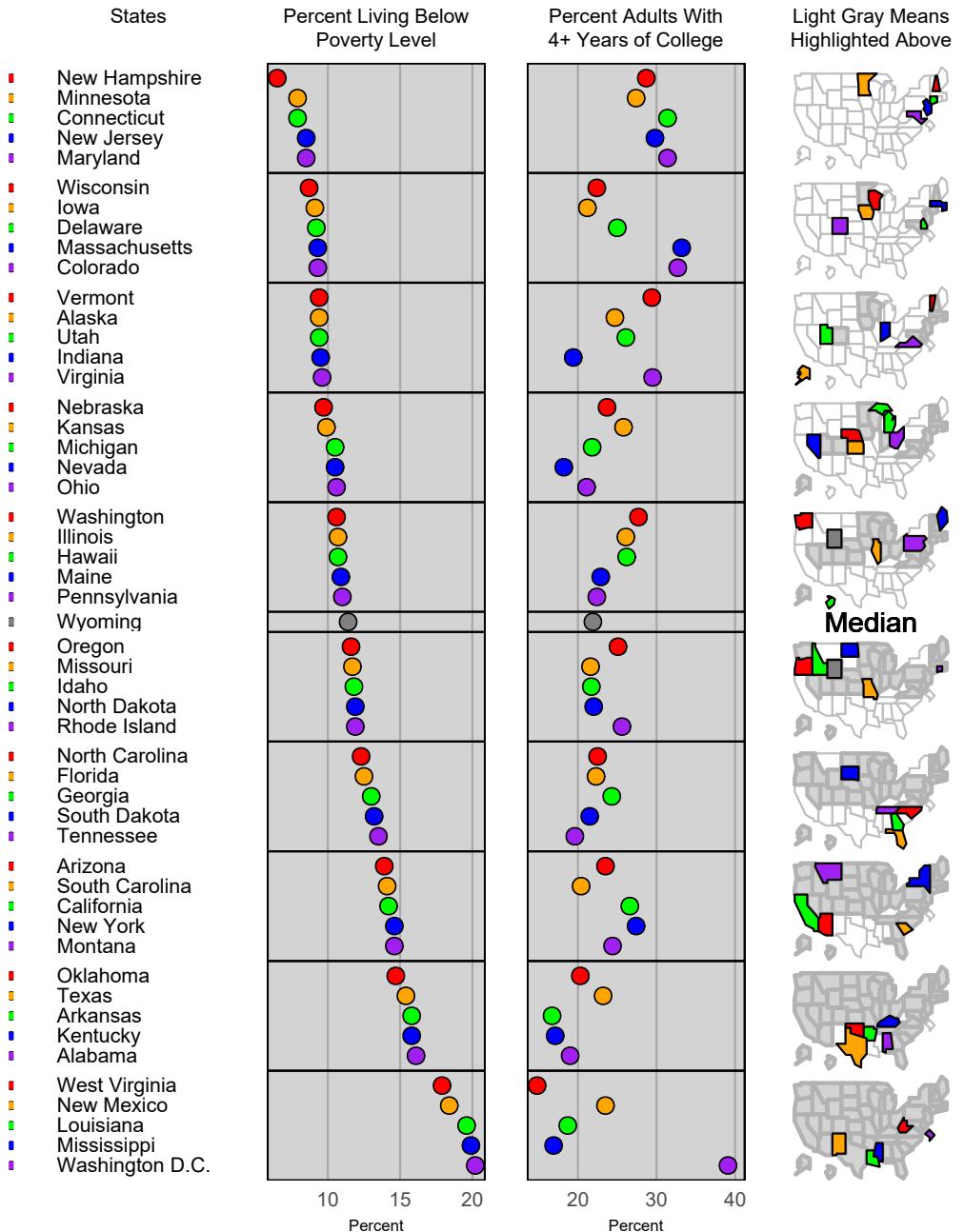
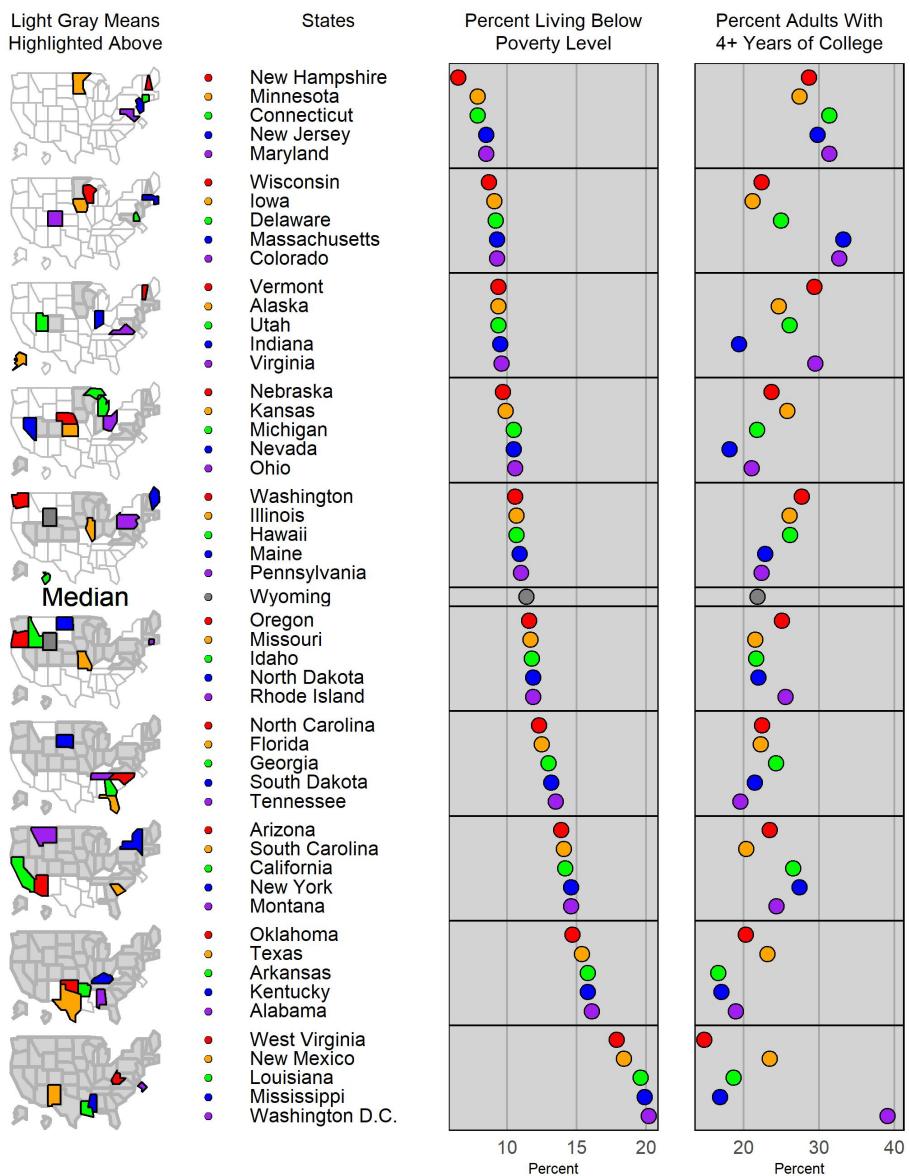


FIGURE 1.8 And now the third (revised) micromap example. Note that this specifies a width and height for the figure. This may be the best solution to scale the micromap plots.

```
plot.height = 9,
colors = c("red", "orange", "green", "blue", "purple"),
map.color2 = "lightgray",
print.file = "img/Ch1-micromap4-external.jpeg",
panel.att = list(
  list(2,
    panel.width = 0.15,
    point.type = 20,
    graph.border.color = "white",
    xaxis.text.display = FALSE,
    xaxis.line.display = FALSE,
    graph.grid.major = FALSE
  ),
  list(3,
    header = "States",
    panel.width = 0.8,
    align = "left",
    text.size = 0.9
  ),
  list(4,
    header = "Percent Living Below\\nPoverty Level",
    graph.bgcolor = "lightgray",
    point.size = 1.5,
    xaxis.ticks = list(10, 15, 20),
    xaxis.labels = list(10, 15, 20),
    xaxis.title = "Percent"
  ),
  list(5,
    header = "Percent Adults With\\n4+ Years of College",
    graph.bgcolor = "lightgray",
    point.size = 1.5,
    xaxis.ticks = list(20, 30, 40),
    xaxis.labels = list(20, 30, 40),
    xaxis.title = "Percent"
  ),
  list(1,
    header = "Light Gray Means\\nHighlighted Above",
    inactive.border.color = gray(0.7),
    inactive.border.size = 2,
    panel.width = 0.8
  )
)
)
```

**FIGURE 1.9** And now the fourth (and final) micromap example.

1.14 Translating the Book Chapters to pdf and html Output

The following instructions assume that you will be working locally on your own computer on your own chapter only. If you are familiar with git and github (and how to use these from RStudio), adjust the workflow accordingly to directly collaborate and interact with other chapter authors who will be using git and github. The use of git and github is optional. If you do not want to use these, you can always provide me with your Rmd and other source files.

Our book lives on <https://github.com/symanzik/MicromapPlotsInR>. This is a public directory. Everyone will be able to see the advancement of our book and provide feedback. I plan to provide stable updates on a regular basis. If you are familiar with git and github, simply pull new versions from the repository.

If you do not want to use git and github at this time, please do the following: Select **Code** -> **Download ZIP** on the github page above. Unzip the ZIP file where you want to host these files locally on your computer.

Open the R project file `MicromapPlotsinR.Rproj` in RStudio by double clicking on it. Ideally, always work in the framework of this R project while working on your book chapter.

Open the Rmd file that serves as a template for your chapter, e.g., `02...Rmd`. Also open `index.Rmd` and `01-introduction.Rmd` to see how certain features (such as R code, references, index entries, etc.) can be produced in bookdown. To do so, simply click on the file names in the RStudio **Files** menu.

Bookdown assumes that all files with extension .Rmd in a directory belong to the same book project. The file `index.Rmd` always is the main document. All additional Rmd files are read in sequential alpha-numerical order. For this book, you can initially speed up the translation process if you move files `02...Rmd`, `03...Rmd`, etc. (except your own chapter) into a different directory so they do not get translated each time. However, this means that cross-references to different chapters will not work. So, when you are about to finalize your chapter, please translate with all currently provided Rmd files in place (i.e., copy these files back into your local directory). Keep all other files and directories in the same directory structure as provided by me! These are needed to produce the book, but it is unlikely that you have to edit them at all. You likely will run into errors when trying to translate to pdf or html if some of these files are missing.

Please do not modify the file `index.Rmd` on your side at all. If something needs to be changed, it likely needs to be changed for all chapters. So, please let me know about such necessary changes for the `index.Rmd` file. You should be fine if you just edit the Rmd file for your chapter, i.e., `02...Rmd`, `03...Rmd`, etc.

Note that the short chapter summary at the start of a chapter has been adapted from my original book proposal for CRC Press. Adjust as needed when your chapter is being written. Also adjust your names, e.g., insert missing middle initials, and change the author order as desired for your chapter.

To translate your chapter to html or pdf (and in fact, also translate all other Rmd files that are in this directory), select **Build** from the RStudio menu that appears next to **Environment** and **History** (often in the upper right corner of RStudio). Note that there exists a second **Build** in the top menu of RStudio (close to **Session** and **Debug**) – this is not what we need. You now should see the menu **Build Book**. Select `bookdown::pdf_book`

to create the pdf version of the book. Alternatively, select **bookdown::gitbook** to create the html version of the book.

If your translation to html or pdf does not work immediately, try the following:

- Check that you have all necessary R packages installed, in particular **bookdown**, **microMap**, **micromapST**, and all their dependencies. Note that the dependencies should be installed automatically with each package, but some may require manual installation. Chapter 2 also needs the **labeling** R package. If necessary, install missing R packages (and their dependencies) via Tools → Install Packages in RStudio.
- Do you have recent versions (from 2022) of RStudio and R itself? If not, update them. Updating R can be done most easiest via the **installr** R package and its **updateR()** function. One recommendation: Do not delete the R packages from your current installation of R until you are sure that they have been successfully transferred into the new installation of R.
- Are all R packages up to date? Check via Tools → Check for Package Updates in RStudio. Mismatching packages may be incompatible. If packages still do not match, enforce an update of all R packages via the following. This is also a good idea when you updated to a new version of R, but copied all your currently installed R packages to the new location:

```
update.packages(checkBuilt = TRUE, ask = FALSE)
```

At this time, you should be able to translate to html. Translation to pdf may still require a few additional steps:

- Do you have some LaTeX version installed? If not, I would suggest to install the full 6GB **texlive** version from <https://www.tug.org/texlive/>. Alternatives are **MiKTeX** (<https://miktex.org/>) or **TinyTeX** (<https://yihui.org/tinytex/>). After the installation, you should restart your computer.
- If you have a current installation of **MiKTeX** or **TinyTeX**, but still get errors when you try to translate to pdf, update your current installation (including the used LaTeX packages). Google for help how to do this for a certain installation of **MiKTeX** or **TinyTeX** for your operating system.
- If all of the above still do not work for you (but you can create the html version), there are two options:
 - (i) Just create the html version. But, this may require some more back and forth later on during the review process of your chapter.
 - (ii) Uninstall your current LaTeX installation, and install the full 6GB **texlive** version from <https://www.tug.org/texlive/>. This usually resolves all problems.

The html version of the book will be visually more appealing. The pdf version will be more complete. Note that some of the pdf features of the book do not appear in the html version, e.g., there are no chapter authors listed, no list of figures and no list of tables is produced, no index is created, there is a different appearance of the references, and others. As the final delivery to CRC Press are the source files for the pdf version, we do not have to focus on the html version at this time (but create the html version if you don't get the translation to pdf to work on your side). If you are aware how to modify the code (Rmd, yml, or other) to obtain a matching html version for some of the missing / different features, please let me

know. As far as I could see, some of these features do not even exist for the html version at this time, but of course, might be implemented while we are working on the book.

At this time, do not worry about any formatting issues, placement of figures, etc. in the pdf version. These will be addressed only once all chapter content has been finalized.

1.15 External Data Files and Additional R Code

Likely, several of the book chapters are based on external data files and/or additional R code. Eventually, these have to be made available to our readers in a meaningful way. This could be done via additional (new) R packages, e.g., `MPIRdata` or `micromapExtra`. The underlying R code for some of the chapters itself could result in new R packages on CRAN or github. There are numerous options. We do not have to decide at this time, but only later in the year once we know how much additional material (data and R code) there will be.

For now, let's just deposit any external data and additional R code that is not directly part of a chapter into the `data` directory at <https://github.com/symanzik/MicromapPlotsInR/data>. Please send your data and additional R files to me, best as a zip file, so I can upload them to github. Alternatively, the data can be pushed directly to the GitHub repository from a local version if you are familiar with Git and GitHub.

Overall, this will allow us to share data and additional R code among multiple chapters. Someone may want to use a modified shapefile, a new plot type, or create a different type of a Shiny app based on a micromap plot introduced elsewhere in the book.

Assuming that the data directory is at the same directory level as the Rmd file of a chapter, data can be imported into a chapter as follows:

```
data <- read.csv("data/file.csv")
load(file = "data/file.RData")
```

See Section 4.4 how I read in shapefiles and data for Chinese micromaps. Note that this chapter is still highly experimental at this time!!!

1.16 Open Bookdown Questions

I have to resolve a few open bookdown questions on my side:

- How to modify the appearance of R code chunks, e.g., reduced font size and reduced spacing? See <https://bookdown.org/yihui/rmarkdown-cookbook/chunk-styling.html> and <https://stackoverflow.com/questions/25646333/code-chunk-font-size-in-rmarkdown-with-knitr-and-latex> and <https://github.com/rstudio/rmarkdown/issues/388>. I need to look at some more examples and consult with CRC Press what they want for their books.
- How to add chapter authors to the header of each chapter and to the table of content? See <https://github.com/admindatahandbook/book/issues/4>, <https://tex.stackexchange.com/questions/156862/displaying-author-for-each-chapter-in-book>, and <https://stackoverflow.com/questions/41655383/r-markdown-similar-feature-to-newcommand-in-latex/41664105>. This works for the pdf version, but needs some fine-tuning and consultation with CRC Press.
- How to create chapter-specific bibliographies at the end of each chapter, rather than a single bibliography at the end of the book? See <https://stackoverflow.com/questions/45028623/is-there-a-way-to-add-chapter-bibliographies-using-bookdown>, <https://tex.stackexchange.com/questions/525778/multiple-bibliographies-using-natbib-and-chapterbib-bibtex-illegal-error>, <https://community.rstudio.com/t/one-bibliography-per-chapter-when-using-pandoc-for-citations/117105/2>, <https://github.com/rstudio/bookdown-chapterbib>, and <https://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib>. This basically works now, but needs some fine-tuning.

Note that `apa` and `authoryear` settings for `biblio-style` are not exactly the same as `apalike`, but they come close. For example, the address field is ignored for books and book chapters. Also, notes and URLs are not exactly placed where they should be. I need to check with CRC Press what is needed on their side.

Also, there may not be a need for a comprehensive bibliography at the end of the book. I need to check this with CRC Press as well.

1.17 Initial Chapter 2 Outline (Merge into Chapter 2 Introduction at the End - Mostly Done)

Note: Chapter 1 will cover elements of a linked micromaps, 4 panels of legend, label, statistic, micromaps and rows of perceptual groups? Also discussion of spatial patterns and interpretation of positive, negative (or no) association in statistical panels of dotplots. - DONE - see Section 1.2.2.

- Overview of Four Steps (**Marcus**: Maybe a good figure could be a conceptual diagram showing the whole process?)
 - Identifying and Geoprocessing of Spatial Boundary Data
 - Linking Spatial Boundary Data and Statistical Data
 - Creating a Draft Linked Micromap Plot
 - Refining the Linked Micromap Plot
 - DONE - see Section 2.2.

- Spatial Polygons in R and micromap Package Updates (**Juergen**: How much of this do we need here in Chapter 2? Would it make more sense to mention this in Chapter 4?)
 - Spatial objects and simple feature objects
 - Updates
 - BRIEFLY MENTIONED - more in Chapter 4.
- Identifying and Geoprocessing of Spatial Boundary Data (**Marcus**: I think this is a good topic to discuss, but may be tricky to balance specificity with brevity. I'm no expert on it, but I know there are a few different methods one could use. Maybe just hit on the why and how, and briefly present tradeoffs of the different methods? I think for MM, the goal is just to increase render time w/o sacrificing spatial specificity, so maybe the exact method is not important? – **Juergen**: In fact, let's keep it simple here and work with ready-to-use shapefiles that are already available in some R packages; refer to Chapter 4 how to bring in external shapefiles and modify them for use in micromaps)
 - Keep this brief as Chapter 4 might cover this in detail?
 - Example of generalizing polygons Figure (Use same figures as Chapter 4?)
 - POSTPONED - to be discussed in Chapter 4.
- Linking Spatial Boundary Data and Statistical Data
 - Explicit link to spatial object
 - Under the hood link to simple features object
 - ONLY AT THE TOP LEVEL - see Section 2.3.2.
- Creating a Draft Linked Micromap Plot
 - Use workshop NARS pH sample & population estimates Figure (**Juergen**: Use simpler edPov data in first example)
 - DONE - see Section 2.3.3.
- Refining the Linked Micromap Plot
 - Use workshop NARS pH sample & population estimates Figure (**Juergen**: Use simpler edPov data in first example)
 - DONE - see Section 2.3.4.
- **Juergen**: Introducing a second basic example for a different geographic region, e.g., state level or environmental, and use of a different built-in plot type (e.g., boxplot). Similar four steps as before, but more condensed.
 - DONE - see Section 2.4
- Challenges to Visualizing Data with Linked Micromap Plots (**Marcus**: Not sure this is the right place for it, but maybe it's worth providing a brief contrast with micromapST? Maybe include this in the overview at the top? I think it would be useful for the reader to understand succinctly how the two differ and why there are two packages, though I'm sure these themes will be present throughout the book. Perhaps the latter point is not that interesting, i.e., it's common to have "competing" software in an open source environment? – **Juergen**: Yes, we need some discussions at the end of Chapters 2 and 3 that outline advantages and limitations of each of the two main micromap packages. Alternatively, there could be some table in Chapter 1 that lists features (and limitations) of the two packages when they are first introduced to the reader. Overall, this last point may be a subsection in the Introduction, or become a final chapter at the end of the book which research is necessary in the future.)
 - DONE in Sections 1.2.3, 1.2.4, and end of 2.1.
 - Problem of many polygons (**Juergen**: There are other problems, e.g., many islands as in the Philippines, narrow shapes as for Chile, subregions of considerably different sizes as for Russia)
 - DONE in Section 1.2.4.
 - Automating linked micromap plot production for reports and interactive visualizations (Check with Emma Jones at VA DEQ)

- I SUPPOSE DONE in Chapter 9.
 - (**Marcus:** Related, a disadvantage of both packages is their lack of tighter integration with ggplot. Not sure we want to go there though. – **Juergen:** I wouldn't go there as these are the packages that currently exist and have to be used.)
-

References

- Baulier, J. (August 18, 2011). Creating Micromaps in JMP [Web Page, <https://community.jmp.com/t5/JMP-Blog/Creating-micromaps-in-JMP/ba-p/30019>].
- Beecham, R., Dykes, J., Meulemans, W., Slingsby, A., Turkay, C., & Wood, J. (2017). Map LineUps: Effects of Spatial Structure on Graphical Inference [<https://doi.org/10.1109/TVCG.2016.2598862>]. *IEEE Transactions on Visualization and Computer Graphics*, 23(1), 391–400.
- Bell, S. B., Hoskins, R. E., Pickle, L. W., & Wartenberg, D. (2006). Current Practices in Spatial Analysis of Cancer Data: Mapping Health Statistics to Inform Policymakers and the Public [<https://doi.org/10.1186/1476-072X-5-49>]. *International Journal of Health Geographics*, 5, 49.
- Carr, D. B. (1994). *Converting Tables to Plots* (tech. rep. No. 101). Center for Computational Statistics, George Mason University, Fairfax, VA.
- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B., Bell, B. S., Pickle, L. W., Zhang, Y., & Li, Y. (2003). The State Cancer Profiles Web Site and Extensions of Linked Micromap Plots and Conditioned Choropleth Map Plots [<https://dl.acm.org/doi/10.5555/1123196.1123307>]. *Proceedings of the Third National Conference on Digital Government Research* (pp. 269–273). Digital Government Research Center (DGRC).
- Carr, D. B., Chen, J., Bell, B. S., Pickle, L. W., & Zhang, Y. (2002). Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps [<https://dl.acm.org/doi/10.5555/1123098.1123147>]. *Proceedings of the Second National Conference on Digital Government Research* (pp. 61–67). Digital Government Research Center (DGRC).
- Carr, D. B., & Nusser, S. M. (1995). Converting Tables to Plots: A Challenge from Iowa State. *Statistical Computing and Statistical Graphics Newsletter*, 6(3), 11–18.
- Carr, D. B., Olsen, A. R., Courbois, J.-Y. P., Pierson, S. M., & Carr, D. A. (1998). Linked Micromap Plots: Named and Described. *Statistical Computing and Statistical Graphics Newsletter*, 9(1), 24–32.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (1998). Boxplot Variations in a Spatial Context: An Omernik Ecoregion and Weather Example. *Statistical Computing and Statistical Graphics Newsletter*, 9(2), 4–13.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (2000). Using Linked Micromap Plots to Characterize Omernik Ecoregions [<https://doi.org/10.1023/A:1009828700017>]. *Data Mining and Knowledge Discovery*, 4(1), 43–67.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.

- Carr, D. B., & Pierson, S. M. (1996). Emphasizing Statistical Summaries and Showing Spatial Context with Micromaps. *Statistical Computing and Statistical Graphics Newsletter*, 7(3), 16–23.
- Carr, D. B., Valliant, R., & Rope, D. J. (1996). Plot Interpretation and Information Webs: A Time-Series Example from the Bureau of Labor Statistics. *Statistical Computing and Statistical Graphics Newsletter*, 7(2), 19–26.
- Carr, D. B., Wallin, J. F., & Carr, D. A. (2000). Two New Templates for Epidemiology Applications: Linked Micromap Plots and Conditioned Choropleth Maps [[https://doi.org/10.1002/1097-0258\(20000915/30\)19:17/18<2521::AID-SIM585>3.0.CO;2-K](https://doi.org/10.1002/1097-0258(20000915/30)19:17/18<2521::AID-SIM585>3.0.CO;2-K)]. *Statistics in Medicine*, 19(17–18), 2521–2538.
- Chen, J. X., Carr, D. B., Wechsler, H., & Pan, Z. (2006). Interactive Visualization of Multivariate Statistical Data [<https://doi.org/10.20870/IJVR.2006.5.3.2701>]. *The International Journal of Virtual Reality*, 5(3), 67–73.
- Das Gupta, D., & Wong, D. (2021). How “Dependent” Are We? A Spatiotemporal Analysis of the Young and the Older Adult Populations in the US [<https://doi.org/10.1007/s11113-020-09590-y>]. *Population Research and Policy Review*, 40(6), 1221–1252.
- Hurst, J., Symanzik, J., & Gunter, L. (2003). Interactive Federal Statistical Data on the Web Using “nViZn” [(CD & <http://interfacesymposia.org/I03/I2003Proceedings/HurstJon/HurstJon.paper.pdf>)]. *Computing Science and Statistics*, 35.
- Jones, L., & Symanzik, J. (2001). Statistical Visualization of Environmental Data on the Web Using nViZn [(CD & <http://interfacesymposia.org/I01/I2001Proceedings/LJones/LJones.pdf>)]. *Computing Science and Statistics*, 33.
- Mast, B. D. (2013b). Visualizing Same-Sex Couple Household Data with Linked Micromaps [<https://www.huduser.gov/portal/periodicals/cityscape/vol15num2/ch23.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 15(2), 267–271.
- Medri, J., Probst, B. D., & Symanzik, J. (2019). Housing Affordability and Immigration: An Exploratory Analysis in New York City [(CD)]. *2019 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Medri Cobos, J. (2021). *Housing Variables and Immigration: An Exploratory and Predictive Data Analysis in New York City* (Master’s thesis) [<https://doi.org/10.26076/dd0e-699c>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Monmonier, M. (1993). *Mapping It Out: Expository Cartography for the Humanities and Social Sciences*. University of Chicago Press, Chicago, IL.
- Olsen, A. R., Carr, D. B., Courbois, J.-Y. P., & Pierson, S. M. (1996). Presentation of Data in Linked Attribute and Geographic Space. *1996 Abstracts, Joint Statistical Meetings, Chicago, Illinois* (p. 271). American Statistical Association, Alexandria, VA.
- Payton, Q. C., McManus, M. G., Weber, M. H., Olsen, A. R., & Kincaid, T. M. (2015). micromap: A Package for Linked Micromaps [<https://doi.org/10.18637/jss.v063.i02>]. *Journal of Statistical Software*, 63(2), 1–16.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Porta, M., & Last, J. M. (2018). Choropleth Map [<https://www.oxfordreference.com/view/10.1093/acref/9780191844386.001.0001/acref-9780191844386-e-725>]. *A Dictionary of Public Health (Second Edition)*. Oxford University Press.
- Symanzik, J., Axelrad, D. A., Carr, D. B., Wang, J., Wong, D., & Woodruff, T. J. (1999). HAPs, Micromaps and GPL — Visualization of Geographically Referenced Statistical Summaries on the World Wide Web. *Annual Proceedings (ACSM-WFPS-PLSO-LSAW 1999 Conference CD)*. American Congress on Surveying & Mapping.

- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.
- Symanzik, J., & Carr, D. B. (2013). Linked Micromap Plots in R. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 213–218). Asian Regional Section of the IASC.
- Symanzik, J., Carr, D. B., Axelrad, D. A., Wang, J., Wong, D., & Woodruff, T. J. (1999). Interactive Tables and Maps — A Glance at EPA’s Cumulative Exposure Project Web Page. *1999 Proceedings of the Section on Statistical Graphics* (pp. 94–99). American Statistical Association, Alexandria, VA.
- Symanzik, J., Carr, D. B., McManus, M. G., & Weber, M. H. (2017). Micromaps [<https://doi.org/10.1002/9781118445112.stat07938>]. *Wiley StatsRef: Statistics Reference Online*. Wiley Online Library.
- Symanzik, J., Hurst, J., & Gunter, L. (2002). Recent Developments for Interactive Statistical Graphics on the Web Using “nViZn” [(CD)]. *2002 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., & Jones, L. (2001). “nViZn” Federal Statistical Data on the Web [(CD)]. *2001 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Wong, D., Wang, J., Carr, D. B., Woodruff, T. J., & Axelrad, D. A. (2000). Web-based Access and Visualization of Hazardous Air Pollutants [<https://webharvest.gov/peth04/20041025104701/http://www.atsdr.cdc.gov/gis/conference98/proceedings/pdf/symanzik.pdf>]. *Geographic Information Systems in Public Health: Proceedings of the Third National Conference, August 18–20, 1998, San Diego, California* (pp. 235–248). Agency for Toxic Substances & Disease Registry.
- Tatalovich, Z., & Stinchcomb, D. G. (2019). Creating Maps and Mapping Systems for Cancer Control and Prevention [https://doi.org/10.1007/978-3-030-18408-7_3]. In D. Berrigan & N. A. Berger (Eds.), *Geospatial Approaches to Energy Balance and Breast Cancer* (pp. 59–79). Springer Nature, Cham, Switzerland.
- Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press.
- Tufte, E. R. (1990). *Envisioning Information*. Graphics Press.
- Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press.
- United States Census Bureau. (2013). co34_d00_shp.zip [Shapefiles for New Jersey] [Web Page, <https://www2.census.gov/geo/tiger/PREVGENZ/co/co00shp/>] (accessed October 17, 2022).
- Wang, X., Chen, J. X., Carr, D. B., Bell, B. S., & Pickle, L. W. (2002). Geographic Statistics Visualization: Web-based Linked Micromap Plots [<https://doi.org/10.1109/5992.998645>]. *Computing in Science & Engineering*, 4(3), 90–94.
- Wartenberg, D. (2009). Some Considerations for the Communication of Results of Air Pollution Health Effects Tracking [<https://doi.org/10.1007/s11869-009-0046-y>]. *Air Quality, Atmosphere & Health*, 2(4), 207–221.
- Wickham, H., Cook, D., Hofmann, H., & Buja, A. (2010). Graphical Inference for Infovis [<https://doi.org/10.1109/TVCG.2010.161>]. *IEEE Transactions on Visualization and Computer Graphics*, 16(6), 973–979.
- Wilkinson, L., Rope, D. J., Carr, D. B., & Rubin, M. A. (2000). The Language of Graphics [<https://doi.org/10.1080/10618600.2000.10474897>]. *Journal of Computational and Graphical Statistics*, 9(3), 530–543.
- Wilkinson, L., Rope, D. J., Rubin, M. A., & Norton, A. (2001). nViZn: An Algebra-Based Visualization System [<https://www.semanticscholar.org/paper/nViZn-%3A-An-Algebra-Based-Visualization-System-Wilkinson/59c5ed82acbfe5fe043be0bdf00338>]

- [5e279c822f\]. 1st International Symposium on Smart Graphics, March 21-23, 2001, Hawthorne, NY, USA.](#)
- Xie, Y. (2022a). *bookdown: Authoring Books and Technical Documents with R Markdown* [R package version 0.29 (<http://CRAN.R-project.org/package=bookdown>)].
- Xie, Y. (2022b). *knitr: A General-Purpose Package for Dynamic Report Generation in R* [R package version 1.40 (<http://CRAN.R-project.org/package=knitr>)].

2

*Linked Micromap Plots via the **micromap** R Package*

JÜRGEN SYMANZIK, MARCUS W. BECK, MICHAEL G. McMANUS

The **micromap** R package (Payton & Olsen, 2024), accessible at <https://cran.r-project.org/web/packages/micromap/index.html>, will be introduced in this chapter. The reader will learn how to make use of the four main steps that are required to create a basic linked micromap plot via this R package. Details will be provided how to optimize and fine-tune such a basic plot into a publication-worthy final linked micromap plot. Example linked micromap plots are created for education and poverty data for the 50 states (and Washington, D.C.) of the United States (U.S.) and for watersheds in West Virginia (one of the 50 U.S. states).

2.1 Introduction

As discussed in Chapter 1, linked micromap plots were originally presented at the Joint Statistical Meetings (JSM) in Chicago, Illinois, in 1996 (Olsen et al., 1996). They quickly gained popularity among researchers at United States (U.S.) Federal Agencies such as the U.S. Department of Agriculture – National Agricultural Statistics Service (USDA–NASS), various branches of the U.S. Environmental Protection Agency (USEPA), the National Cancer Institute (NCI), the U.S. Census Bureau, and the U.S. Bureau of Labor Statistics (BLS). Early main applications of linked micromap plots can be found in the environmental field (Carr, Olsen, Pierson, et al., 1998, 2000) and the medical field (Carr et al., 2003; Carr, Chen, et al., 2002).

While early linked micromap plots were created via S-Plus and Java, later ones were created in R (see Symanzik and Carr (2013) for an overview). However, even with the availability of R code that was provided in support of Carr and Pickle (2010), creating linked micromap plots was challenging which considerably limited their use. In fact, Payton et al. (2012) observed that “Producing LMplots [...] has typically been somewhat difficult, and therefore LMplots have seen limited use.” However, linked micromap plots continued to play an important role at the USEPA. Eventually, a team of researchers including Anthony R. Olsen, Quinn C. Payton, Michael G. McManus, Marc H. Weber, and Thomas M. Kincaid, all originally with the USEPA in Corvallis, Oregon, started to develop an R package for linked micromap plots. First uses of this package can be seen in poster presentations in May 2012 (Payton et al., 2012) and April 2013 (Payton et al., 2013). At about the same time, in December 2012, the first publicly available version of the **micromap** R package (version 1.5), was released to CRAN (Payton & Olsen, 2012). Eventually, Marcus W. Beck, then also with the USEPA in Gulf Breeze, Florida, joined the team of the original developers

with the release of version 1.9.3 of this R package in February 2018 (Payton & Olsen, 2018) and has also served as the maintainer of this R package since then.

As frequently happens in an open software environment such as R, the ***micromap*** R package is not the only R package for linked micromap plots. Independently, but motivated by similar past uses and resources, the ***micromapST*** R package has been developed in parallel and its first version (version 1.0) was released to CRAN in June 2013 (Carr & Pearson Jr., 2013), i.e., only a few months after the first release of the ***micromap*** R package. The ***micromapST*** R package will be extensively discussed in Chapter 3.

From a user's perspective, there are little differences in the appearance and quality of the final linked micromap plots that can be created by these two R packages. There are, however, differences in details in how linked micromap plots are created in each package that can be important for users. Perhaps the two biggest differences between the two packages are that the ***micromap*** R package makes it easy to bring in one's own boundary files, in particular external shapefiles, while the ***micromapST*** R package initially supports a larger number of glyph types. However, even these differences are relatively minor as users can create their own plot types for use in the ***micromap*** R package as discussed in Chapter 6 and incorporate external shapefiles, into the ***micromapST*** R package as discussed in Chapter 5. Ultimately, the decision is up to the analyst on which of these two R packages to use for the construction of linked micromap plots.

The remainder of this chapter is organized as follows: In Section 2.2, we will introduce the four main steps that are required to create a basic linked micromap plot via the ***micromap*** R package. Examples in Section 2.3 and Section 2.4 will outline how to apply these steps to data for the 50 U.S. States and Washington, D.C. and to data for watersheds in West Virginia, respectively. This chapter concludes with a summary and suggestions for further reading in Section 2.5.

2.2 Steps to Create a Linked Micromap Plot with the ***micromap*** R Package

Four main steps, shown in Figure 2.1, are needed to create a linked micromap plot with the ***micromap*** R package:

1. **Identifying and Geoprocessing of Spatial Boundary Data** (see Sections 2.3.1 and 2.4.1 for details): In addition to a data frame that contains the statistical data, the user must identify a data structure that contains the spatial boundary data for the region and subregions that are colored in the maps. This spatial boundary data typically comes in the form of a `SpatialPolygonsDataFrame`. In this chapter, we will work with ready-to-use boundary files. In Chapter 4, we will discuss how to make use of boundary files that are provided as external shapefiles. In particular, we will see in that chapter how to simplify complex boundaries, enlarge small subregions in the maps, and move subregions closer that are far from the main area of the map.
2. **Linking Spatial Boundary Data and Statistical Data** (see Sections 2.3.2 and 2.4.2 for details): To link spatial boundary data and statistical data, the `SpatialPolygonsDataFrame` first has to be transformed into a regular data frame via the `create_map_table()` function. Next, we have to identify one variable from

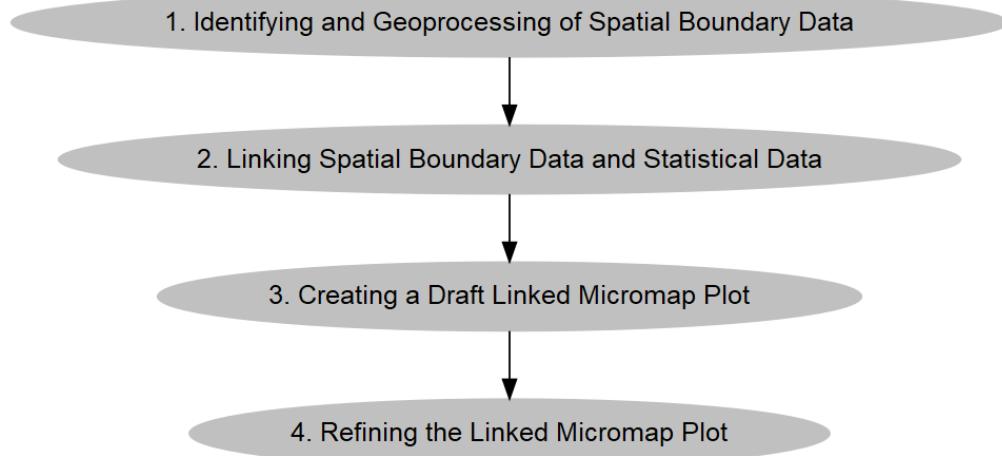


FIGURE 2.1 Workflow to create a linked micromap plot with the ***micromap*** R package (Diagram created with the **DiagrammeR** R package (Iannone, 2022)).

the statistical data frame and one variable from the newly created data frame with the boundary information that allow us to link statistical data and boundary data for each subregion. This step is not required when the spatial boundary data and statistical data are adequately stored in a simple features (sf) format (Open Geospatial Consortium, 2022; Pebesma, 2018) instead of a SpatialPolygonsDataFrame. From a user’s perspective, we think that the approach that links a data frame with the statistical data to another data frame with the underlying spatial boundary data is the easier approach for most users. First, this approach is most intuitive if the statistical data is stored in an external Excel file that is read into a data frame in R before any linked micromap plot is created. Second, this approach does not require any familiarity with spatial objects in R and spatial data in general. For more advanced users, we recommend experimenting with the approach that is based on the simple features format, in particular if the external shapefile contains both, the statistical data and the boundary data. While most of the linked micromap plots in this book that are created via the ***micromap*** R package make use of the more traditional approach that links the two data frames, Example 2 (in Section 2.4) demonstrates how both approaches can be used for the same underlying data sets. One example in Section 13.3 directly creates a linked micromap plot based on the simple features option.

3. **Creating a Draft Linked Micromap Plot** (see Sections 2.3.3 and 2.4.3 for details): While not necessary, it is always a good idea to first create a minimal linked micromap plot to ensure that the statistical data and boundary data are matching and a correct draft linked micromap plot is created. Skipping this step and trying to create a complex linked micromap plot immediately may complicate debugging the R code.
4. **Refining the Linked Micromap Plot** (see Sections 2.3.4 and 2.4.4 for details):

Once a draft linked micromap plot has been created, this plot usually needs fine-tuning of its appearance and plot aesthetics, e.g., modification of colors, change of the layout and of perceptual groups, addition of labels and legends, and possibly the addition of additional statistical variables or changes to different graph types for some of the variables.

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C.

In this first linked micromap plot example created with the **micromap** R package, we work with the *USstates* and *edPov* datasets from the **micromap** R package. We follow the four steps outlined in Section 2.2.

2.3.1 Identifying and Geoprocessing of Spatial Boundary Data

USstates is a SpatialPolygonsDataFrame for the 50 U.S. states (and Washington, D.C.) that was created for use with linked micromap plots. Notably, the boundaries of many of the states have been simplified, Alaska and Hawaii have been moved closer to the contiguous 48 states (and also have been resized), and Washington, D.C. has been pulled out of the main map, placed further to the east, and also has been enlarged as shown in Figure 2.2. In the following R code, we first load the **micromap** R package and the *USstates* dataset, verify that this object indeed is a SpatialPolygonsDataFrame, then look at some of the data in the *data* component of this object, and finally plot it. For figures that only contain maps, it is often helpful to remove all margin space on all four sides of the plot via `par(mar = c(0, 0, 0, 0))` to maximize the plot, i.e., the actual map. When loading the **micromap** R package, the reader will notice that it depends on the **RColorBrewer** (Neuwirth, 2022), **sp** (Pebesma & Bivand, 2022), and **sf** (Pebesma, 2022) R packages. These packages are dependencies for **micromap** and are installed automatically with the package.

```
library(micromap)

data(USstates)

class(USstates)

## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"

head(USstates@data)

##   ST    ST_NAME AREA_KM PERIM_KM
## 0 AK    Alaska 1506038    60261
## 1 AL  Alabama 133761     2355
## 2 AR Arkansas 137734     2172
## 3 AZ  Arizona 295267     2395
## 4 CA California 409603     5682
```

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 35

```
## 5 CO Colorado 269600 2100
```

```
par(mar = c(0, 0, 0, 0))
plot(USstates)
```

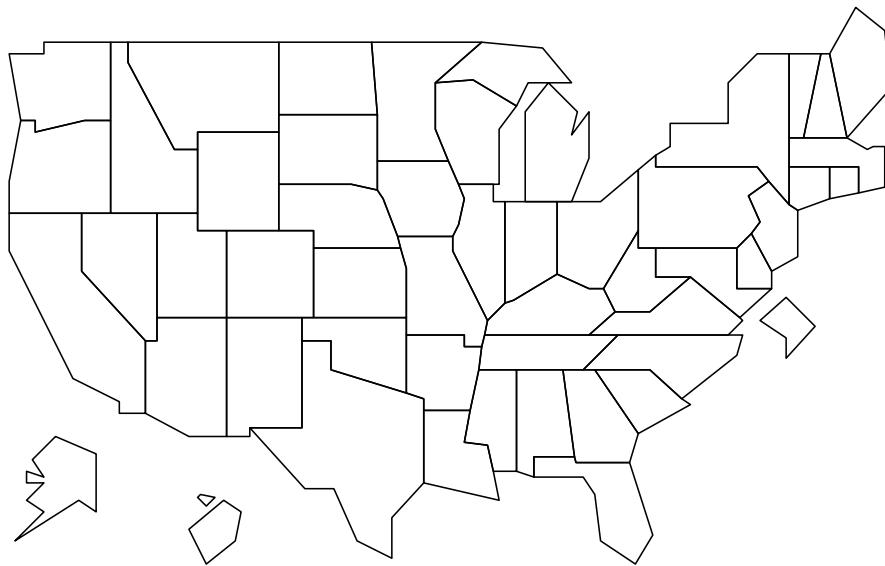


FIGURE 2.2 Map representation of the *USstates* spatial boundary dataset for the United States that frequently is used as the basis for linked micromap plots that are created with the **micromap** R package.

2.3.2 Linking Spatial Boundary Data and Statistical Data

In this step, the `SpatialPolygonsDataFrame` first is transformed into a regular data frame for use in a linked micromap plot via the `create_map_table()` function. We have to indicate a variable from the `data` component of the `SpatialPolygonsDataFrame` object that can be used as an ID column. A matching ID column must also be identified in the statistical dataset for linking with the spatial boundary dataset. For the *USstates* dataset, this is usually the `ST` variable that contains the 51 abbreviations for the 50 U.S. states (and for Washington, D.C.).

```
head(USstates@data$ST)
```

```
## [1] AK AL AR AZ CA CO
## 51 Levels: AK AL AR AZ CA CO CT DC DE FL GA HI IA ID IL IN KS KY LA MA ... WY
```

```
state_polys_table <- create_map_table(
  tmp.map = USstates,
  IDcolumn = "ST"
)
```

```
class(state_polys_table)

## [1] "data.frame"

dim(state_polys_table)

## [1] 460   8

names(state_polys_table)

## [1] "ID"      "region"   "poly"     "coordsx"  "coordsy"  "hole"
## [7] "plotorder" "plug"
```

The resulting `state_polys_table` is a regular data frame that will be used for creating a linked micromap plot in the next step. `state_polys_table` consists of 460 rows. This implies that the map is based on 460 line segments.

`edPov` is a data frame that contains education and poverty level data for the 50 U.S. states (and Washington, D.C.). This data frame has 51 rows, one for each of the 50 U.S. states (and one for Washington, D.C.). We have to identify one variable from this data frame that can be used for linking the two datasets. Here, this is the `StateAb` variable.

The last expression in the following R code verifies that there is indeed at least one matching ID in the `state_polys_table` data frame for each ID in the `edPov` dataset. Missing IDs in the statistical data frame may prevent data from appearing on the maps. Similarly, mismatching identifiers in the statistical data frame may also prevent data from being shown on the maps, e.g., if the statistical data frame uses `D.C.` as ID while the spatial data frame uses `DC`. Here, everything is matching.

```
data(edPov)
dim(edPov)

## [1] 51  5

head(edPov)

##      state ed pov region StateAb
## IL Illinois 26.1 10.7    MW    IL
## IN Indiana 19.4  9.5    MW    IN
## IA Iowa 21.2  9.1    MW    IA
## KS Kansas 25.8  9.9    MW    KS
## MI Michigan 21.8 10.5   MW    MI
## MN Minnesota 27.4  7.9   MW    MN

head(edPov$StateAb)

## [1] "IL" "IN" "IA" "KS" "MI" "MN"
```

```
all(sort(edPov$StateAb) == sort(unique(state_polys_table$ID)))
## [1] TRUE
```

2.3.3 Creating a Draft Linked Micromap Plot

Now we can create a minimal draft linked micromap plot using the `mmplot()` function, based on the previously created `state_polys_table` data frame and the `edPov` dataset. In our function call, seven arguments are required, none of which have a default value. For all other arguments of this function, the default settings will be used here. The statistical data (`edPov`) is assigned to the `stat.data` argument and the spatial boundary data (in the `state_polys_table` data frame) is assigned to the `map.data` argument.

The `map.link` argument is needed to link the statistical data and the spatial boundary data. A vector with the names of the two linking variables identified in the previous step is needed for this argument. The first variable name (`StateAb`) must come from the statistical data (here `edPov`) and the second variable name (`ID`) must come from the `state_polys_table` data frame that contains the spatial boundary information. Changing the order of these two variable names typically results in an error.

The `panel.types` and `panel.data` arguments are closely related. As the name suggests, `panel.types` is a vector that specifies the layout of the columns of panels in the linked micromap plot. Here, we have panels with a `dot_legend` in the first column, `labels` in the second column, the statistical data represented as dotplots (`dot`) in the third and fourth columns, and the micromaps in the fifth, i.e., final, (`map`) column. This is matched with a list of data that is used for each of the five columns of panels. A list is necessary for this argument as some of the data itself can be lists as we will see in the R code for Figures 2.14 and 2.15 later on.

The `dot_legend` simply shows a plotting symbol in a certain color that represents that row in the linked micromap plot. Thus, no further data is needed and `NA` is assigned. `labels` requires some text argument, typically some identifiers of the subregions in the maps. Here `state` from `edPov` is used. The variables `pov` and `ed` from `edPov` are used for the statistical displays in columns three and four. It should be noted that all data specified in `panel.data` by default is taken from the data frame specified in the `stat.data` argument. The fifth and final column contains the micromaps. Their boundaries are obtained from the `map.data` data frame. Thus, no further data has to be specified here and `NA` is assigned instead.

Two more arguments have to be specified: `ord.by` specifies the sorting variable of the rows in the linked micromap plot. Here, `pov` from the `stat.data` argument is used. The sorting of the rows goes from smallest (at the top) to largest (at the bottom). Finally, `grouping` specifies the number of rows in each of the perceptual groups. If a single integer value is provided, that value is used for all perceptual groups. If a vector of integer values is provided, each perceptual group may have a different number of rows as shown in the R code for Figure 2.6. Here, `grouping` is set to 5, meaning there are five rows of data in each perceptual group.

The resulting draft linked micromap plot is shown in Figure 2.3. It is noteworthy that there are eleven perceptual groups overall — ten with five subregions and one with just one subregion. This is because there are 51 subregions overall: The 50 U.S. states and Washington, D.C. This results in only one row of data and one subregion highlighted in the final (bottom) perceptual group. There is no automatic balancing of the number of rows in each perceptual group. A layout such as the default one shown in Figure 2.3 should

be avoided in general. Table 1.1 in Chapter 1 provides suggestions how to group data for various numbers of subregions. We will also address this as one of the refinement steps in the next section.

We abstain from interpreting this figure at this stage, but we will provide some helpful interpretation once we have created the last refined version of this figure at the end of the next section. However, we encourage the reader to examine the draft maps at this time and determine whether any spatial patterns may be visible and to assess the statistical relationship between the two variables shown in the third and fourth columns of the plot.

```
mmpplot(
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("dot_legend", "labels", "dot", "dot", "map"),
  panel.data = list(NA, "state", "pov", "ed", NA),
  ord.by = "pov",
  grouping = 5
)
```

2.3.4 Refining the Linked Micromap Plot

We continue with the draft linked micromap plot from the previous section and refine it in multiple small steps. This refinement process should only be started once functional R code has been obtained in the previous step and an initial linked micromap plot has been created.

First, we remove the eleventh perceptual group (with just one subregion) and introduce a median row via `median.row = TRUE` instead. A median row is often a good solution if the number of subregions is odd such as for the 50 U.S. states (and Washington, D.C.). Here, Wyoming is the state shown in the median row. It has the 26th highest (or lowest) value for the sorting variable, i.e., `pov`. It does not appear in a map by itself, but rather is added to the perceptual groups above and below the median row in a neutral color, thus increasing the number of subregions shown in each of these two maps by one (i.e., six here). Also, we reverse the sorting order via `rev.ord = TRUE`. Now the sorting of the rows goes from largest (at the top) to smallest (at the bottom). The resulting linked micromap plot is shown in Figure 2.4. While we keep `pov` as the sorting variable in this first refined version, the reader is encouraged to use `ed` as the sorting variable to see how the spatial patterns highlighted in the maps change. This can be done for the original sorting order or for the reversed sorting order. The rows can even be sorted by `region` or alphabetically by `state` or `StateAb` even though such an alphabetical sorting in most cases is not very meaningful.

```
mmpplot(
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("dot_legend", "labels", "dot", "dot", "map"),
  panel.data = list(NA, "state", "pov", "ed", NA),
  ord.by = "pov",
  rev.ord = TRUE,
```

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 39

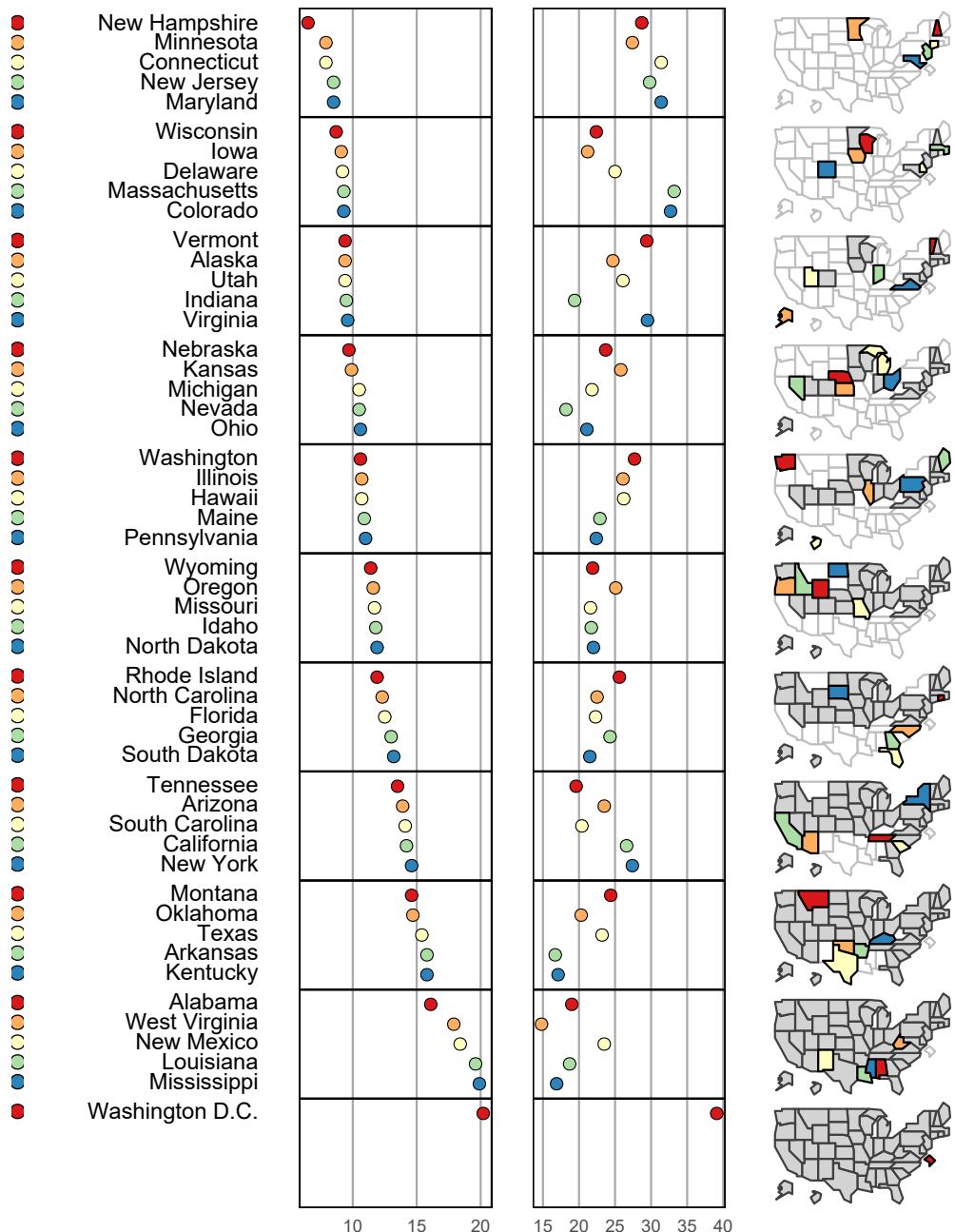


FIGURE 2.3 Draft linked micromap plot, based on the *edPov* dataset.

```

grouping = 5,
median.row = TRUE
)

```

We continue with modifications to the first refined linked micromap plot. Next, we change the order of the two statistical graphics columns, i.e., we place `ed` to the left of `pov` and use `ed` as the sorting variable (in reverse order). While any column of the linked micromap plot or even variables not shown can be used as sorting variables, in most cases, the first (leftmost) statistical graphics column is used for sorting. Moreover, we place the maps on the left side of the plot. As previously stated, the `panel.types` and `panel.data` arguments of the `mmplot()` function are closely related. Thus, if we change the order of one, the order of the other one has to be changed accordingly. The resulting linked micromap plot is shown in Figure 2.5. As stated in Section 1.2, there is no strong recommendation where the column with the maps should be placed.

```

mmplot(
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, NA, "state", "ed", "pov"),
  ord.by = "ed",
  rev.order = TRUE,
  grouping = 5,
  median.row = TRUE
)

```

We continue making changes to the second refined linked micromap plot. We change the grouping to nine perceptual groups overall (and no median row) via `grouping = c(6, 6, 6, 6, 3, 6, 6, 6, 6)` and `median.row = FALSE` (which is the default and could be omitted) and vertically align the rows in each perceptual group via `vertical.align = "center"`. This grouping is not a recommended partitioning from Table 1.1 in Chapter 1 and is mostly done for experimental purposes here. Finally, we start making changes to individual columns of the plot via the `panel.att` argument. Here, the third column that shows the `labels` is aligned on the left. The resulting linked micromap plot is shown in Figure 2.6.

```

mmplot(
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, NA, "state", "ed", "pov"),
  ord.by = "ed",
  rev.order = TRUE,
  grouping = c(6, 6, 6, 6, 3, 6, 6, 6, 6),
  median.row = FALSE,
  vertical.align = "center",
  panel.att = list(list(3, align = "left"))
)

```

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 41

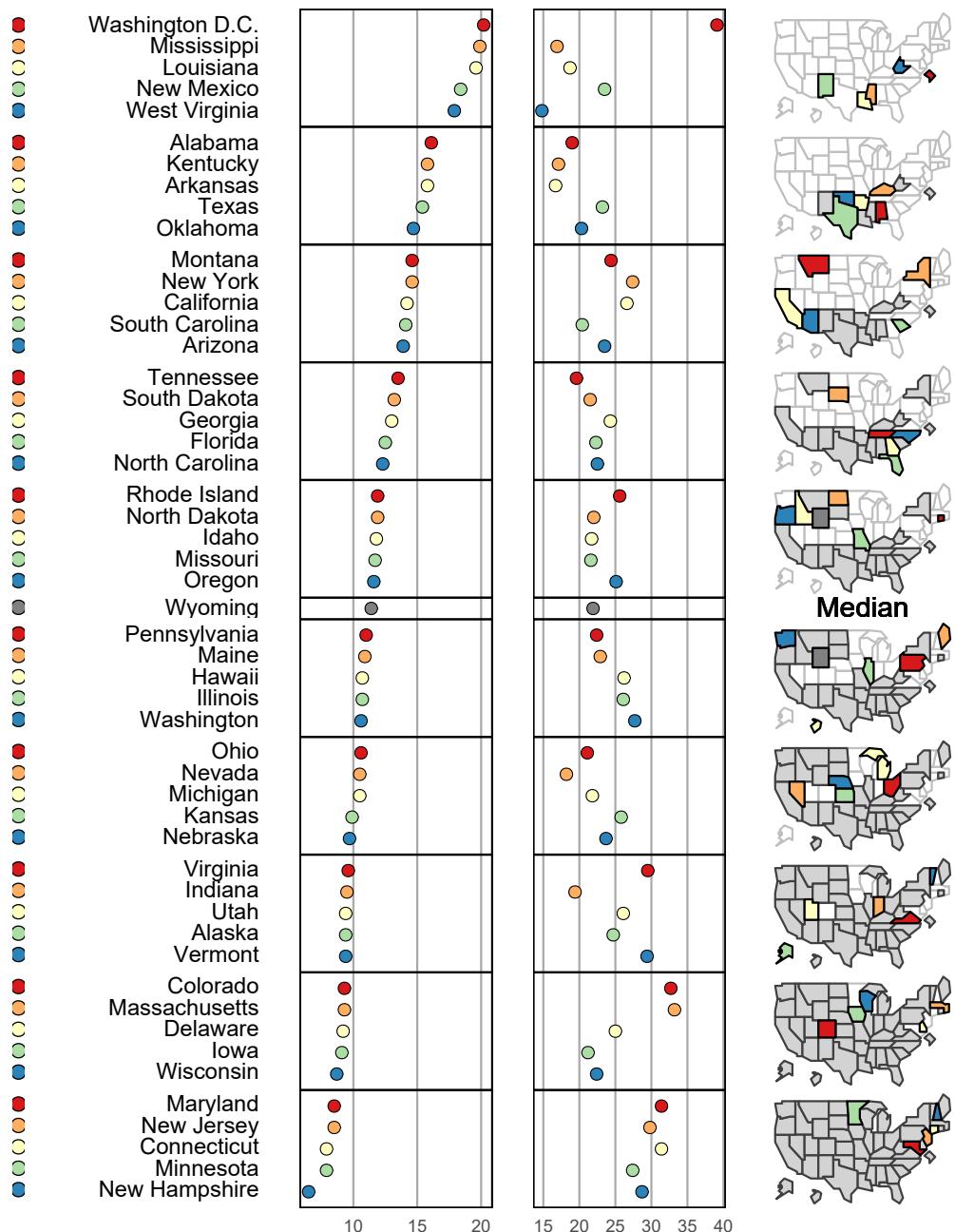


FIGURE 2.4 First refined linked micromap plot, based on the *edPov* dataset. Main changes are the introduction of a median row instead of the eleventh perceptual group and the reverse ordering of the *pov* data in the first statistical graphics column.

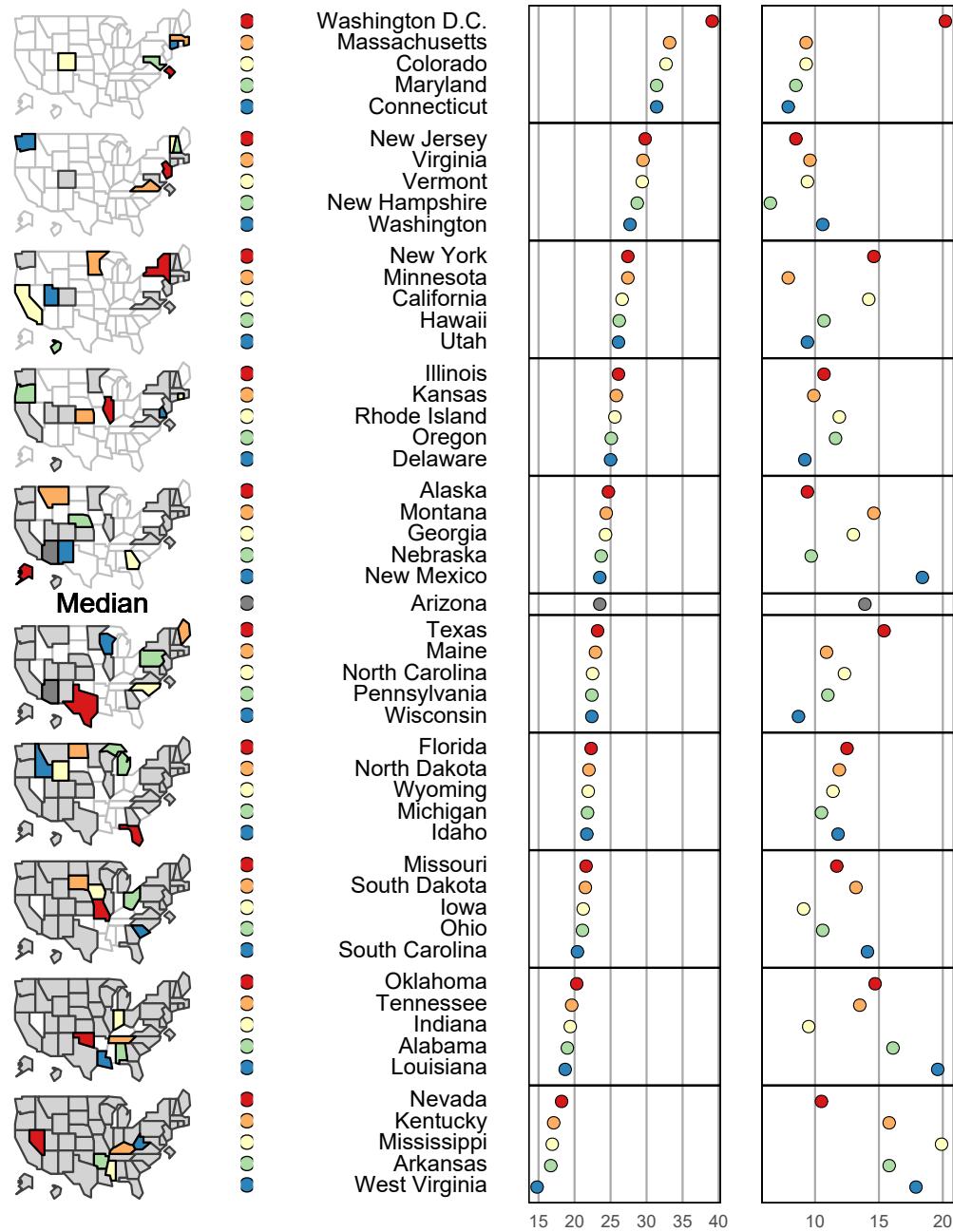


FIGURE 2.5 Second refined linked micromap plot, based on the *edPov* dataset. Main changes are related to the order of the five columns in the plot. Most notable, the map column is shown on the left here.

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 43

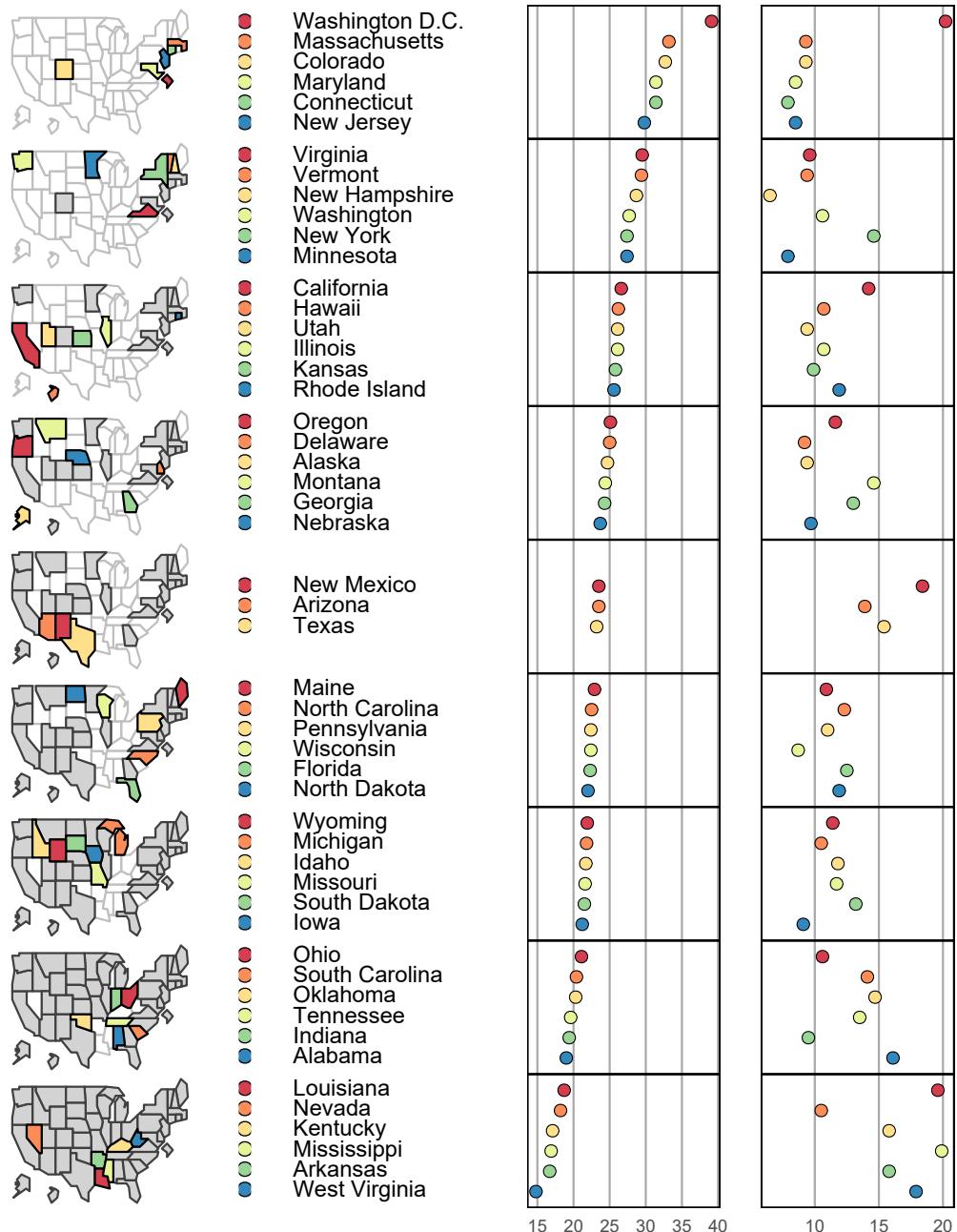


FIGURE 2.6 Third refined linked micromap plot, based on the *edPov* dataset. Main changes are related to perceptual groups with different numbers of subregions and the vertical alignment of rows in the middle of the plot. Also, labels are aligned on the left.

After this experiment with a different grouping, we revert back to the more traditional grouping of ten perceptual groups with five rows each and a median row that is frequently used for the 50 U.S. states (and Washington, D.C.) as suggested as Partitioning 1 from Table 1.1 in Chapter 1. Next, the choice of colors for the subregions in each map (and thus for the `dot_legend` appearance in the `panel.type` argument as well) is discussed. The default setting for the `colors` argument makes use of `max(grouping)` different colors from a spectral color scheme. Thus, when `grouping = 5` as in Figures 2.3-2.5, a five-class spectral color scheme is selected, whereas a six-class spectral color scheme is selected when `grouping = c(6, 6, 6, 6, 3, 6, 6, 6, 6)` as in Figure 2.6. Historically, many linked micromap plots, e.g., in Carr, Olsen, Courbois, et al. (1998) and Wang et al. (2002), made use of rainbow colors that could be obtained via the `colors = c("red", "orange", "green", "blue", "purple")` setting of the `colors` argument. While these colors work well for readers with normal color vision, they may not work well for readers with certain types of color vision deficiencies. Instead, some color schemes that are colorblind safe are better suited for such readers. Options are single-hue or multi-hue sequential color schemes or selected divergent color schemes. Such color schemes can be obtained from the **RColorBrewer** R package (Neuwirth, 2022). The reader is encouraged to read more about the theoretical background of these color schemes in Brewer et al. (2003) and Harrower and Brewer (2003) and experiment with different settings at the supporting web page at <https://colorbrewer2.org/>. Symanzik et al. (2014) used a five-class greyscale sequential color scheme from **RColorBrewer** in reverse sorting (where the darkest grey color comes first) for publication in a greyscale publication, obtained via `colors = RColorBrewer::brewer.pal(n = 5, name = "Greys")[5:1]`. Symanzik et al. (2016) used a five-class divergent red-yellow-blue (RdYlBu) color scheme from **RColorBrewer** that is colorblind safe and print friendly, obtained via `colors = RColorBrewer::brewer.pal(n = 5, name = "RdYlBu")`. In the linked micromap plot shown in Figure 2.7, we use a five-class divergent brown-blue-green (BrBG) color scheme from **RColorBrewer** that is also colorblind safe and print friendly, obtained via `colors = RColorBrewer::brewer.pal(n = 5, name = "BrBG")`.

```
mmpplot(  
  stat.data = edPov,  
  map.data = state_polys_table,  
  map.link = c("StateAb", "ID"),  
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),  
  panel.data = list(NA, NA, "state", "ed", "pov"),  
  ord.by = "ed",  
  rev.ord = TRUE,  
  grouping = 5,  
  median.row = TRUE,  
  colors = RColorBrewer::brewer.pal(n = 5, name = "BrBG"),  
  panel.att = list(list(3, align = "left"))  
)
```

So far, labels for the columns and titles for the statistical graphics columns are not shown. Tic marks and tic mark labels, in particular in the second statistical graphics column, could be improved, background colors in the statistical graphics columns and maps could be modified, font and symbol sizes could be modified, and the widths of the columns could be adjusted. All of this is done via the `panel.att` argument that controls the panel specific attributes of each column in the linked micromap plot. The content of this argument typically is a list of lists where the attributes for each column of the linked micromap plot

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 45

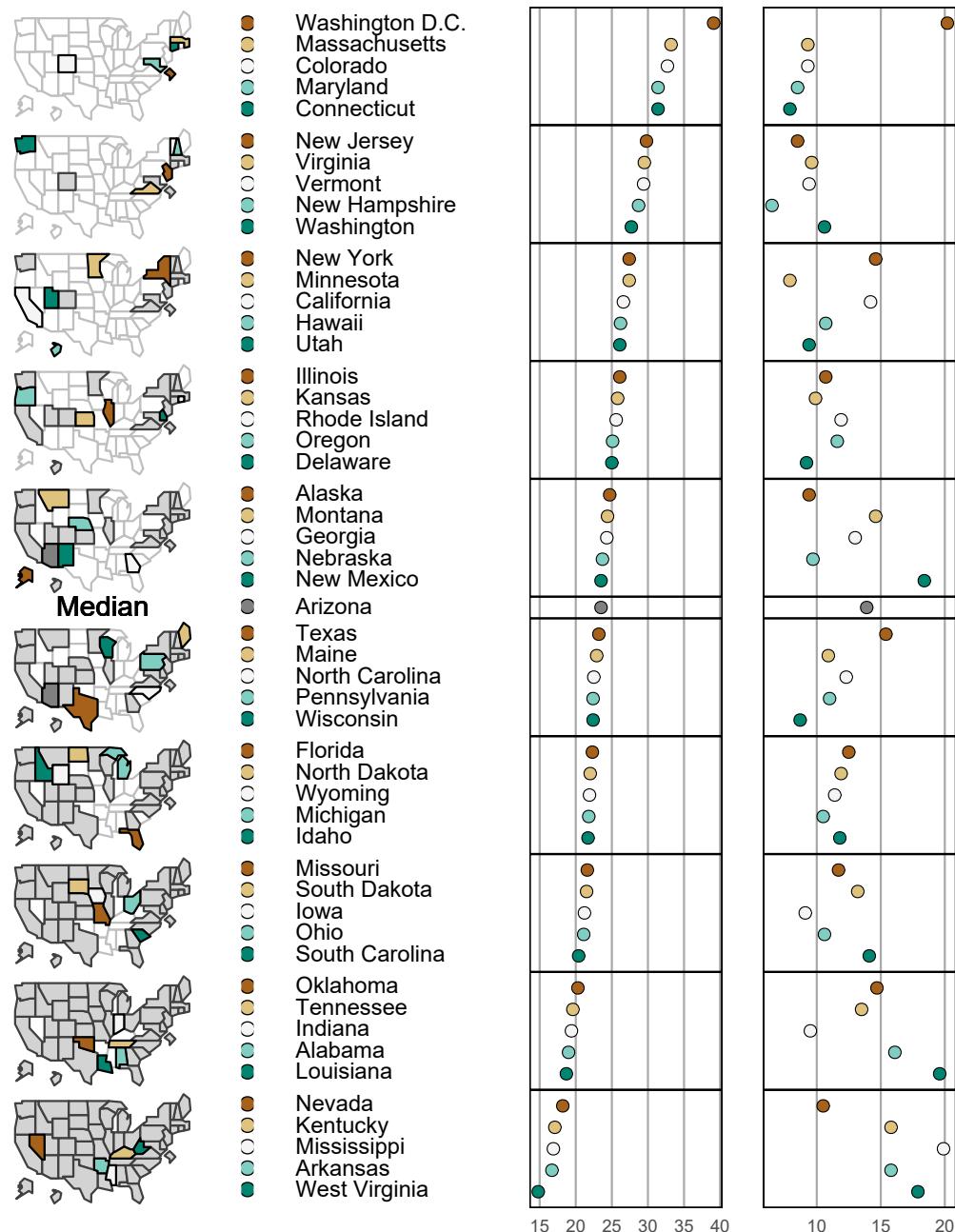


FIGURE 2.7 Fourth refined linked micromap plot, based on the *edPov* dataset. Main changes are the use of a divergent brown-blue-green color scheme and the conversion back to the traditional grouping for 51 subregions.

are modified via a separate list. These inner lists are numbered from 1 to the number of elements in the `panel.types` vector where 1 is related to `map`, 2 to `dot_legend`, 3 to `labels`, and 4 and 5 to `dot`, i.e., the dotplots in the two statistical graphics columns, in the linked micromap plot shown in Figure 2.8.

We leave it to the reader to further experiment with the different elements in these lists. If the purpose of a certain element is not immediately obvious, it is useful to considerably increase or decrease the numeric value of that element or change the color to red or yellow to highlight that element.

Here, we only want to explain the purpose of the `fill.regions` element with the matching `header` element in the list for column 1, i.e., the `map` column: The setting `fill.regions = "aggregate"` (which in fact is the default setting) fills in the subregions from all previous perceptual groups in the subsequent perceptual groups. This filing proceeds from the top perceptual group to the bottom perceptual group by sequentially filling the subregions that have already been displayed. Thus, in the map for the final perceptual group at the bottom, all subregions have been filled. The text in the `header` is used to communicate this information to the reader. Alternatively, the setting `fill.regions = "with data"` only fills those subregions in a map that actually show data in that perceptual group. No additional subregions are filled in any of the maps. Another setting for `fill.regions` is discussed in the next refinement step.

```
mmpplot
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, NA, "state", "ed", "pov"),
  ord.by = "ed",
  rev.ord = TRUE,
  grouping = 5,
  median.row = TRUE,
  colors = RColorBrewer::brewer.pal(n = 5, name = "BrBG"),
  panel.att = list(
    list(
      1,
      header = "Light Gray Means\nPreviously Displayed",
      map.all = TRUE,
      fill.regions = "aggregate",
      active.border.color = "black",
      active.border.size = 1.2,
      inactive.border.color = gray(0.7),
      inactive.border.size = 1,
      panel.width = 0.85
    ),
    list(
      2,
      point.type = 20,
      point.border = TRUE,
      point.size = 2,
      panel.width = 1.0
    )
  )
}
```

```
),
list(
  3,
  header = "States",
  align = "left",
  text.size = 0.9,
  panel.width = 0.75
),
list(
  4,
  header = "Percent Adults With\n4+ Years of College",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  xaxis.ticks = list(10, 20, 30, 40),
  xaxis.labels = list(10, 20, 30, 40),
  xaxis.title = "Percent"
),
list(
  5,
  header = "Percent Living Below\nPoverty Level",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  xaxis.ticks = list(5, 10, 15, 20),
  xaxis.labels = list(5, 10, 15, 20),
  xaxis.title = "Percent"
)
)
```

In the final refined linked micromap plot shown in Figure 2.9, we make three more types of changes. First, we make use of the `labeling::extended()` function of the `labeling` R package (Talbot, 2020). This package is not included with `micromap` and it must be installed separately. This function is provided with the minimum and maximum values of a variable and the tentative number of tic marks and tic marks labels for that variable and it then creates a vector with near-optimal axis labels. The argument `m` is used as a guideline for the number of axis labels, but the actual number of near-optimal axis labels may differ slightly. The reader is encouraged to experiment with `m = 2` to `m = 6` in the R code below.

Second, we use the setting `fill.regions = "two ended"`. This setting makes most sense when `median.row = TRUE` (as is the case here) and the focus of the maps is to indicate which subregions are above or below the median value of the variable specified in the `ord.by` argument (here `ed`). Similar to the setting `fill.regions = "aggregate"`, the subregions from all previous perceptual groups are filled in the subsequent perceptual groups. This filling proceeds from the top perceptual group to the median row and from the bottom perceptual group to the median row by sequentially filling the subregions that have already been displayed on the more extreme ends. The text in the header has been updated to communicate this information to the reader.

Third, we reduce the margin space between some of the columns via the `right.margin` and `left.margin` settings. Negative values are allowed. Fine-tuning the margin spacing and the widths of the columns can require a few iterations. The reader always should check carefully

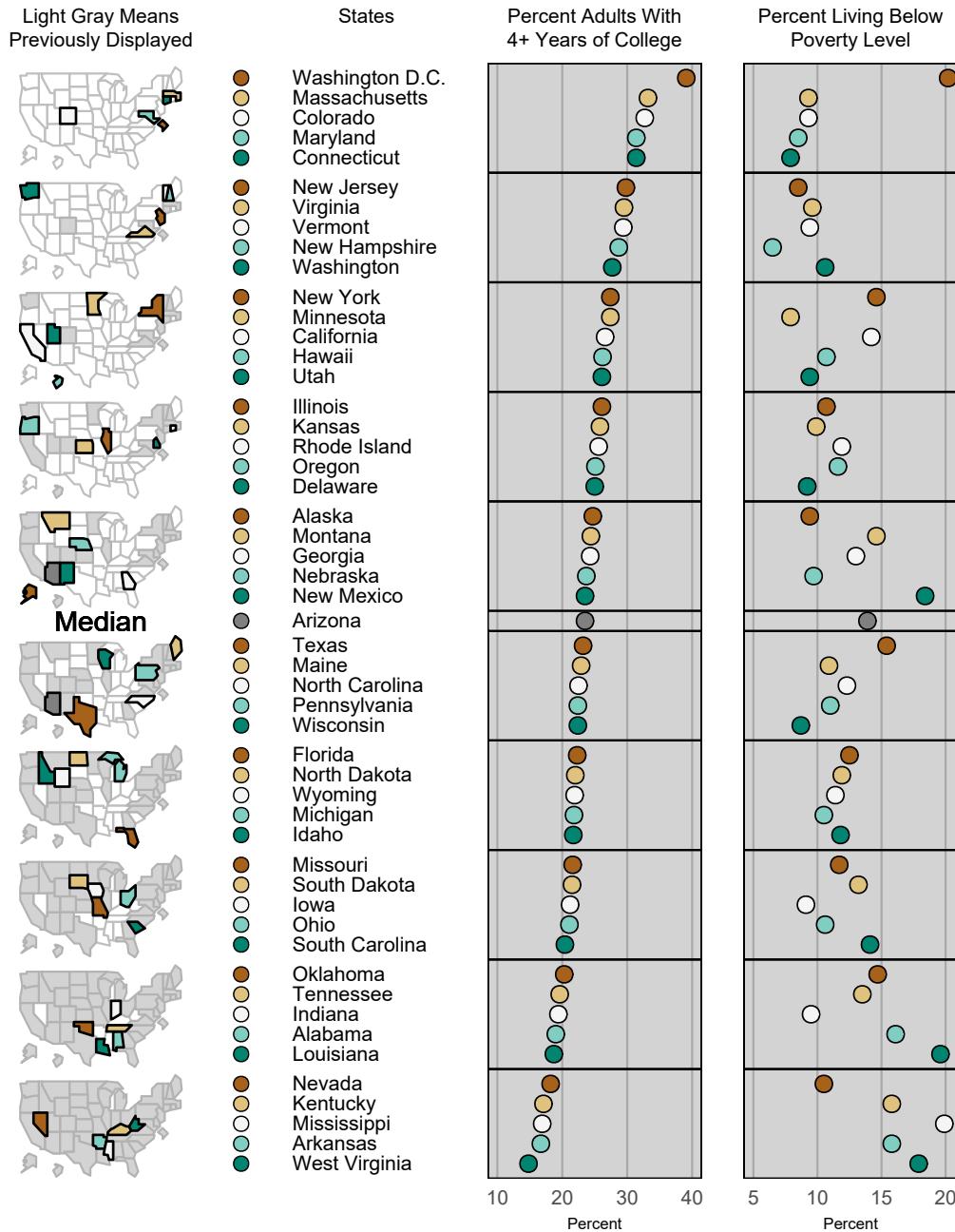


FIGURE 2.8 Fifth refined linked micromap plot, based on the *edPov* dataset. Main changes are related to the panel specific attributes.

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 49

that no identifiers are truncated or overprinted, in particular that no letters from the longest identifier (here Washington D.C.) are cut off.

```
library(labeling)

mmpplot(
  stat.data = edPov,
  map.data = state_polys_table,
  map.link = c("StateAb", "ID"),
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, NA, "state", "ed", "pov"),
  ord.by = "ed",
  rev.ord = TRUE,
  grouping = 5,
  median.row = TRUE,
  colors = RColorBrewer::brewer.pal(n = 5, name = "BrBG"),
  panel.att = list(
    list(
      1,
      header = "Two-ended\nCumulative Maps",
      map.all = TRUE,
      fill.regions = "two ended",
      active.border.color = "black",
      active.border.size = 1.2,
      inactive.border.color = gray(0.7),
      inactive.border.size = 1,
      panel.width = 0.85
    ),
    list(
      2,
      point.type = 20,
      point.border = TRUE,
      point.size = 2,
      panel.width = 1.0
    ),
    list(
      3,
      header = "States",
      align = "left",
      right.margin = 0,
      left.margin = -1,
      text.size = 0.9,
      panel.width = 0.75
    ),
    list(
      4,
      header = "Percent Adults With\n4+ Years of College",
      graph.bgcolor = "lightgray",
      right.margin = 0,
      left.margin = -0.6,
    )
  )
)
```

```

point.size = 1.5,
xaxis.ticks = as.list(labeling::extended(
  dmin = min(edPov$ed),
  dmax = max(edPov$ed),
  m = 5
)),
xaxis.labels = as.list(labeling::extended(
  dmin = min(edPov$ed),
  dmax = max(edPov$ed),
  m = 5
)),
xaxis.title = "Percent"
),
list(
  5,
  header = "Percent Living Below\nPoverty Level",
  graph.bgcolor = "lightgray",
  right.margin = 0.25,
  left.margin = -0.6,
  point.size = 1.5,
  xaxis.ticks = as.list(labeling::extended(
    dmin = min(edPov$pov),
    dmax = max(edPov$pov),
    m = 4
)),
  xaxis.labels = as.list(labeling::extended(
    dmin = min(edPov$pov),
    dmax = max(edPov$pov),
    m = 4
)),
  xaxis.title = "Percent"
)
)

```

What remains to be done is a summary and interpretation of the final refined linked micromap plot shown in Figure 2.9. This plot shows dotplots of two statistical variables, the percentage of adults with four or more years of college (in the first statistical graphics column which is the fourth column overall) and the percentage living below poverty level (in the second statistical graphics column which is the fifth column overall) in the 50 U.S. states and Washington, D.C.

The percentage of adults with four or more years of college is used as the sorting variable for the rows in the plot – with highest percentages shown at the top and lowest percentages shown at the bottom. The map panel in the first column shows some noticeable, but not very strong spatial patterns. Highest percentages of adults with four or more years of college can be found in the northeastern states. In fact, eight of the top-10 states are located in the northeast, with Washington, D.C., having the highest percentage with almost 40%. When looking at the other maps above the median state (here, Arizona), additional eastern states, but also several western states can be seen. When looking at the two maps at the bottom

2.3 Example 1: A Linked Micromap Plot for the 50 U.S. States and Washington, D.C. 51

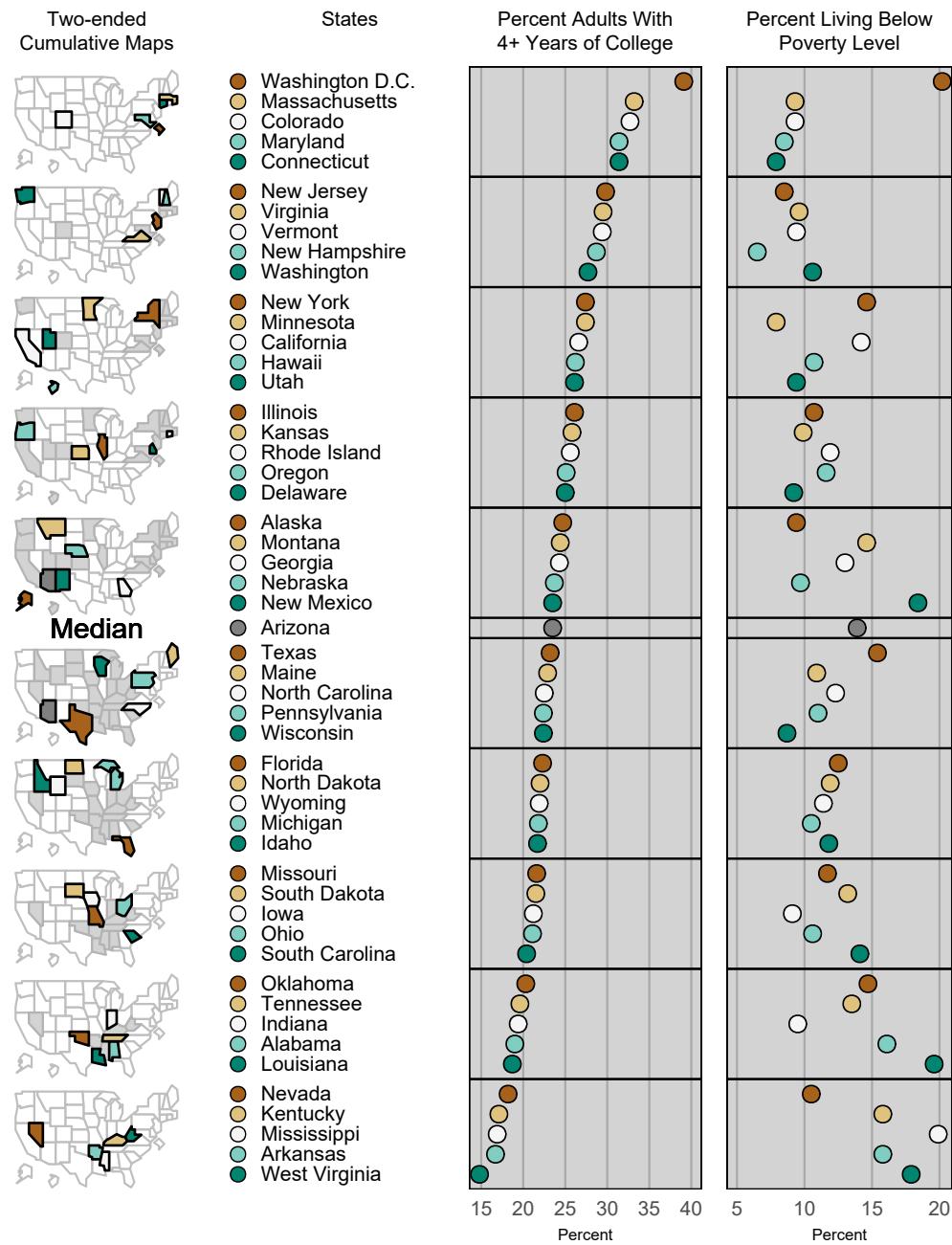


FIGURE 2.9 Sixth (and final) refined linked micromap plot, based on the *edPov* dataset. Main changes are related to labeling, the coloring of perceptual groups above and below the median row, and the column spacing.

of the plot, mostly southern states can be seen. West Virginia is the state with the lowest percentage of only about 15% of adults with four or more years of college. Overall, primarily southern and central states can be seen in the maps below the median state.

The percentage living below poverty level (in the fifth column overall) shows a different pattern. States with a high percentage of adults with four or more years of college have a low percentage living below poverty level. Visually, the dots in the fourth and fifth column diverge, forming some crude caret shape (resembling an upside down V-shape). Washington, D.C., is a major outlier as it has the highest percentage of adults with four or more years of college (almost 40%) but also the highest percentage living below poverty level (more than 20%). Another interesting state is New Mexico with an above median percentage of adults with four or more years of college, but with the fourth highest percentage living below poverty level (about 18%). As expected, the overall correlation between these two variables is negative, but given these two major outliers and several minor outliers (such as Indiana and Nevada that both have a relatively low percentage living below poverty level despite being among the bottom-10 states with respect to the percentage of adults with four or more years of college), the (negative) correlation is relatively weak. The correlation coefficient r is only -0.37.

2.4 Example 2: A Linked Micromap Plot for Watersheds in West Virginia

In this second linked micromap plot example created with the **micromap** R package, we work with the *WV_Watershed* dataset from the **micromapExtra** R package. We follow the four general steps outlined in Section 2.2 again. However, there are several differences compared to the first example in Section 2.3.

First, we work with external shapefiles (Environmental Systems Research Institute, Inc., 1998) that contain both the boundary for watersheds in West Virginia and the statistical data used in the following linked micromap plots, rather than having a separate dataset for the statistical data. In general, shapefiles are a collection of related files with the same prefix that contain the geography and attributes (i.e., data) of geographically referenced spatial features. Shapefiles consist of at least three files: a main file that stores the feature geometry (with suffix `.shp`), an index file that stores the index of the feature geometry (with suffix `.shx`), and a dBASE table that contains the attribute information of the spatial features (with suffix `.dbf`). Additional files may be included in a shapefile. Here, for the *WV_Watershed* dataset, four additional files are provided: a file with suffix `.prj` that contains the spatial coordinate system information (i.e., the projection), two files with suffix `.sbn` and `.sbx` that store the spatial index of the features, and a file with suffix `.xml` that contains metadata for the shapefile. Additional suffixes may be used for other shapefiles as outlined in Environmental Systems Research Institute, Inc. (2016). For use in linked micromap plots that are created with the **micromap** R package, external shapefiles can be handled in two ways.

Option (i) is to read in the external shapefile as a `SpatialPolygonsDataFrame`, and then split it into the geographic information and the statistical data component, use a modified statistical data component, or use a statistical data component from a different source, in particular if the shapefile does not contain any statistical data. Numerous R packages, such as **raster** (Hijmans, 2022a), **sf** (Pebesma, 2022), **shapefiles** (Stabler, 2022), and **terra**

(Hijmans, 2022b), support the use of shapefiles in R. From these R packages, only the **raster** R package directly creates a `SpatialPolygonsDataFrame`. For functions from the other R packages, an additional transformation step would be needed. Therefore, we use the `raster::shapefile()` function here to read in the shapefile.

Option (ii) is to read in the external shapefile in a simple features format (Open Geospatial Consortium, 2022) via the `sf::st_read()` function from the **sf** R package (Pebesma, 2022), instead of creating a `SpatialPolygonsDataFrame`. It is worthwhile to mention that geography and attributes that are stored in a simple features format no longer have to be split when used in linked micromap plots. Rather, the **micromap** R package can handle them directly from the simple features object.

Overall, options (i) and (ii) both will eventually result in the same final linked micromap plot. Option (i) may be preferred if the external shapefile does not contain the statistical data or if some considerable modifications have to be made to the statistical data prior to creating the linked micromap plot. Not all R package can handle simple features objects so that a split into a geographic component and a statistical data component may be necessary anyway for some advanced processing of the statistical data. Option (ii) may be preferred if the statistical data from the external shapefile can be used almost as provided in the external shapefile.

Moreover, in this second example we introduce two new statistical displays for the statistical graphics columns of the linked micromap plot: a boxplot and a dotplot with confidence bounds. Different arguments are used to fine-tune this linked micromap plot. Finally, we will demonstrate how to add an overall statistics (or criteria) line to the statistical graphics columns of the linked micromap plot.

2.4.1 Identifying and Geoprocessing of Spatial Boundary Data

The *WV_Watershed* dataset from the **micromapExtra** R package is stored in external shapefiles that can be read in as a `SpatialPolygonsDataFrame` via the `raster::shapefile()` function from the **raster** R package (Hijmans, 2022a) or via the `sf::st_read()` function from the **sf** R package (Pebesma, 2022) as discussed in the previous section.

The shapefile contains 25 aggregated watersheds and subbasins in West Virginia in the United States. These watersheds were introduced and discussed in more detail in McManus et al. (2016). In addition to the geographic information, this dataset also contains the statistical information for the linked micromap plots created in this section.

We first demonstrate the steps necessary for option (i). Similar to the first example in Section 2.3.1, we verify that this object (once read into R) indeed is a `SpatialPolygonsDataFrame`, then view some of the data in the `data` component of this object, and finally plot it, as shown in Figure 2.10. The last step is the extraction of the statistical data from this `SpatialPolygonsDataFrame` into a regular data frame. This can be done by accessing the data from the `@data` slot in the `SpatialPolygonsDataFrame`.

There are 41 variables in the resulting data frame. In contrast to McManus et al. (2016), we will focus on the specific conductance variables in the following linked micromap plots.

Variable names beginning with an uppercase letter, e.g., `Cond_med`, `Cond_LCB95`, `Cond_UCB95`, are the population estimates, representing the median, the lower 95% confidence bound, and the upper 95% confidence bound of a variable (here, specific conductance) in each of the 25 watersheds, respectively. These will be used for the construction of dotplots with confidence bounds.

Variable names starting with a lowercase letter, e.g., `cond_min`, `condq1`, `cond_med1`, `cond_q3`, `cond_max`, are the descriptive statistics, representing the minimum, first quartile, median, third quartile, and maximum of a variable (here, specific conductance) in each of the 25 watersheds, respectively. These will be used for the construction of boxplots.

```
wv_watershed <- raster::shapefile(
  x = "data/WV_Watershed/RandomWatershed2_stats_smooth.shp",
  verbose = FALSE
)

class(wv_watershed)

## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"

head(wv_watershed@data, n = 2)

##   OBJECTID      Random_Wat OBJECTID_1 mgm_id      Subpopulat
## 1       1 Big Sandy/Lower Ohio       1       1 Big Sandy/Lower Ohio
## 2       2 Cacapon/Shenandoah Hardy       2       2 Cacapon/Shenandoah Hardy
##          Type_x Indicator_ Cond_stat Cond_N Cond_med StdError_x Cond_LCB95
## 1 Random_Watershed    CONDUCT     50Pct     15    204.3      <NA>    183.88
## 2 Random_Watershed    CONDUCT     50Pct     15    103.1      <NA>     79.41
##   Cond_UCB95      Type_y Indicator1 Wvsci_stat Wvsci_N Wvsci_med
## 1     224.4 Random_Watershed      WVSCI     50Pct     14     67.77
## 2     123.8 Random_Watershed      WVSCI     50Pct     13     81.08
##   StdError_y Wvsci_LCB9 Wvsci_UCB9 variable_x cond_n1 cond_min condq1 cond_med1
## 1      <NA>    59.17     74.78    CONDUCT     30      96    175.5    203.5
## 2      <NA>    74.66     86.62    CONDUCT     30      33     64.5    102.5
##   condq3 cond_max variable_y wvsi_n wvsci_min wvsciq1 wvsci_med1 wvsciq3
## 1    244.2     328      WVSCI     30     19.88    53.23     69.05    80.80
## 2    149.0     273      WVSCI     30     31.67    72.44     81.59    90.55
##   wvsci_max Shape_Leng Shape_Area cond_iqr wvsci_iqr MaxSimpTol MinSimpTol
## 1     98.81    362714 7.656e+08    68.75    27.57    10000    10000
## 2     99.00    326757 2.218e+09    84.50    18.12    10000    10000

par(mar = c(0, 0, 0, 0))
plot(wv_watershed)

wv_data <- wv_watershed@data

names(wv_data)

## [1] "OBJECTID"    "Random_Wat"  "OBJECTID_1"  "mgm_id"      "Subpopulat"
## [6] "Type_x"      "Indicator_"  "Cond_stat"   "Cond_N"      "Cond_med"
## [11] "StdError_x"  "Cond_LCB95"  "Cond_UCB95"  "Type_y"      "Indicator1"
## [16] "Wvsci_stat"  "Wvsci_N"     "Wvsci_med"   "StdError_y"  "Wvsci_LCB9"
## [21] "Wvsci_UCB9" "variable_x"  "cond_n1"     "cond_min"   "condq1"
```

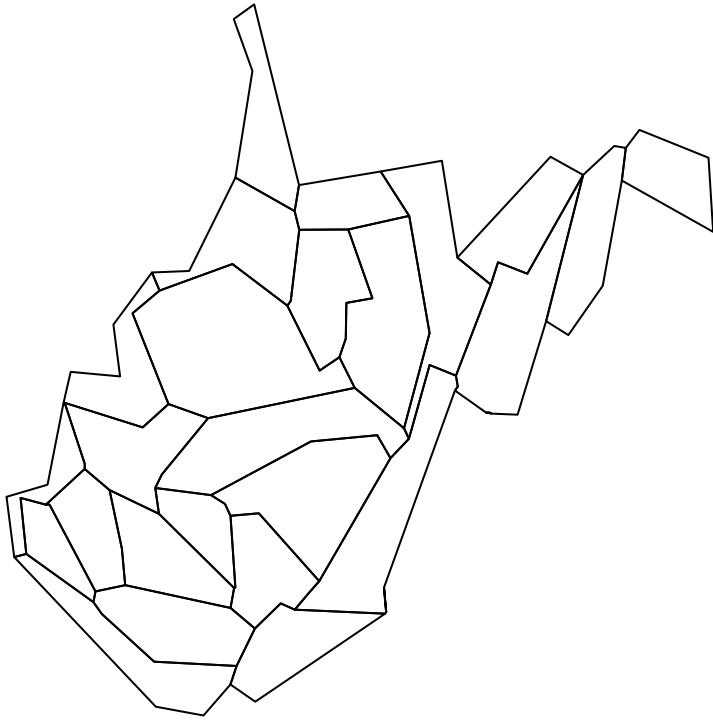


FIGURE 2.10 Map representation of the *WV_Watershed* spatial boundary dataset for the 25 watersheds in West Virginia that is used as the basis for the linked micromap plots for option (i).

```
## [26] "cond_med1"   "condq3"      "cond_max"     "variable_y"  "wvsi_n"
## [31] "wvsci_min"   "wvsciq1"     "wvsci_med1"   "wvsciq3"    "wvsci_max"
## [36] "Shape_Leng"   "Shape_Area"   "cond_iqr"     "wvsci_iqr"   "MaxSimpTol"
## [41] "MinSimpTol"
```

```
dim(wv_data)
```

```
## [1] 25 41
```

Alternatively, we demonstrate the steps necessary for option (ii). Similar to option (i), we verify that this object (once read into R) indeed is in the simple features format, then look at some of the data of this sf object, and finally plot it, as shown in Figure 2.11.

```
wv_watershed_sf <- sf::st_read(
  dsn = "data/WV_Watershed/RandomWatershed2_stats_smooth.shp",
  quiet = TRUE
)

class(wv_watershed_sf)

## [1] "sf"           "data.frame"
```

```
head(wv_watershed_sf, n = 2)

## Simple feature collection with 2 features and 41 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 1155000 ymin: 1758000 xmax: 1499000 ymax: 1986000
## Projected CRS: USA_Contiguous_Albers_Equal_Area_Conic_USGS_version
##   OBJECTID          Random_Wat OBJECTID_1 mgm_id           Subpopulat
## 1      1    Big Sandy/Lower Ohio      1      1    Big Sandy/Lower Ohio
## 2      2 Cacapon/Shenandoah Hardy      2      2 Cacapon/Shenandoah Hardy
##   Type_x Indicator_ Cond_stat Cond_N Cond_med StdError_x Cond_LCB95
## 1 Random_Watershed    CONDUCT     50Pct     15    204.3       <NA>    183.88
## 2 Random_Watershed    CONDUCT     50Pct     15    103.1       <NA>     79.41
##   Cond_UCB95        Type_y Indicator1 Wvsci_stat Wvsci_N Wvsci_med
## 1      224.4 Random_Watershed      WVSCI     50Pct     14      67.77
## 2      123.8 Random_Watershed      WVSCI     50Pct     13      81.08
##   StdError_y Wvsci_LCB9 Wvsci_UCB9 variable_x cond_n1 cond_min condq1 cond_med1
## 1      <NA>      59.17      74.78    CONDUCT      30      96    175.5     203.5
## 2      <NA>      74.66      86.62    CONDUCT      30      33     64.5     102.5
##   condq3 cond_max variable_y wvsi_n wvsci_min wvsciq1 wvsci_med1 wvsciq3
## 1     244.2      328      WVSCI      30      19.88     53.23      69.05     80.80
## 2     149.0      273      WVSCI      30      31.67     72.44      81.59     90.55
##   wvsci_max Shape_Leng Shape_Area cond_iqr wvsciq1 MaxSimpTol MinSimpTol
## 1     98.81      362714  7.656e+08     68.75     27.57     10000     10000
## 2     99.00      326757  2.218e+09     84.50     18.12     10000     10000
##   geometry
## 1 MULTIPOLYGON (((1188307 184...
## 2 MULTIPOLYGON (((1492137 198...

par(mar = c(0, 0, 0, 0))
plot(sf::st_geometry(wv_watershed_sf))
```

2.4.2 Linking Spatial Boundary Data and Statistical Data

This step is only required for option (i) and can be skipped entirely for option (ii). Similar to Section 2.3.2, the `SpatialPolygonsDataFrame` first is transformed into a regular data frame for use in a linked micromap plot via the `create_map_table()` function. The `Random_Wat` variable from the `data` component of the `SpatialPolygonsDataFrame` object can be used as an ID column here.

```
wv_polys_table <- create_map_table(
  tmp.map = wv_watershed,
  IDcolumn = "Random_Wat"
)

head(wv_polys_table)

##           ID region poly coordsx coordsy hole plotorder plug
## 1 Big Sandy/Lower Ohio      1     1 1188307 1841538      0       1     0
```

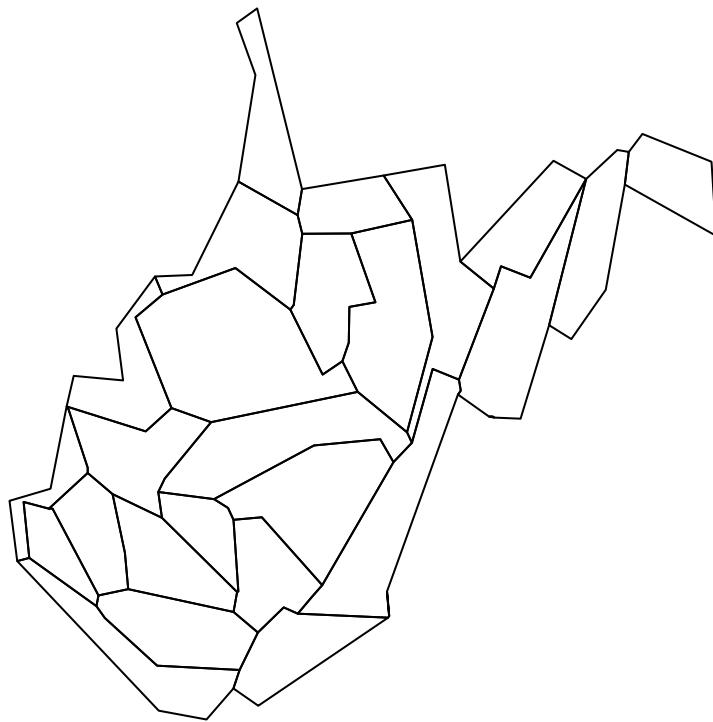


FIGURE 2.11 Map representation of the *WV_Watershed* spatial boundary dataset for the 25 watersheds in West Virginia that is used as the basis for the linked micromap plots for option (ii). It is identical to the one for option (i) shown in Figure 2.10.

```

## 2 Big Sandy/Lower Ohio      1     1 1198789 1809623      0      1      0
## 3 Big Sandy/Lower Ohio      1     1 1198780 1806726      0      1      0
## 4 Big Sandy/Lower Ohio      1     1 1177282 1786943      0      1      0
## 5 Big Sandy/Lower Ohio      1     1 1163277 1790675      0      1      0
## 6 Big Sandy/Lower Ohio      1     1 1166429 1759662      0      1      0

class(wv_polys_table)

## [1] "data.frame"

dim(wv_polys_table)

## [1] 238    8

names(wv_polys_table)

## [1] "ID"        "region"    "poly"       "coordsx"   "coordsy"   "hole"
## [7] "plotorder" "plug"

```

The resulting *wv_polys_table* is a regular data frame that will be used for creating a linked

micromap plot in the next step. `wv_polys_table` consists of 238 rows. This implies that the map is based on 238 line segments.

We have already extracted the statistical data into the `wv_data` data frame. Same as for the spatial component, the `Random_Wat` variable serves as the linking variable between the two data frames. As in Section 2.3.2, we want to verify that there is indeed at least one matching ID in the `wv_polys_table` data frame for each ID in the `wv_data` data frame. In fact, everything is matching here.

```
all(sort(wv_data$Random_Wat) == sort(unique(wv_polys_table$ID)))
## [1] TRUE
```

2.4.3 Creating a Draft Linked Micromap Plot

We can now create a minimal draft linked micromap plot using the `mmplot()` function, based on the previously created `wv_polys_table` and `wv_data` data frames for option (i). Similar to our first example in Section 2.3.3, we only provide seven required arguments in our function call. None of these arguments have a default setting. For all other arguments of this function, the default settings will be used here. The statistical data (`wv_data`) is assigned to the `stat.data` argument and the spatial boundary data (in the `wv_polys_table` data frame) is assigned to the `map.data` argument.

The `map.link` argument is needed to tell the function how to link the statistical data and the spatial boundary data. A vector with the names of the two linking variables identified in the previous step is needed for this argument. As explained in Section 2.3.3, the first variable name (`Random_Wat`) must come from the statistical data frame (`wv_data`) and the second variable name (`ID`) must come from the data frame that contains the spatial boundary information (`wv_polys_table`). Changing the order of these two variable names typically results in an error.

As discussed in Section 2.3.3, the `panel.types` and `panel.data` arguments are closely related. Even though we want to ultimately display boxplots and dotplots with confidence bounds in the statistical graphics columns of the linked micromap plot, it often is prudent to start with simple dotplots. Therefore, we start with these initial settings for the `panel.types` vector: We have the `dot_legend` in the first column, `labels` in the second column, two columns with statistical data represented as dotplots (`dot`) in the third and fourth columns, and the micromaps (`map`) in the final fifth column. This is matched with a list of data that is used for each of the five columns. Here, we use `Random_Wat` for the second column and `cond_med1` and `cond_med` for the third and fourth columns. No further data has to be used for the first and fifth columns, so `NA` is assigned here.

Two more arguments have to be specified: `ord.by` specifies the sorting variable of the rows in the linked micromap plot. Here, `cond_med1` from the `wv_data` argument is used. The sorting of the rows goes from smallest (at the top) to largest (at the bottom) median observed specific conductance in a certain subbasin. Finally, `grouping` specifies the number of rows in each of the perceptual groups (here 5 rows). This implies that the 25 subbasins will be split into five perceptual groups showing five subbasins each. This is the recommended Partitioning 1 from Table 1.1 in Chapter 1.

The resulting draft linked micromap plot is shown in Figure 2.12. We abstain from interpreting this figure at this stage, but we will provide some helpful interpretation once we have created the final refined version of this figure in the next step.

```
mmpplot(
  stat.data = wv_data,
  map.data = wv_polys_table,
  map.link = c("Random_Wat", "ID"),
  panel.types = c("dot_legend", "labels", "dot", "dot", "map"),
  panel.data = list(NA, "Random_Wat", "cond_med1", "Cond_med", NA),
  ord.by = "cond_med1",
  grouping = 5
)
```

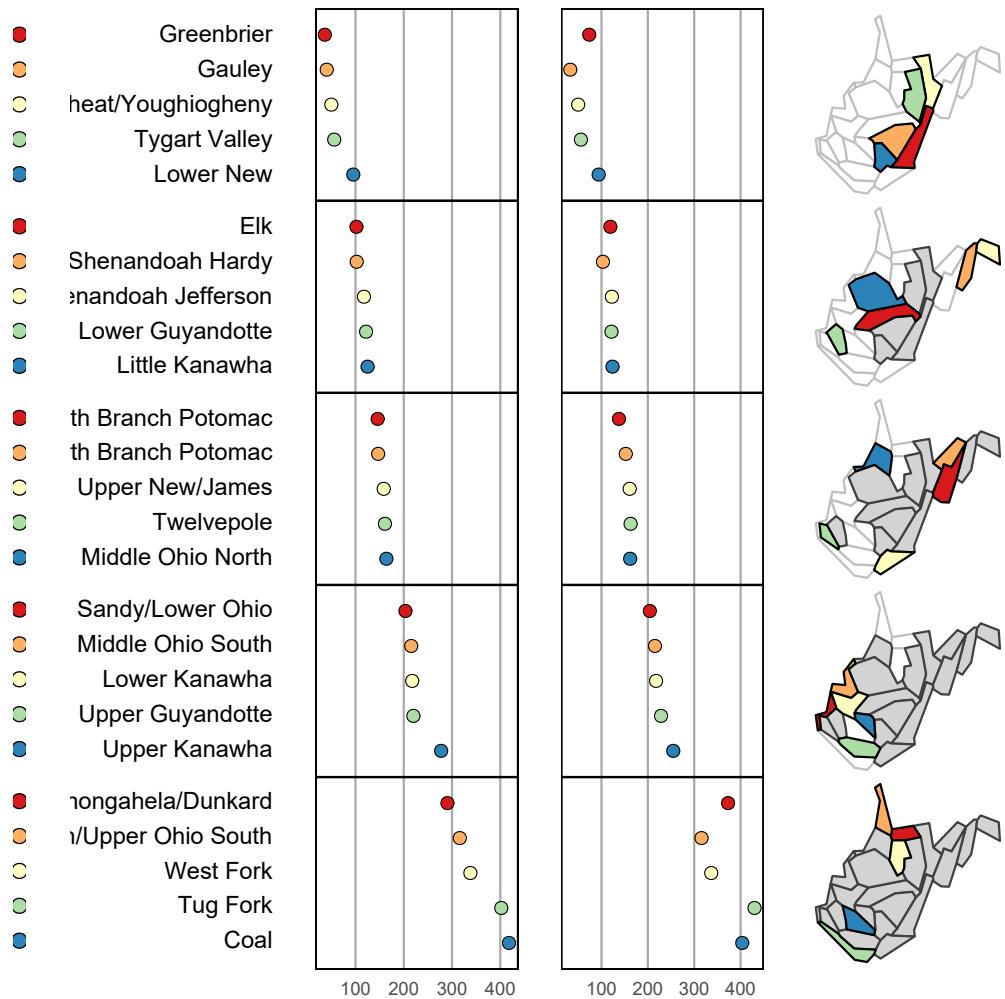


FIGURE 2.12 Draft linked micromap plot for option (i), based on the previously created `wv_data` and `wv_polys_table` data frames for the West Virginia watershed dataset.

For option (ii), the same draft linked micromap plot can be created directly from the `wv_watershed_sf` object (see Figure 2.13). It is not necessary to call the `create_map_table()` function first. Also, the `stat.data` and `map.link` arguments are not needed as the

`wv_watershed_sf` object contains all relevant geographic and statistical data. The linked micromap plots in Figure 2.12, and Figure 2.13 are identical. Both draft linked micromap plots can be refined in a similar way as shown in the next section.

```
mmpplot(
  map.data = wv_watershed_sf,
  panel.types = c("dot_legend", "labels", "dot", "dot", "map"),
  panel.data = list(NA, "Random_Wat", "cond_med1", "Cond_med", NA),
  ord.by = "cond_med1",
  grouping = 5
)
```

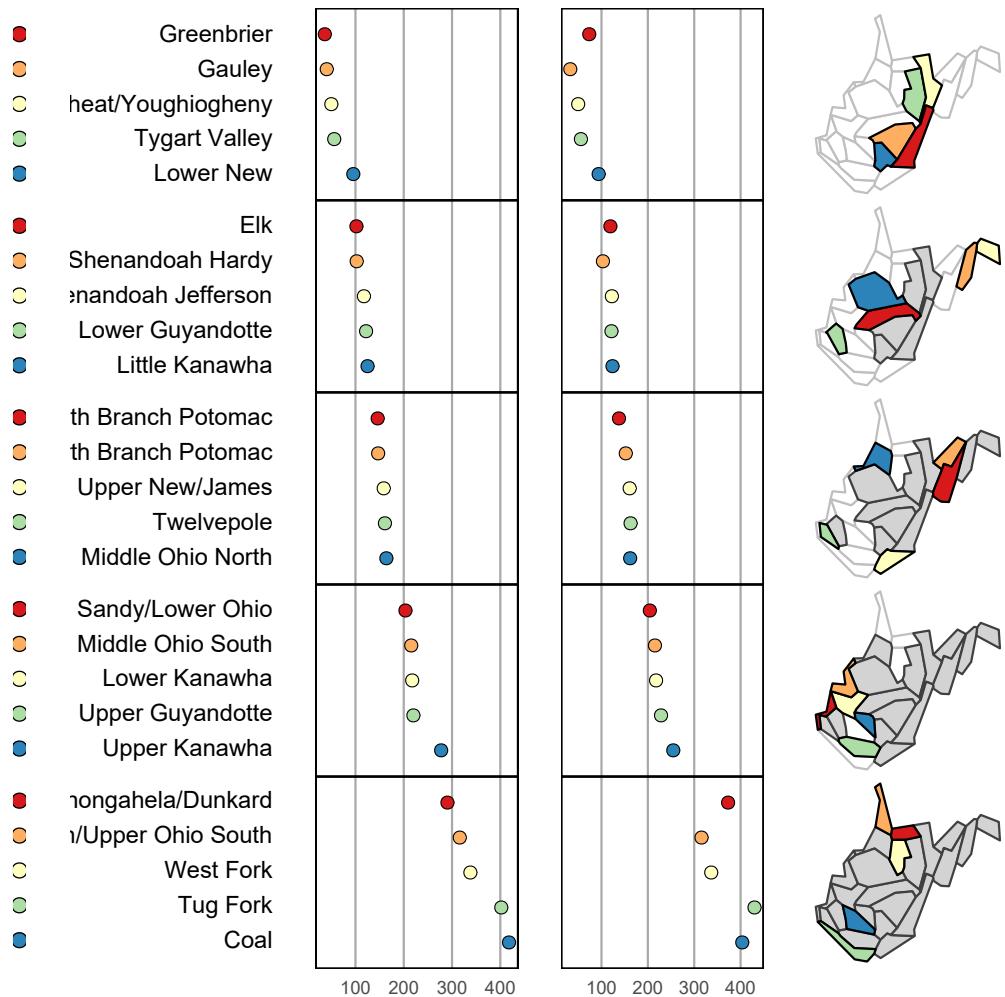


FIGURE 2.13 Draft linked micromap plot for option (ii), based on the West Virginia watershed dataset. In contrast to the previous figure, this figure has been created directly from the external shapefiles without using the `create_map_table()` function first.

2.4.4 Refining the Linked Micromap Plot

We continue with the draft linked micromap plot from the previous step and refine it in two main steps. As mentioned in Section 2.3.4, this refinement should only be started once functional R code has been obtained in the previous step.

A problem that did not occur in the first example in Section 2.3.4 is the length of many subregion names that considerably exceed the allocated space for this column as can be seen in Figure 2.12. Instead of using full terms, we shorten the names of the subbasins, in particular those seven names that combine smaller hydrologic unit codes (HUC) with larger neighboring HUCs (here HUC 8 at the subbasin level) in the West Virginia sampling frame. Overall, the following abbreviations are introduced and added as part of the new `short_name` variable to the `wv_data` data frame: L. (Lower), M. (Middle), U. (Upper), N. (North), S. (South), Br. (Branch), Dunk. (Dunkard), Mononga. (Monongahela), Shen. (Shenandoah), and Youg. (Youghiogheny). Shortening the subregion names often is recommended to prevent the names column from occupying too much space in the final linked micromap plot.

```
wv_data$short_name <- as.factor(wv_data$Random_Wat)

wv_watershed_short <- list(
  "S. Br. Potomac" = "South Branch Potomac",
  "N. Br. Potomac" = "North Branch Potomac",
  "Cacapon/Shen." = "Cacapon/Shenandoah Hardy",
  "Potomac/Shen." = "Potomac Direct Drains/Shenandoah Jefferson",
  "U. New/James" = "Upper New/James",
  "Tygart Valley" = "Tygart Valley",
  "West Fork" = "West Fork",
  "Mononga./Dunk." = "Monongahela/Dunkard",
  "Cheat/Youg." = "Cheat/Youghiogheny",
  "U. Ohio N./S." = "Upper Ohio North/Upper Ohio South",
  "M. Ohio N." = "Middle Ohio North",
  "M. Ohio S." = "Middle Ohio South",
  "Little Kanawha" = "Little Kanawha",
  "Greenbrier" = "Greenbrier",
  "L. New" = "Lower New",
  "Gauley" = "Gauley",
  "U. Kanawha" = "Upper Kanawha",
  "Elk" = "Elk",
  "L. Kanawha" = "Lower Kanawha",
  "Coal" = "Coal",
  "U. Guyandotte" = "Upper Guyandotte",
  "L. Guyandotte" = "Lower Guyandotte",
  "Tug Fork" = "Tug Fork",
  "Big Sandy/L. Ohio" = "Big Sandy/Lower Ohio",
  "Twelvepole" = "Twelvepole"
)
levels(wv_data$short_name) <- wv_watershed_short
```

In the first refined linked micromap plot for option (i), shown in Figure 2.14, we make

use of these abbreviations via the newly added `short_name` variable of the `wv_data` data frame. Moreover, we introduce initial boxplots and dotplots with confidence bounds in the two statistical graphics columns. This is done via `box_summary` and `dot_cl` for the `panel.types` argument. A boxplot requires five components (minimum, first quartile, median, third quartile, and maximum) and a dotplot with confidence bounds requires three components (median, lower 95% confidence bound, and upper 95% confidence bound) for each subregion. These have to be passed on as lists to the `panel.data` argument list. Here, we make use of these components from the specific conductance variable, specifically using `list("cond_min", "condq1", "cond_med1", "condq3", "cond_max")` for the boxplots and `list("Cond_med", "Cond_LCB95", "Cond_UCB95")` for the dotplots with confidence bounds. It should be noted that these eight components have been precalculated for each subregion and are variables of the `wv_data` data frame. The `mmpplot()` function does not calculate these summary statistics by itself from a provided data frame.

```
mmpplot(
  stat.data = wv_data,
  map.data = wv_polys_table,
  map.link = c("Random_Wat", "ID"),
  panel.types = c("dot_legend", "labels", "box_summary", "dot_cl", "map"),
  panel.data = list(
    NA,
    "short_name",
    list("cond_min", "condq1", "cond_med1", "condq3", "cond_max"),
    list("Cond_med", "Cond_LCB95", "Cond_UCB95"),
    NA
  ),
  ord.by = "cond_med1",
  grouping = 5
)
```

In the second (and final) refined linked micromap plot for option (i), shown in Figure 2.15, we make similar changes as for the third through sixth refined linked micromap plots in Section 2.3.4. This includes a reverse sorting (`rev.ord = TRUE`) that places the largest observed median specific conductance at the top, a specific color scheme (`colors = RColorBrewer::brewer.pal(n = 5, name = "YlGnBu")[5:1]`), numerous changes to the sizes and spacing of the perceptual groups and columns, and the addition of titles, scales, and axis labels. We keep the maps on the right side in the fifth column. A square symbol (instead of a circular one) is used for the color-coded legend in the first column via `point.type = 15` in the first list of the `panel.att` list. The thickness of the boxes in the third column (i.e., the first statistical graphics column) is reduced via `graph.bar.size = 0.7` in the third list of the `panel.att` list. The dashed black line in the fourth column (i.e., the second statistical graphics column) is created via the settings `add.line = 129`, `add.line.col = "black"`, `add.line.typ = "dashed"` in the fourth list of the `panel.att` list. The value 129 microSiemens per cm ($\mu\text{S}/\text{cm}$) represents the estimated median specific conductance across all 4th order streams (or less). The Greek letter μ in the two axis titles is produced via its unicode, i.e., `\u03BC`.

```
mmpplot(
  stat.data = wv_data,
  map.data = wv_polys_table,
```

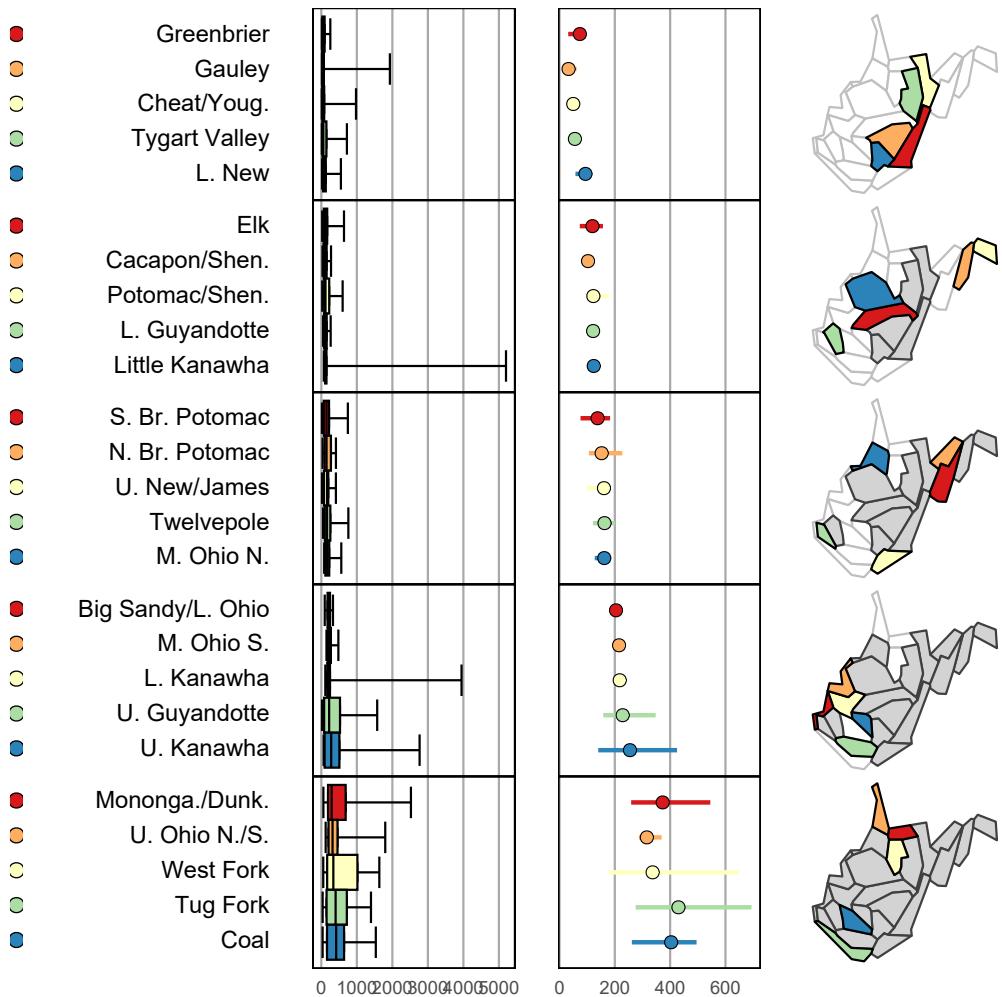


FIGURE 2.14 First refined linked micromap plot for option (i), based on the `wv_data` data frame. Main changes are the introduction of abbreviated subregion names and initial boxplots and dotplots with confidence bounds in the two statistical graphics columns.

```
map.link = c("Random_Wat", "ID"),
panel.types = c("dot_legend", "labels", "box_summary", "dot_cl", "map"),
panel.data = list(
  NA,
  "short_name",
  list("cond_min", "condq1", "cond_med1", "condq3", "cond_max"),
  list("Cond_med", "Cond_LCB95", "Cond_UCB95"),
  NA
),
ord.by = "cond_med1",
rev.ord = TRUE,
```

```
grouping = 5,
plot.pGrp.spacing = 1.2,
colors = RColorBrewer::brewer.pal(n = 5, name = "YlGnBu")[5:1],
panel.att = list(
  list(
    1,
    panel.width = 1.6,
    point.type = 15,
    point.size = 0.85,
    point.border = TRUE
  ),
  list(
    2,
    header = "WVDEP\nSubbasins",
    panel.width = 1.3,
    align = "left",
    text.size = 0.75
  ),
  list(
    3,
    header = "Observed specific conductance\n4th Order Streams or Less",
    graph.bgcolor = "lightgray",
    graph.bar.size = 0.7,
    xaxis.ticks = c(0, 1000, 2000, 3000, 4000, 5000),
    xaxis.labels = c(0, 1, 2, 3, 4, 5),
    xaxis.labels.size = 1.25,
    xaxis.title = "[1,000 \u03BCS/cm]",
    xaxis.title.size = 1.25,
    panel.width = 2.1,
    right.margin = 0,
    left.margin = -0.5
  ),
  list(
    4,
    header = "Estimated specific conductance\n4th Order Streams or Less",
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(0, 150, 300, 450, 600, 750),
    xaxis.labels = list(0, 150, 300, 450, 600, 750),
    xaxis.labels.size = 1.25,
    xaxis.title = "[\u03BCS/cm]",
    add.line = 129,
    add.line.col = "black",
    add.line.typ = "dashed",
    xaxis.title.size = 1.25,
    panel.width = 2.1,
    right.margin = 0.1,
    left.margin = -0.5
  ),
  list(
    5,
    header = "Predicted specific conductance\n4th Order Streams or Less",
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(0, 150, 300, 450, 600, 750),
    xaxis.labels = list(0, 150, 300, 450, 600, 750),
    xaxis.labels.size = 1.25,
    xaxis.title = "[\u03BCS/cm]",
    add.line = 129,
    add.line.col = "black",
    add.line.typ = "dashed",
    xaxis.title.size = 1.25,
    panel.width = 2.1,
    right.margin = 0.1,
    left.margin = -0.5
  )
)
```

```

      5,
      header = "Light Gray Means\nHighlighted Above",
      inactive.border.color = gray(0.7),
      inactive.border.size = 1.5,
      panel.width = 1.5
    )
  )
)

```

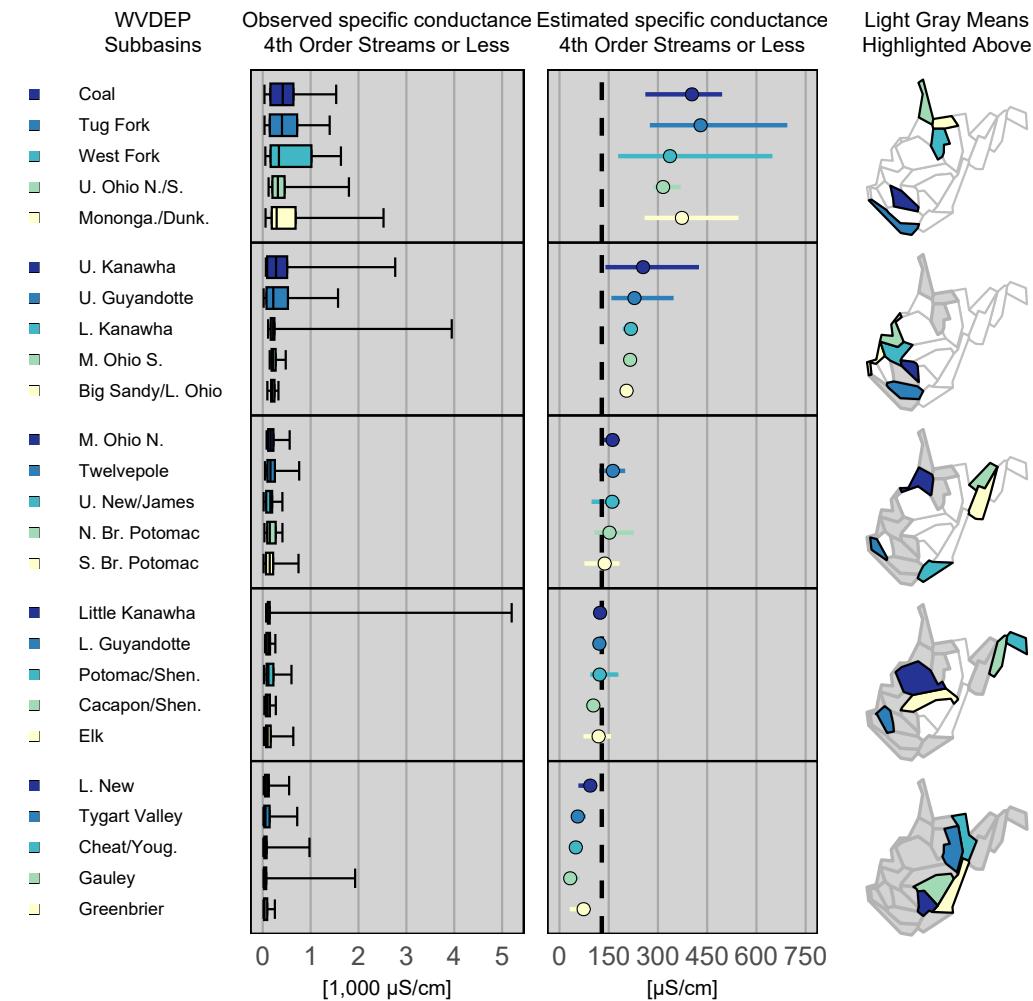


FIGURE 2.15 Second (and final) refined linked micromap plot for option (i), based on the `wv_data` data frame. Main changes are related to sorting, coloring, spacing, and labeling. An overall statistics (or criteria) line has been added to the second statistical graphics column.

Similarly to the modifications for option (i), we make changes of the subregion names of the `wv_watershed_sf` object (using the previously created list of shortened subregion names in `wv_watershed_short`) and immediately create the final refined linked micromap plot (shown

in Figure 2.16) for option (ii), rather than splitting this into two steps. Again, it is not necessary to call the `create_map_table()` function first. Also, the `stat.data` and `map.link` arguments are not needed as the `wv_watershed_sf` object contains all relevant geographic and statistical data. All other arguments for the refinement remain the same as used for Figure 2.15.

```
wv_watershed_sf$short_name <- as.factor(wv_watershed_sf$Random_Wat)

levels(wv_watershed_sf$short_name) <- wv_watershed_short

mmpplot(
  map.data = wv_watershed_sf,
  panel.types = c("dot_legend", "labels", "box_summary", "dot_cl", "map"),
  panel.data = list(
    NA,
    "short_name",
    list("cond_min", "condq1", "cond_med1", "condq3", "cond_max"),
    list("Cond_med", "Cond_LCB95", "Cond_UCB95"),
    NA
  ),
  ord.by = "cond_med1",
  rev.ord = TRUE,
  grouping = 5,
  plot.pGrp.spacing = 1.2,
  colors = RColorBrewer::brewer.pal(n = 5, name = "YlGnBu")[5:1],
  panel.att = list(
    list(
      1,
      panel.width = 1.6,
      point.type = 15,
      point.size = 0.85,
      point.border = TRUE
    ),
    list(
      2,
      header = "WVDEP\nSubbasins",
      panel.width = 1.3,
      align = "left",
      text.size = 0.75
    ),
    list(
      3,
      header = "Observed specific conductance\n4th Order Streams or Less",
      graph.bgcolor = "lightgray",
      graph.bar.size = 0.7,
      xaxis.ticks = c(0, 1000, 2000, 3000, 4000, 5000),
      xaxis.labels = c(0, 1, 2, 3, 4, 5),
      xaxis.labels.size = 1.25,
      xaxis.title = "[1,000 \u03bcS/cm]",
      xaxis.title.size = 1.25,
    )
  )
)
```

```
panel.width = 2.1,
right.margin = 0,
left.margin = -0.5
),
list(
  4,
  header = "Estimated specific conductance\\n4th Order Streams or Less",
  graph.bgcolor = "lightgray",
  xaxis.ticks = list(0, 150, 300, 450, 600, 750),
  xaxis.labels = list(0, 150, 300, 450, 600, 750),
  xaxis.labels.size = 1.25,
  xaxis.title = "[\u03BCS/cm]",
  add.line = 129,
  add.line.col = "black",
  add.line.typ = "dashed",
  xaxis.title.size = 1.25,
  panel.width = 2.1,
  right.margin = 0.1,
  left.margin = -0.5
),
list(
  5,
  header = "Light Gray Means\\nHighlighted Above",
  inactive.border.color = gray(0.7),
  inactive.border.size = 1.5,
  panel.width = 1.5
)
)
```

What remains to be done is a summary and interpretation of the final refined linked micromap plot shown in Figures 2.15 and 2.16. This plot is related to specific conductance variables in 25 aggregated watersheds and subbasins of 4th order streams (or less) in West Virginia in the United States.

The first statistical graphics column shows boxplots for the observed specific conductance for each of these watersheds. The second statistical graphics column shows dotplots with confidence bounds that are the population estimates of the specific conductance, representing the median, the lower 95% confidence bound, and the upper 95% confidence bound for each of these watersheds. The rows in the plot are sorted from highest (at the top) to lowest (at the bottom) median observed specific conductance in a certain subbasin.

Each consecutive map shows which subregions have been colored previously in the map(s) above. There is a weak spatial pattern with the highest median observed specific conductance in the northern and western subbasins of West Virginia. There is a notable spatial cluster in the central-eastern subbasins of West Virginia with the five lowest median observed specific conductance values shown in the perceptual group at the bottom of the plot.

The second statistical graphics column reveals that there is a remarkable relationship between the widths of the confidence bounds and the estimated specific conductance. The

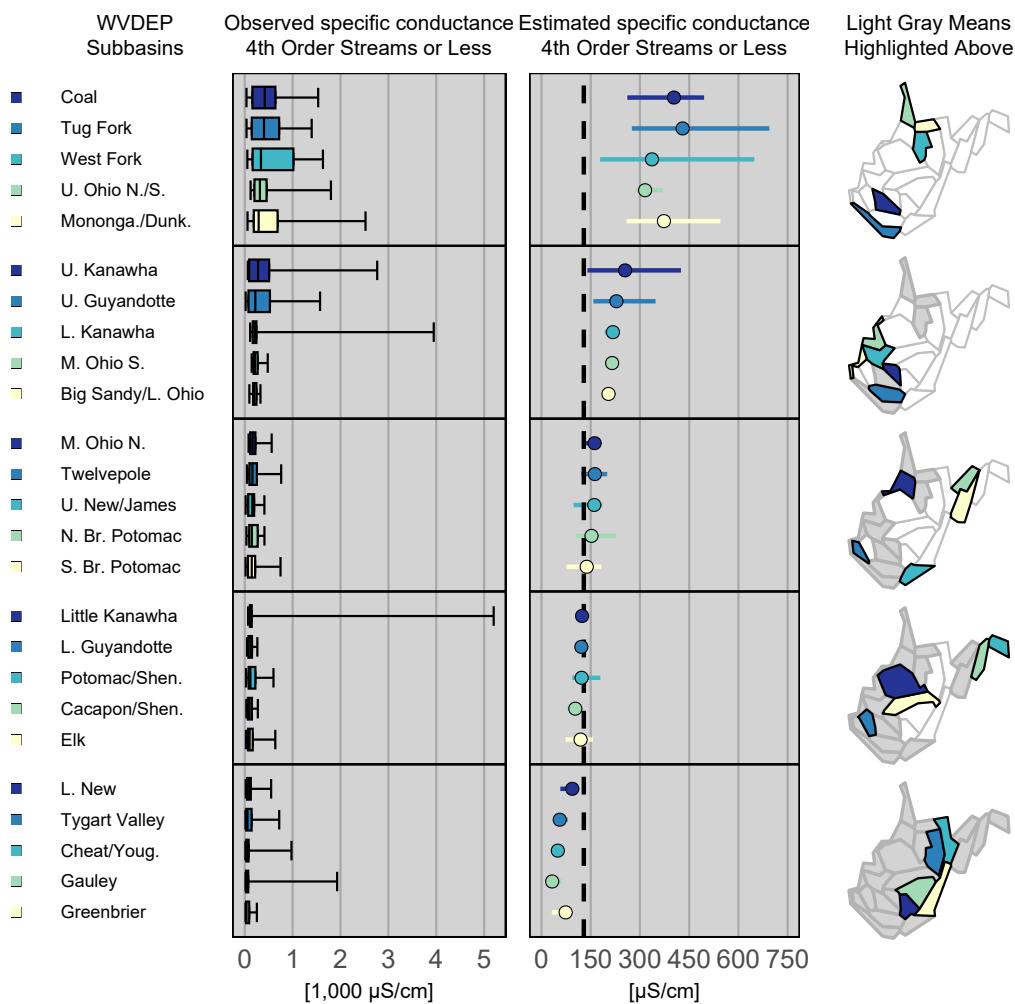


FIGURE 2.16 Second (and final) refined linked micromap plot for option (ii), based on the West Virginia watershed dataset. In contrast to the previous figure, this figure has been created directly from the external shapefiles without using the `create_map_table()` function first. It is identical to the one for option (i) shown in Figure 2.15.

higher the estimated specific conductance, the wider the confidence bound in general. The value 129 $\mu\text{S}/\text{cm}$, representing the estimated median specific conductance across all 4th order streams (or less), has been overlaid on this column.

Such a pattern cannot be noticed in the boxplots in the first statistical graphics column. Here, some extreme outliers dominate, e.g., observed specific conductance of about 5,000 $\mu\text{S}/\text{cm}$ in the Little Kanawha subbasin and of about 4,000 $\mu\text{S}/\text{cm}$ in the Lower (L.) Kanawha subbasin.

While this is hard to visually detect due to these extreme outliers, there is a considerable positive correlation between the observed median specific conductance and the estimated median specific conductance. Not surprisingly, the correlation coefficient r is 0.98.

2.5 Summary and Further Reading

In this chapter, we have discussed the four main steps that are required to create linked micromap plots with the **micromap** R package (Payton & Olsen, 2024). This R package is one of two major R packages for the construction of linked micromap plots. We will encounter it in several of the following chapters again. When designing linked micromap plots in those chapters, the authors typically have followed the same main steps. However, in most cases, only the final refined linked micromap plot will be shown in the following chapters. While most of those examples will follow option (i) and make use of the `create_map_table()` function as outlined in Section 2.3, the example shown in Figure 13.5 in Chapter 13 follows option (ii) and directly creates a linked micromap plot from a simple features object. Additional details and examples for both options can also be found in Section 2.4 and in the vignette (Payton et al., 2024) of the **micromap** R package.

In Chapter 4, we will see how to modify external shapefiles that contain too many edges and/or contain subregions that are too large, too small, or located too far away from the main geographic area — and then construct linked micromap plots that are based on the modified shapefiles. In Chapter 6, we will discuss how to display different types of statistical graphics in the statistical graphics columns in addition to dotplots, boxplots, and dotplots with confidence bounds that were introduced in this chapter.

Chapter 8 discusses linked micromap plots that are based on point locations, rather than on areal data as in this chapter. In Chapter 9, we will see how to construct linked micromap plots in a web-based environment. Finally, in Chapter 13, we will find applications of linked micromap plots in the environmental field. We will see the example with the watersheds in West Virginia, first introduced in Section 2.4, again in that chapter. These three chapters primarily make use of the **micromap** R package although the **micromapST** R package (Pearson Jr. & Carr, 2024) could be used in a similar way.

We do not want to forget that a user of linked micromap plots has a choice which R package to use for their creation. Chapter 3 introduces the **micromapST** R package (Pearson Jr. & Carr, 2024). Chapters 5 and 7 resemble Chapters 4 and 6 and discuss how to process external shapefiles and display different types of statistical graphics with the **micromapST** R package. Each of the two R packages for linked micromap plots has its own strengths and weaknesses. The **micromapST** R package has a larger number of built-in types of statistical graphics for the statistical graphics columns while the **micromap** R package allows (at least in theory) to add any meaningful type of statistical graphics that is not currently part of this R package. Similarly, the **micromapST** R package supports a larger number of built-in geographic regions while the **micromap** R package makes it easy to quickly construct linked micromap plots whenever external shapefiles are available. Such an option also exists for the **micromapST** R package, but it takes additional work to process external shapefiles. Ultimately, it is up to the user of linked micromap plots to decide which of these two R packages to use.

We want to finish this chapter with some practical recommendations. When working with **R Markdown** (Xie et al., 2018; Xie et al., 2021) and/or the **bookdown** R package (Xie, 2022a), sizing the figures with the linked micromap plots for the United States as `fig.width = 7, fig.height = 9` works well. For linked micromap plots used in journal publications, using the arguments `plot.width = 7, plot.height = 9` often works well. For linked micromap plots used in poster and presentation slides, using the arguments `plot.width = 9, plot.height = 6` usually works better. Ultimately, the width is somewhat determined by the

number of statistical graphics columns. A linked micromap plot with only one statistical graphics column has to be less wide than one with three or four statistical graphics columns. Similarly, the height is somewhat determined by the number of perceptual groups. A linked micromap plot with only two or three perceptual groups has to be less tall than one with nine or ten perceptual groups.

Some additional published examples of linked micromap plots created with the **micromap** R package are related to pesticides in Florida Department of Environmental Protection (FDEP) drainage basins (Silvanima et al., 2018), to variables from the Virginia Stream Condition Index (VSCI) and the Virginia Coastal Plain Macroinvertebrate Index (VCPMI) (Virginia Department of Environmental Quality, 2020), and to housing and immigration variables from the subboroughs in New York City (Medri Cobos, 2021) — all within the United States. International examples are related to population data in selected countries from South America (Symanzik et al., 2014), to suicide data by prefecture in Japan (Kubota et al., 2015), and to the spatial distributions of religions in China (Symanzik et al., 2016). Some of these examples needed additional modifications of external shapefiles as discussed in Chapter 4.

References

- Brewer, C. A., Hatchard, G. W., & Harrower, M. A. (2003). ColorBrewer in Print: A Catalog of Color Schemes for Maps [<https://doi.org/10.1559/152304003100010929>]. *Cartography and Geographic Information Science*, 30(1), 5–32.
- Carr, D. B., Bell, B. S., Pickle, L. W., Zhang, Y., & Li, Y. (2003). The State Cancer Profiles Web Site and Extensions of Linked Micromap Plots and Conditioned Choropleth Map Plots [<https://dl.acm.org/doi/10.5555/1123196.1123307>]. *Proceedings of the Third National Conference on Digital Government Research* (pp. 269–273). Digital Government Research Center (DGRC).
- Carr, D. B., Chen, J., Bell, B. S., Pickle, L. W., & Zhang, Y. (2002). Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps [<https://dl.acm.org/doi/10.5555/1123098.1123147>]. *Proceedings of the Second National Conference on Digital Government Research* (pp. 61–67). Digital Government Research Center (DGRC).
- Carr, D. B., Olsen, A. R., Courbois, J.-Y. P., Pierson, S. M., & Carr, D. A. (1998). Linked Micromap Plots: Named and Described. *Statistical Computing and Statistical Graphics Newsletter*, 9(1), 24–32.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (1998). Boxplot Variations in a Spatial Context: An Omernik Ecoregion and Weather Example. *Statistical Computing and Statistical Graphics Newsletter*, 9(2), 4–13.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (2000). Using Linked Micromap Plots to Characterize Omernik Ecoregions [<https://doi.org/10.1023/A:1009828700017>]. *Data Mining and Knowledge Discovery*, 4(1), 43–67.
- Carr, D. B., & Pearson Jr., J. B. (2013). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.

- Environmental Systems Research Institute, Inc. (1998). ESRI Shapefile Technical Description [White Paper, <https://www.esri.com/Library/Whitepapers/Pdfs/Shapefile.pdf>].
- Environmental Systems Research Institute, Inc. (2016). ArcGIS for Desktop: Shapefile File Extensions [Web Page, <https://desktop.arcgis.com/en/arcmap/10.3/manage-data/shapefiles/shapefile-file-extensions.htm> (accessed August 31, 2022)].
- Harrower, M. A., & Brewer, C. A. (2003). ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps [<https://doi.org/10.1179/000870403235002042>]. *The Cartographic Journal*, 40(1), 27–37.
- Hijmans, R. J. (2022a). *raster: Geographic Data Analysis and Modeling* [R package version 3.6–3 (<http://CRAN.R-project.org/package=raster>)].
- Hijmans, R. J. (2022b). *terra: Spatial Data Analysis* [R package version 1.6–17 (<http://CRAN.R-project.org/package=terra>)].
- Iannone, R. (2022). *DiagrammeR: Graph/Network Visualization* [R package version 1.0.9 (<http://CRAN.R-project.org/package=DiagrammeR>)].
- Kubota, T., Iizuka, M., & Tsubaki, H. (2015). Visualization of Spatial and Paneled Data for Reason-Specified Suicide Data by Prefecture in Japan [<https://2015.isiproceedings.org/Files/CPS415-P1-S.pdf>]. *Proceedings of the 60th World Statistics Congress of the International Statistical Institute, Rio de Janeiro, Brazil, 26-31 July 2015*. International Statistical Institute, The Hague, The Netherlands.
- McManus, M. G., Pond, G. J., Reynolds, L., & Griffith, M. B. (2016). Multivariate Condition Assessment of Watersheds with Linked Micromaps [<https://doi.org/10.1111/1752-1688.12399>]. *Journal of the American Water Resources Association*, 52(2), 494–507.
- Medri Cobos, J. (2021). *Housing Variables and Immigration: An Exploratory and Predictive Data Analysis in New York City* (Master's thesis) [<https://doi.org/10.26076/dd0e-699c>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Neuwirth, E. (2022). *RColorBrewer: ColorBrewer Palettes* [R package version 1.1-3 (<http://CRAN.R-project.org/package=RColorBrewer>)].
- Olsen, A. R., Carr, D. B., Courbois, J.-Y. P., & Pierson, S. M. (1996). Presentation of Data in Linked Attribute and Geographic Space. *1996 Abstracts, Joint Statistical Meetings, Chicago, Illinois* (p. 271). American Statistical Association, Alexandria, VA.
- Open Geospatial Consortium. (2022). Simple Feature Access - Part 1: Common Architecture [Web Page, <https://www.ogc.org/standards/sfa> (accessed August 31, 2022)].
- Payton, Q. C., Beck, M., Weber, M., McManus, M., Olsen, A. R., & Kincaid, T. (2024). *Vignette: Linked Micromaps* [R package version 1.9.10 (https://cran.r-project.org/web/packages/micromap/vignettes/Introduction_Guide.pdf)].
- Payton, Q. C., & Olsen, A. R. (2012). *micromap: Linked Micromap Plots* [R package version 1.5 (<http://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2018). *micromap: Linked Micromap Plots* [R package version 1.9.3 (<http://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2024). *micromap: Linked Micromap Plots* [R package version 1.9.10 (<http://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., Weber, M. H., McManus, M. G., Kincaid, T. M., & Olsen, A. R. (2013). Linked Micromaps: Statistical Summaries in a Spatial Context Using the micromap R Package [<https://www.usiale.org/austin2013/presentation-details/4693>]. *Landscape Dynamics Along Environmental Gradients, 2013 Annual Symposium, April 14–18, 2013, Austin, TX*. US Regional Association of the International Association for Landscape Ecology (US–IALE).

- Payton, Q. C., Weber, M. H., McManus, M. G., & Olsen, A. R. (2012). Linked Micromaps: Statistical Summaries in a Spatial Context [https://acwi.gov/monitoring/conference/2012/posters/posters_2012_conference.pdf]. *Water: One Resource — Shared Effort — Common Future, 8th National Monitoring Conference, April 30–May 4, 2012, Portland, Oregon*. National Water Quality Monitoring Council.
- Pearson Jr., J. B., & Carr, D. B. (2024). *micromapST: Linked Micromap Plots for U. S. and Other Geographic Areas* [R package version 3.0.3 (<http://CRAN.R-project.org/package=micromapST>)].
- Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data [<https://doi.org/10.32614/RJ-2018-009>]. *The R Journal*, 10(1), 439–446.
- Pebesma, E. (2022). *sf: Simple Features for R* [R package version 1.0–8 (<http://CRAN.R-project.org/package=sf>)].
- Pebesma, E., & Bivand, R. S. (2022). *sp: Classes and Methods for Spatial Data* [R package version 1.5–0 (<http://CRAN.R-project.org/package=sp>)].
- Silvanima, J., Woeber, A., Sunderman-Barnes, S., Copeland, R., Sedlacek, C., & Seal, T. (2018). A Synoptic Survey of Select Wastewater-Tracer Compounds and the Pesticide Imidacloprid in Florida's Ambient Freshwaters [<https://doi.org/10.1007/s10661-018-6782-4>]. *Environmental Monitoring and Assessment*, 190(7), 435.
- Stabler, B. (2022). *shapefiles: Read and Write ESRI Shapefiles* [R package version 0.7.2 (<http://CRAN.R-project.org/package=shapefiles>)].
- Symanzik, J., Bao, S., Dai, X., Shui, M., & She, B. (2016). Recent Advancements in Geovisualization, with a Case Study on Chinese Religions [https://doi.org/10.1007/978-3-319-42571-9_8]. In Z. Jin, M. Liu, & X. Luo (Eds.), *New Developments in Statistical Modeling, Inference and Application: Selected Papers from the 2014 ICSA/KISS Joint Applied Statistics Symposium in Portland, OR* (pp. 151–166). Springer, Cham, Switzerland.
- Symanzik, J., & Carr, D. B. (2013). Linked Micromap Plots in R. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 213–218). Asian Regional Section of the IASC.
- Symanzik, J., Dai, X., Weber, M. H., Payton, Q., & McManus, M. G. (2014). Linked Micromap Plots for South America — General Design Considerations and Specific Adjustments [<https://doi.org/10.15446/rce.v37n2spe.47949>]. *Revista Colombiana de Estadística*, 37(2), 451–469.
- Talbot, J. (2020). *labeling: Axis Labeling* [R package version 0.4.2 (<http://CRAN.R-project.org/package=labeling>)].
- Virginia Department of Environmental Quality. (2020). Chapter 4.4 Freshwater Probabilistic Monitoring Results [<https://www.deq.virginia.gov/home/showpublisheddocument/2221/637436316328230000>]. *2020 305(b)/303(d) Water Quality Assessment Integrated Report, EPA Approved, December 9, 2020* (pp. 112–152). Virginia DEQ, Richmond, VA.
- Wang, X., Chen, J. X., Carr, D. B., Bell, B. S., & Pickle, L. W. (2002). Geographic Statistics Visualization: Web-based Linked Micromap Plots [<https://doi.org/10.1109/5992.998645>]. *Computing in Science & Engineering*, 4(3), 90–94.
- Xie, Y. (2022a). *bookdown: Authoring Books and Technical Documents with R Markdown* [R package version 0.29 (<http://CRAN.R-project.org/package=bookdown>)].
- Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R Markdown: The Definitive Guide* [<https://bookdown.org/yihui/rmarkdown/>]. CRC Press, Boca Raton, FL.
- Xie, Y., Dervieux, C., & Riederer, E. (2021). *R Markdown Cookbook* [<https://bookdown.org/yihui/rmarkdown-cookbook>]. CRC Press, Boca Raton, FL.

3

*Linked Micromap Plots via the **micromapST** R Package*

LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR., DANIEL B. CARR

Similar to Chapter 2, the **micromapST** R package (Carr & Pearson Jr., 2015), accessible at <https://cran.r-project.org/web/packages/micromapST/index.html>, will be introduced in this chapter. The reader will learn how to create a basic linked micromap plot via this R package.

Although the default parameter settings give publication-ready output, details will be provided on additional display options for U.S. states and other geospatial boundary files.

3.1 Introduction

Linked micromap plot designs can serve many purposes, including communicating to audiences in different application domains and identifying patterns useful in exploratory data analysis. Users ranging from the general public to sophisticated data analysts can understand these plots. The design removes the technical detail of plot creation from the viewer, allowing the data patterns to stand out visually.

The **micromapST** package simplifies creation of the micromap plot by incorporating default plot parameters that produce publication-ready output for most applications. Furthermore, its execution is quick because it plots the micromaps using fundamental R functions rather than calling another plotting package.

As noted in Chapter 1, the linked micromap plot design arose from Dan Carr's wish to make geospatial data available from many federal agencies more accessible to the public. The goal was primarily to communicate those statistics in a compact, easily-understood form. Up to that point, most federal statistical reports consisted of tables sorted alphabetically by the names of the areas represented. In a meeting with Tony Olsen at the Environmental Protection Agency (EPA) in the early 1990s, Olsen commented that he liked Dan's sorted row plot graphical design, but also wanted to see the geospatial context of the data. Dan's immediate response was the initial linked micromap design. Dan collaborated on specific geospatial graphical design questions throughout the 1990s with researchers at the Environmental Protection Agency, the Bureau of Labor Statistics, the National Center for Health Statistics and the National Cancer Institute, leading to refinements of the design.

A secondary goal of the micromap design was to serve as a tool for exploratory spatial data analyses. For example, static maps of death rates due to cancer (Mason et al., 1975; Pickle

et al., 1987, 1990) and other causes (Pickle et al., 1996) generated hypotheses about possible causes of the observed geospatial patterns in these rates that could subsequently be tested by epidemiologic studies. Linked micromaps provide the analyst a tool to explore visual associations between the values of the outcome variable, such as lung cancer rates, and possible risk factors for that outcome, such as smoking rates, income and education levels. Sorting on the outcome variable or a putative risk factor can reveal geospatial clusters of similar values in the small maps, identifying the most promising places for subsequent study. These patterns can also trigger further thoughts on other variables to examine. Linked micromap plots of larger geospatial units, such as U.S. states, can help analysts quickly identify broad geospatial patterns in the data and the more promising risk factors for future study. Subsequent micromap plots can drill down to smaller geographic units, such as counties, within larger units of interest.

Early implementations of the micromap design were done using S-plus software (Carr, Olsen, Courbois, et al., 1998; Carr, Olsen, Pierson, et al., 1998, 2000; Carr & Pierson, 1996; Carr, Wallin, et al., 2000). When the National Cancer Institute (NCI) became interested in including a web-based interactive version of linked micromaps on its proposed State Cancer Profiles website, Dan Carr worked with Sue Bell at NCI to finalize the design and functionality of the new graphical tool, which was then implemented in JAVA (Carr et al., 2003; Carr, Chen, et al., 2002). Since this JAVA applet was specifically for publicly-available cancer data and could not be changed from the NCI production website version, Carr led the efforts to convert the earlier S-plus software to R code so that refinements of the basic design could continue.

Later, Jim Pearson collaborated with Carr to create an R package, first released in 2013. The goal of this package development was to quickly display a visually appealing linked micromap plot from a list of user-provided parameters without requiring customizing code. The package structure used a table-driven description of the desired micromap. Filling in the table cells was analogous to filling in spreadsheet cells, a task so familiar to many users at that time. That is, the user only needed to know how to read in the data, then load and execute a package in R. Pearson generalized the original R code for ease of maintenance and added checks and meaningful messages for input parameter errors. Functionality of the package was expanded by adding many new glyphs and the ability to display a variable number of areas and maps. Because the initial implementation was for a fixed design to display U.S. states, and to distinguish this package from the EPA version (**micromap**), we named this package **micromapST**. This was an unfortunate choice, because users think that the package still only displays U.S. states although it has since been expanded to show data for sub-state cancer registries, selected states at the county level, several other countries and now has the ability to read in user-defined border files.

In this chapter we demonstrate how to use **micromapST** to create a basic linked micromap plot of U.S. state data. From the default output, we will show how to add or modify display options and then will illustrate how to plot other geographic boundary sets that are included in the package. In Chapter 7, we will demonstrate the various glyphs that are currently available in the package. In Chapter 5, we illustrate a new feature that enables users to supply their own boundary files and to create other geospatial displays.

3.2 Basic Plot Layout and Function Call

We start by producing the most basic linked micromap plot from **micromapST**. This just involves specifying the dataset and variables to use, the glyphs to use for each statistical column and the order of displayed columns. All other parameters are set to their defaults. In the next sections, we will add display options and change the perceptual grouping of areas, using one of the growing list of built-in boundary files as an example.

The first examples examine the growing numbers of U.S. state covid-19 cases from February 24, 2020, to April 14, 2022. The monthly cumulative case counts, both confirmed and presumptive, are from USAFacts (USA Facts, 2022). In order to compare states with widely varying populations, we divided each count by the state's 2019 population to obtain rates representing the numbers of cases per 1000 population. The same process was used to produce cumulative case rates for each U.S. county.

We wondered if the clear differences in the emergence and later surges of covid seen among the states were associated with differences in preventive behavior, such as social distancing and mask wearing. To illustrate how this question could be examined using linked micromap plots, we merged the cumulative case rate dataset with data on mask wearing collected from the Carnegie Mellon University's Delphi Group's Facebook survey (Salomon et al., 2021). The survey question in 2020 was "In the last 5 days, how often did you wear a mask in public?". We downloaded the percents by state who responded that they wore a mask most or all of the time in September, 2020. To illustrate, the merged input data for Alaska (AK) and Alabama (AL) are shown below:

```
# Input data files.
# Data Set # 1
st_covid_case_rates <- read.csv("data/STCaseRatesPer1000.csv",
  row.names = 2
)
head(st_covid_case_rates, n = 2L)

##      seqno PctWoreMask0920 PctMaskStderr PctMaskSample_size Feb_24_2020
## AK        1          79.44           2.66             230            0
## AL        2          84.73           1.14            1004            0
##   Mar_25_2020 Apr_24_2020 May_24_2020 Jun_23_2020 Jul_23_2020 Aug_22_2020
## AK        0.06         0.46           0.56           1.13           3.42           6.99
## AL        0.08         1.23           2.96           6.34          15.14          23.36
##   Sep_21_2020 Oct_21_2020 Nov_20_2020 Dec_20_2020 Jan_19_2021 Feb_18_2021
## AK       10.19        16.55          35.29          59.09          70.49          77.30
## AL       29.73        35.83          46.58          65.76          86.99          98.79
##   Mar_20_2021 Apr_19_2021 May_19_2021 Jun_18_2021 Jul_18_2021 Aug_17_2021
## AK       82.94        89.49          94.41          95.38          97.7           106.6
## AL      104.24       106.57          110.31          111.90          114.2           130.8
##   Sep_16_2021 Oct_16_2021 Nov_15_2021 Dec_15_2021 Jan_14_2022 Feb_13_2022
## AK      128.5         166.5          189.2          199.2          235.0          300.7
## AL      155.4         165.8          171.3          174.3          204.9          256.9
##   Mar_15_2022 Apr_14_2022
## AK       313.2         320.2
```

```
## AL      263.0      264.6
```

To create the linked micromap, the user first specifies what to display in a data frame called `panelDesc`. This data frame has a spreadsheet-like structure, with each column of the data frame defining a column on the micromap display and each row of the data frame defining a particular display parameter. For this simple example `panel_desc_covid1` is constructed with 3 columns and 3 rows. The `type` column specifies the content of the micromap from left to right with maps on the left, area i.d. next and dot plots encoding state values on the right (`type = c("map", "id", "dot")`). The `col1` column of the data frame specifies that `Nov_20_2020`, the cumulative number of covid cases on November 20, 2020, is the variable to use in the dot plot. We need to include `NA` as placeholders for the first and second columns because only the third column has data input. Although not required, we added a label for the dot plots (`lab1`). Similar to the `col1` specification, `lab1` includes an empty string ("") for the map and i.d. display columns, indicating that these do not have extra text.

After setting up the display format, we call the `micromapST` function within the package. For this simple example, we only need to specify the names of the relevant data frames: where the data to be plotted reside and the name of the plot description (`panel_desc_covid1`). We also add a title for clarity.

```
library(micromapST)

panel_desc_covid1 <- data.frame(
  type = c("map", "id", "dot"),
  col1 = c(NA, NA, "Nov_20_2020"),
  lab1 = c("", "", "Cases/1000")
)
panel_desc_covid1

##   type      col1      lab1
## 1 map      <NA>
## 2 id       <NA>
## 3 dot Nov_20_2020 Cases/1000

# default = alphabetic state name sort
micromapST(
  statsDFrame = st_covid_case_rates,
  panelDesc = panel_desc_covid1,
  title = c("US State Covid Cumulative Case Rates per 1000 Population, Nov. 20, 2020")
)

## rowNames value used is : ab
```

This is a nice-looking plot that would work well for the task of reading the case rates for particular states. Thin black horizontal lines delineate the perceptual groups of 5 states each and white vertical lines on the light gray background help guide the reader's eye up or down to the axis labels. The default grouping for 51 areas is 10 groups of 5 areas each with the median-valued area shown separately in the middle of the plot.

The default order of the states in a column is alphabetic by U.S. state postal codes. This order is not useful for identifying geospatial patterns in the data. Sorting the column of states by their data values, such as covid case rates on November 20, 2020, provides a

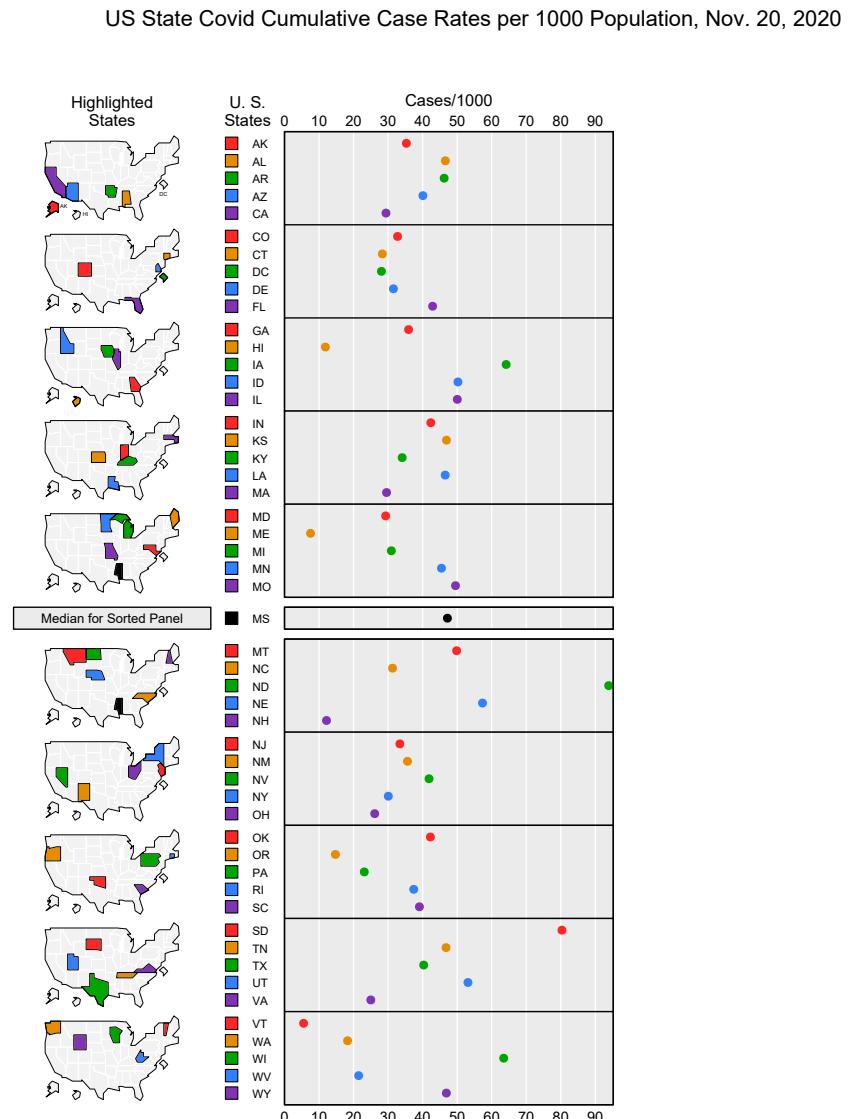


FIGURE 3.1 Cumulative number of covid-19 cases reported by U.S. states on Nov. 20, 2020.

more useful design. Since this variable is already included in the `panel_desc_covid1` data frame, the arguments `sortVar = "Nov_20_2020"` and `'ascend = FALSE` in the **micromapST** function call are the only additions needed to plot the states in a descending case rate order.

```
micromapST(
  statsDFrame = st_covid_case_rates,
  panelDesc = panel_desc_covid1,
  sortVar = "Nov_20_2020",
  ascend = FALSE,
  title = c("US State Covid Cumulative Case Rates per 1000 Population, Nov. 20, 2020")
)

## rowNames value used is : ab
```

Now we can see that North and South Dakota and nearby states had high case rates in the fall of 2020. Others noticed this spike in cases at that time and speculated that a huge motorcycle rally in Sturgis ND during August 2020 might have been a covid superspreader event, because so many people were crowded together and most attendees were reportedly strongly against mask mandates (Firestone et al., [2020](#)).

Let's see how our linked micromap plot could explore this hypothesis. Included in our dataset is the percent who reported that they usually wore a mask in public on a Facebook survey in September 2020 as described at the beginning of this section. We chose September because it is close to the date of the motorcycle rally but before the case spike had been reported. We can add this mask-wearing variable to the plot and look for a visual pattern suggestive of a correlation or a nonlinear functional relationship with the case rate column.

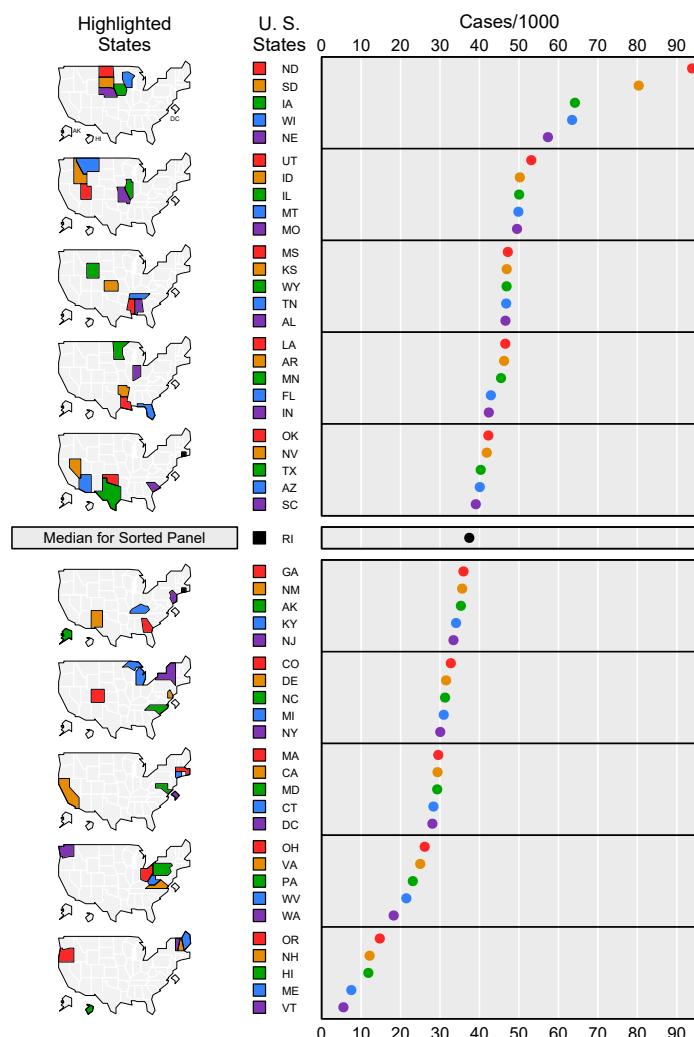
First, we define the plot (`panel_desc_covid2`) to include four columns, the fourth (rightmost) one being the mask variable `PctWoreMask0920`, which will also be shown as a dot. Then we call the plotting function, again specifying that the states be sorted in descending order by case count. Note that since the plot title has gotten long, we split the title into two character strings; the second string prints on a second title line.

```
panel_desc_covid2 <- data.frame(
  type = c("map", "id", "dot", "dot"),
  lab1 = c("", "", "Cases/1000", "% wore mask"),
  col1 = c(NA, NA, "Nov_20_2020", "PctWoreMask0920")
)

micromapST(
  statsDFrame = st_covid_case_rates,
  panelDesc = panel_desc_covid2,
  sortVar = "Nov_20_2020",
  ascend = FALSE,
  title = c(
    "US State Covid Cumulative Case Rates, Nov. 20, 2020",
    " and percent who usually wore a mask, Sep. 2020"
  )
)

## rowNames value used is : ab
```

US State Covid Cumulative Case Rates per 1000 Population, Nov. 20, 2020

**FIGURE 3.2** Cumulative number of covid-19 cases reported by U.S. states on Nov. 20, 2020.

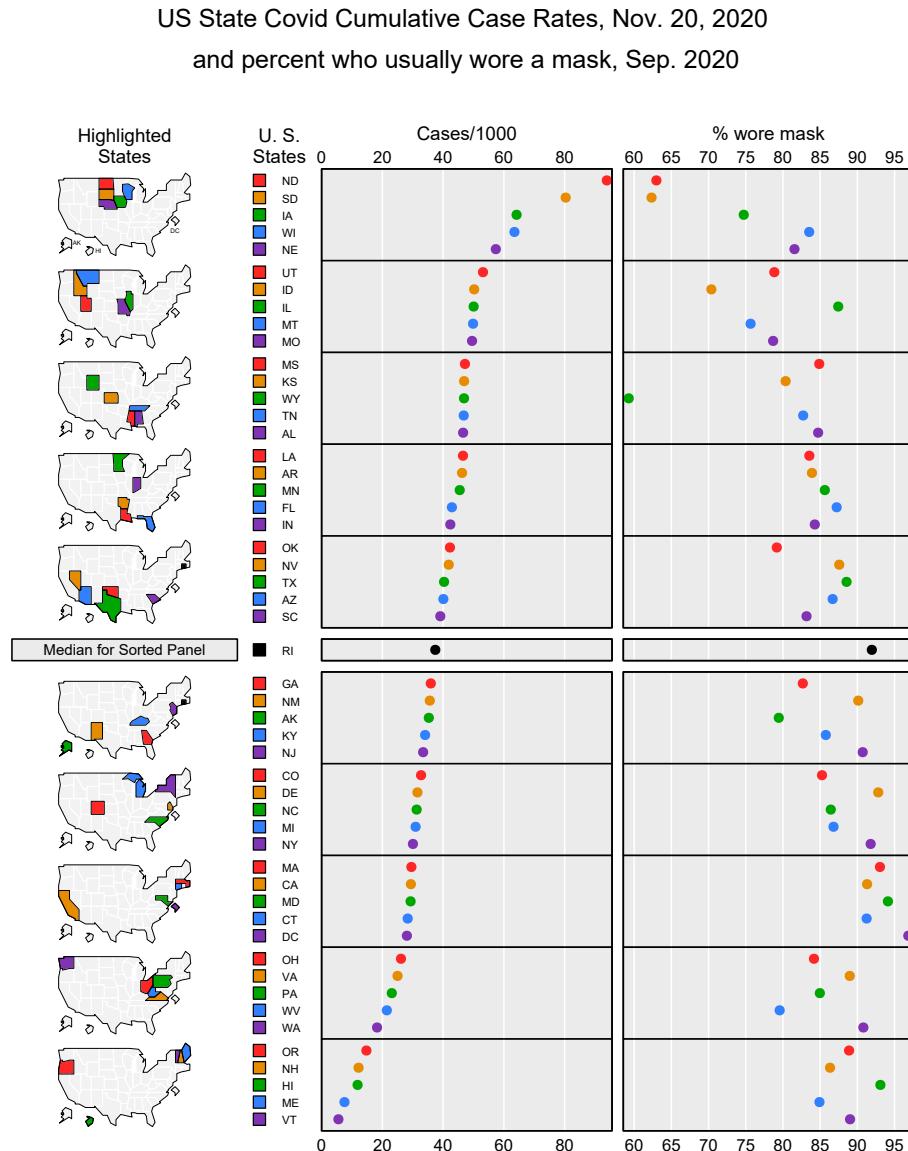


FIGURE 3.3 Cumulative number of covid-19 cases per 1000 by state on Nov. 20, 2020, and percent who usually wore a mask in Sep. 2020.

There appears to be a negative correlation between the two variables. For example, we can see that the Dakotas had very high covid totals by November 2020 and residents there reported relatively low mask wearing in September. This micromap supports the theory that the Sturgis motorcycle rally was a covid superspreader event, i.e., exposure in North Dakota in August led to covid infections soon after in surrounding states where most attendees came from. This example examines the influence of mask wearing on covid case rates. The states could be sorted instead by the mask-wearing variable to look for the reverse relationship. An epidemiologic study would be needed to confirm any apparent relationship between the two variables, but this example illustrates how linked micromaps can be used to quickly identify case clusters and explore suspected risk factors.

3.3 Display Options

In this section we illustrate display modifications that support communication and pattern identification. Using full state names can help those uncertain about state abbreviations. Enhancing labels support interpretation. Bringing sequences of featured states into the foreground using color fill and outline color can reveal larger geospatial patterns. Adding a vertical reference line, such as the U.S. rate, supports state comparisons against a common standard. Adding state rate standard error intervals shows which states have rates plausibly similar to the national rate and which do not.

Most design changes are made by changing values in the `panelDesc` data frame. For the next example, we modify the default display for the same data shown in Figure 3.3. Since there are four columns in the display, every parameter specification must have four values, one for each column, left to right. `NA` is used as a placeholder for each column where that parameter is not applicable. First, we change the simple dot glyph for the percent who wore a mask to include a standard error bar by changing `dot` to `dotse` in the definition of the plot columns (`type`). In Chapter 7 many more glyphs will be defined. Compared to the simple dot, this glyph needs the additional information of the length of the error bar, so a parameter (`col2`) is added to point to that value in the data set. Note that this extra value is only needed for the `dotse` column, the fourth column from left to right on the plot, so the first three positions in `col2` are shown as `NA`.

Another addition to `panelDesc` here is `refVals`, which defines a vertical dashed line in any or all of the graphical columns. In this example, we request a line to be drawn at 85% on the mask-wearing column, the reported percent wearing masks in the U.S. on that date. `refTexts` provides a label (`us %`) for the reference line legend.

Finally, we request cumulative map shading by changing `map` to `mapcum` in the column definitions (`type`). This option shades in light yellow any map areas that had been already displayed above. That is, from the top down, each map shows the active perceptual group's colored areas as well as the previous groups' areas shaded in yellow. This extra color fill pops the previously-displayed states into the foreground so that any growing geospatial cluster of sorted values is evident. This alleviates the problem of missing a cluster of similar-valued states that may fall into different, but adjacent, perceptual groups. The shading can be defined to accumulate from the top down (`mapcum`) or from the extremes to the middle (`maptail`). A third option (`mapmedian`) is a simple binary shading whereby areas that have values above the median are shaded in one color in the upper maps and areas in the lower maps with values below the median are shaded in another color.

In general, options for the display of the statistical data and glyphs are specified in `panelDesc` while options for the entire micromap display are specified in the function call to **micromapST**. In this example, the geographic identifier is full state names, rather than the default state postal codes used in the previous examples (`plotNames = "full"`).

```
panel_desc_covid2 <- data.frame(
  type = c("mapcum", "id", "dot", "dotse"),
  lab1 = c("", "", "Cases/1000", "% wore mask"),
  col1 = c(NA, NA, "Nov_20_2020", "PctWoreMask0920"),
  col2 = c(NA, NA, NA, "PctMaskStderr"),
  refVals = c(NA, NA, NA, 85),
  refTexts = "US %"
)

micromapST(
  statsDFrame = st_covid_case_rates,
  panelDesc = panel_desc_covid2,
  sortVar = "Nov_20_2020",
  ascend = FALSE,
  plotNames = "full",
  title = c(
    "US State Covid Cumulative Case Rates, Nov. 20, 2020,",
    "and percent who usually wore a mask, Sep. 2020"
  )
)

## rowNames value used is : ab
```

Looking down the map column with cumulative shading from the highest to lowest case rates reveals an enlarging cluster of states that starts in the North Central region of the U.S. This is consistent with the superspreader North Dakota motorcycle rally hypothesis. By adding an error bar to the mask-wearing dot, we can see that even though North Dakota, South Dakota and Wyoming have wide error bars, mostly due to their smaller populations, they still appear to have values lower than most other states. There appears to be a slight negative correlation between the case and mask columns. Adding the U.S. value referent line (85%) shows that nearly all of the highest 15 case rates are in states with mask wearing percents below the U.S. value and that most of the other states have mask wearing percents about the same (error bar touches the U.S. line) or slightly above the U.S. value.

In the next figure, the maps are moved to the right side of the plot by reordering the specified glyphs in `type` and the dot with error bar is changed to a horizontal bar (`bar`). Note that this requires a change in the order of values in all of the parameter lines. In addition, a second line of column labels is added (`lab2`) and the colors are changed from the default colors to predefined shades of gray. We recommend using the default colors but sometimes publishers are unable to print color. The gray scale available for optional area and glyph shading is the recommended set of shades that maximizes readability and distinctions among the areas when colors are not available (Brewer et al., 2003; Brewer & Pickle, 2002). The user can also specify a different color set by the `colors` parameter, defining the colors by their names or hexadecimal definitions. The only palette name recognized is `grays`. See the package documentation for details.

There are 12 default colors used:

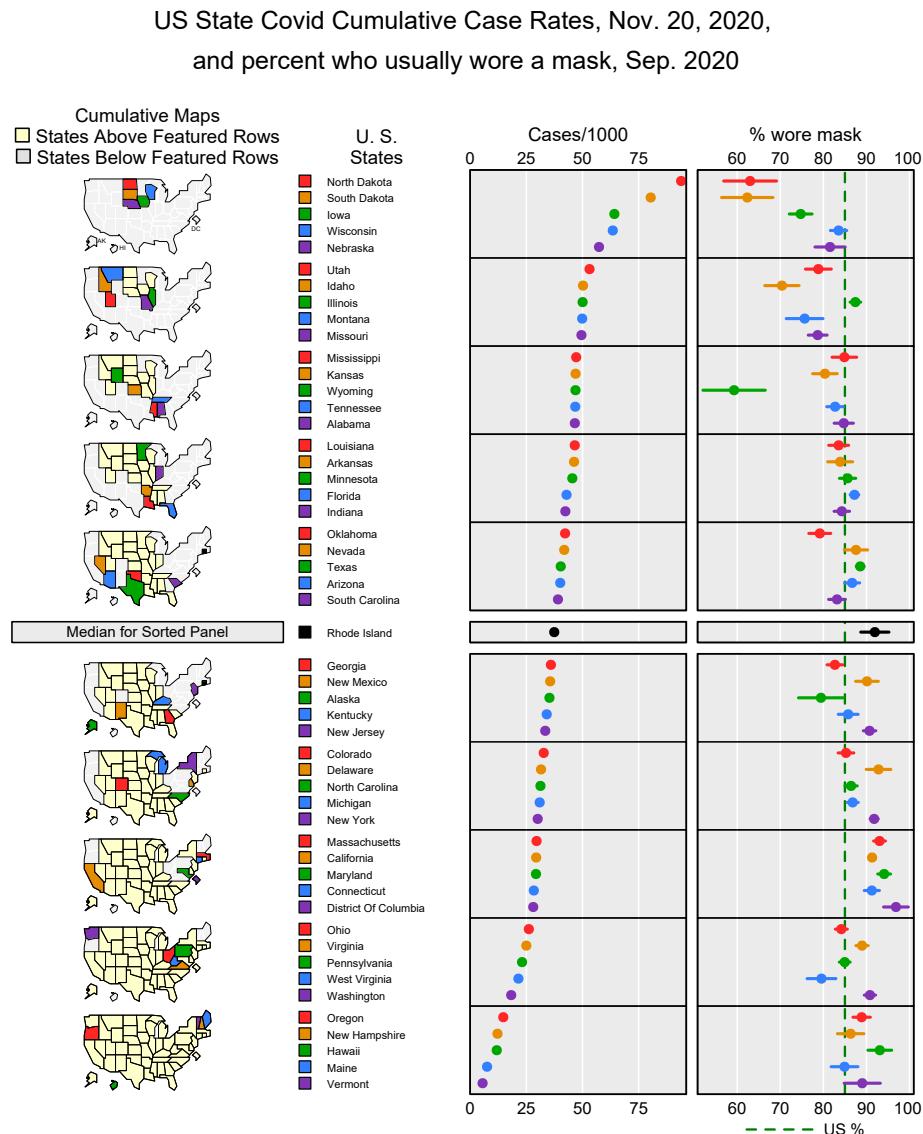


FIGURE 3.4 Cumulative number of covid-19 cases by state on Nov. 20, 2020, and percent who usually wore a mask in Sep 2020, with optional error bars, cumulative map shading and full state names.

- Up to 6 for the map areas and symbols on the glyphs (red, orange, green, blue/green, lavender, magenta/brown);
- 1 for the median area (black);
- 3 foreground colors for highlighted areas on maps with cumulative shading (light yellow, light red, light blue);
- 1 for areas not referenced (lightest gray);
- 1 for non-active background areas (lighter gray).

```

panel_desc_covid2 <- data.frame(  

  type = c("id", "dot", "bar", "mapcum"),  

  lab1 = c("", "Cases/1000", "% wore mask", ""),  

  lab2 = c("", "Nov 20, 2020", "Sep 2020", ""),  

  col1 = c(NA, "Nov_20_2020", "PctWoreMask0920", NA),  

  col2 = c(NA, NA, "PctMaskStderr", NA),  

  refVals = c(NA, NA, 85, NA),  

  refTexts = "US %"  

)  
  

micromapST(  

  statsDFrame = st_covid_case_rates,  

  panelDesc = panel_desc_covid2,  

  sortVar = "Nov_20_2020",  

  ascend = FALSE,  

  plotNames = "full",  

  colors = "grays",  

  title = c(  

    "US State Covid Cumulative Case Rates, Nov. 20, 2020",  

    " and percent who usually wore a mask, Sep. 2020"  

  )  

)

```

rowNames value used is : ab

All of the examples in this book direct the micromap display to print on these book pages. However, the output could be written to a file for inclusion in other documents. For example, to write to a PDF file, open an output file by `pdf(file = 'samplefilename', width = 7, height = 8)` before the function call to `micromapST`, then close the file by `dev.off()` after the call. `pdf` could be replaced by `png` to create that type of file or `windows` to direct the plot to a Windows display (with no filename specified). The default is to write to the open graphical window. The PDF and PNG file formats retain sufficient information to allow scaling and zooming in to see detail in the plot and so are preferred over other output formats that do not. Because the characteristics of a graphical window are much different from a file, it is difficult to predict how the window will display the micromap. Writing the micromap image to a file gives a consistent format. See Chapter 5 for an example of how to read an external image file back in for display.

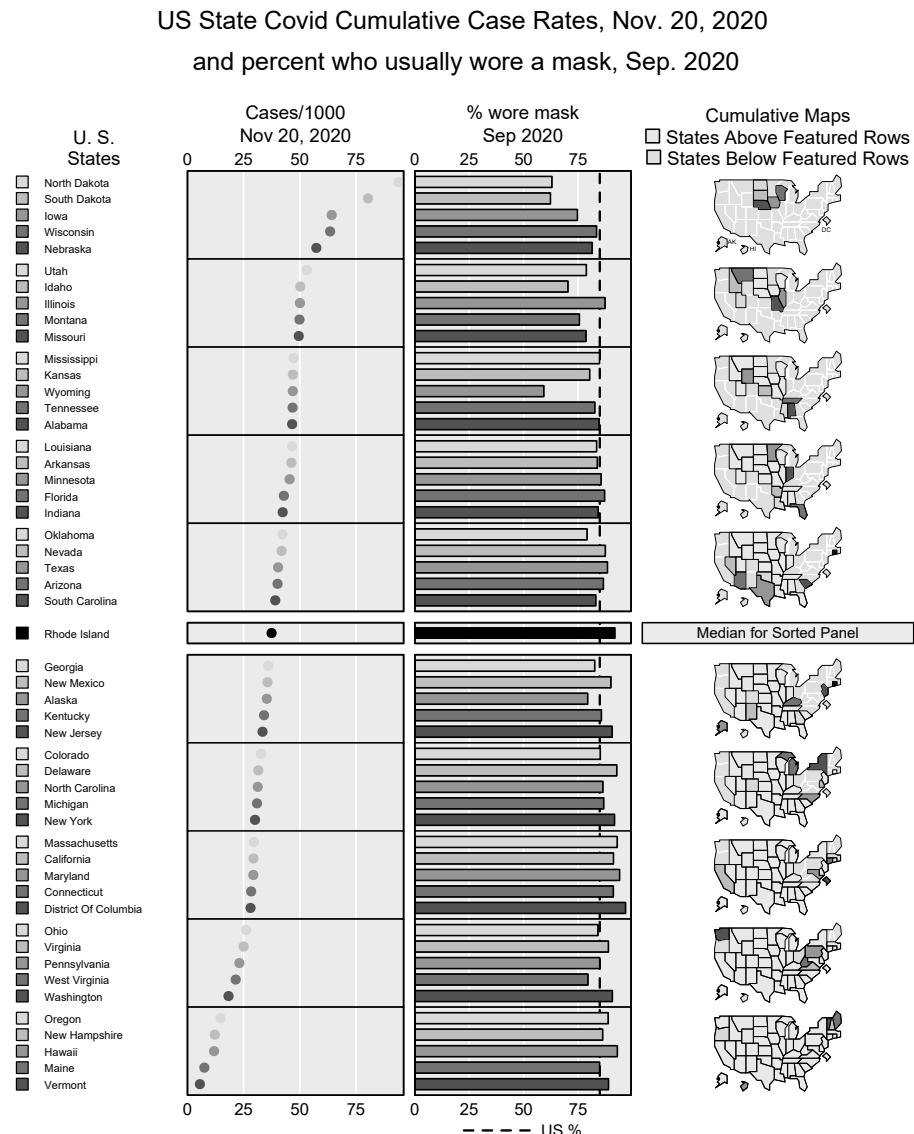


FIGURE 3.5 Cumulative number of covid-19 cases by state on Nov. 20, 2020, and percent who usually wore a mask in Sep 2020, in gray scale with expanded labels and bar glyph.

3.4 Mapping beyond U.S. states

Over time, boundary files have been added to micromapST that represent geographic units other than the original U.S. states. We refer to these as border groups. At this writing, the package includes generalized border groups for counties within the U.S. states of Kansas, Maryland, New York and Utah; the NCI cancer registries (National Cancer Institute, Surveillance, Epidemiology, and End Results Program, 2022b); districts in Seoul, South Korea; provinces and municipalities in China; and the countries of Africa. We will examine education and poverty levels for the 24 Maryland counties to illustrate how to use these additional boundary files. In Chapter 5, we will demonstrate how the user may read in their own border group file to create a linked micromap.

The data frame `MDPovEd` consists of various measures of education and poverty by county as reported on the Census Bureau's publicly available American Community Survey for 2016-2020 (United States Census Bureau, 2022a, 2022b). Maryland is a diverse state, with mostly agricultural jobs east of the Chesapeake Bay, mostly white-collar jobs in and around Baltimore and Washington D.C. (adjacent to Montgomery and Prince George's counties in MD), and mostly rural areas in the southern and western counties. Let's see if a linked micromap plot can discern these expected patterns by plotting the percent of adults who had at least four years of college and the percent of residents living below 150% of the federal poverty level. The cost of living in Maryland tends to be high, so the 150% criterion will capture those who are struggling to make ends meet even though they technically are not living below the poverty line.

First, the data file is read in, specifying the third column as containing the row names to match the area names in the border group dataset; these are the numeric county FIPS codes. The first two counties' data are printed for illustration. Then a simple linked micromap is requested, with full county names listed as the geographic identifier and the rows sorted by college education in descending order (`sortVar = "PctBachDegree"`, `ascend = FALSE`). This example follows the design and display specification for Figure 3.3. Note that because the plot is for a border group other than U.S. states, we need to specify `bordGrp = "MarylandBG"`.

```
library(micromapST)

MDPovEd <- read.csv("data/MDPovEducACSData20162020.csv", row.names = 3)
head(MDPovEd, n = 2L)

##          CountyName StateName PopTotal PopMarginOfErr Pop25Up Pop25MarginErr
## 24001      Allegany    Maryland     62610            538    44371            387
## 24003   Anne Arundel    Maryland     558904           1489   388227            912
##          NumLessHS NumHS NumSomeColl NumBachDegree NumLess125Pov NumLess150Pov
## 24001      3672 17411        13856         9432     13376        16509
## 24003      24922 86242       106991        170072     42915        54604
##          PctLessHS PctHS PctSomeColl PctBachDegree PctLess125Pov PctLess150Pov
## 24001      0.083 0.392        0.312        0.213      0.214        0.264
## 24003      0.064 0.222        0.276        0.438      0.077        0.098
```

```

panel_desc_MD <- data.frame(  

  type = c("mapcum", "id", "dot", "dot"),  

  lab1 = c("", "", "% College", "% Poverty"),  

  col1 = c(NA, NA, "PctBachDegree", "PctLess150Pov")  

)  
  

# sort by % college  

micromapST(  

  statsDFrame = MDPovEd,  

  panelDesc = panel_desc_MD,  

  rowNames = "id",  

  sortVar = "PctBachDegree",  

  ascend = FALSE,  

  bordGrp = "MarylandBG",  

  plotNames = "full",  

  title = c(  

    "Maryland counties: % Living < 150% of Poverty Level",  

    "and % with 4+ Years of College"  

)
)
)

```

```
# Note 2 character strings for 2 lines of the long title
```

This plot shows the expected negative correlation between college education and income – counties with more college-educated residents tend to have a lower instance of poverty. The maps show a cluster of the highest education/lowest poverty in the DC suburbs plus Talbot County on the Eastern Shore, followed by a cluster of the Baltimore suburbs plus two more Eastern Shore counties. The cumulative shading through the first three maps defines a contiguous block of 14 counties in central Maryland with higher income and lower poverty than in the other 10 counties, which are located in western Maryland, the lower Eastern Shore, Cecil County in the northeast corner and Charles County in the southwest corner of the state.

The 24 counties on this plot were split into the default perceptual group size of five counties each from the bottom and top, with the remaining four counties shown in the middle group. Perhaps the geospatial patterns would be clearer by using smaller groups so that there were fewer than four counties in the middle group. Using the same design specified by `panel_desc_MD`, the revised perceptual grouping is specified by `grpPattern = c(5, 4, 3, 3, 4, 5)` from top to bottom (Symanzik & Carr, 2008). Note that the panel height is determined by the map size and so is not affected by the number of areas in each panel.

```

micromapST(  

  statsDFrame = MDPovEd,  

  panelDesc = panel_desc_MD,  

  rowNames = "id",  

  sortVar = "PctBachDegree",  

  ascend = FALSE,  

  bordGrp = "MarylandBG",  

  plotNames = "full",
)

```

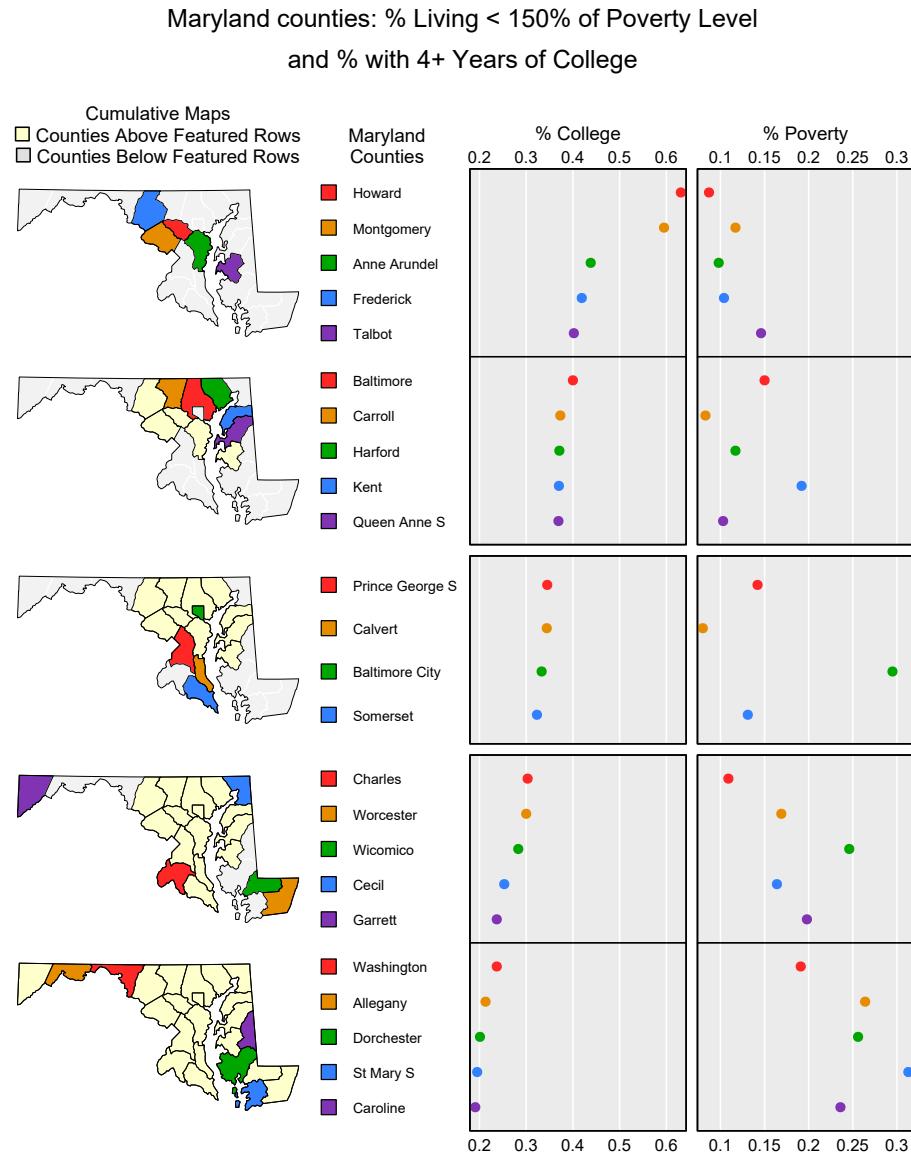


FIGURE 3.6 Percent of Maryland residents with income < 150% of the federal poverty level and percent with 4+ years of college, by county.

```
grpPattern = c(5, 4, 3, 3, 4, 5),
title = c(
  "Maryland counties: % Living < 150% of Poverty Level",
  " and % with 4+ Years of College"
)
)
```

Since the default grouping led to a fairly clear clustering of counties within Maryland, this revised grouping is only slightly better.

3.5 Summary and Further Reading

In this chapter the reader has learned how to create a basic linked micromap plot using the package **micromapST**, to enhance that plot with several display options and to use the additional boundary group files included in the package. In Chapter 7 we will describe the many graphics (glyphs) built into the package, beyond the simple ones used in this chapter to illustrate the basic plot. In Chapter 5, we will show how to read in an external geographic boundary file and to process it for use as a micromap.

The precursor to **micromapST**, written in JAVA, was implemented on the National Cancer Institute's State Cancer Profiles web site to communicate cancer data to interested readers and to allow them to quickly explore the data in conjunction with possible risk factors (Bell et al., 2006; Carr et al., 2003). Functions from the JAVA package were rewritten in R to support creation of many examples in Carr and Pickle's 2010 book (Carr & Pickle, 2010). That book provides much of the background on the three types of micromaps, such as the cognitive and data visualization principles underlying their designs.

Version 1.0 of **micromapST** was released in 2013, with the R code and resulting examples further described in a 2015 paper (Pickle et al., 2015). This version of the package was used for data visualization classes at George Mason University in Fairfax, Virginia. Student feedback led to additional options and built-in geographic boundaries beyond U.S. states (version 1.1). Later versions added the ability to import the user's own boundary file (version 2.0) and a rewrite of code to avoid dependence on several retiring R packages (version 3.0).

The advantages of building many functions and datasets into the package include ease of use by those not familiar with R coding, freeing the user to focus on the task of examining patterns in the data. The package **micromap** uses a different approach, requiring the user to know enough about R coding to take advantage of that package's greater flexibility in plot design. Because **micromapST** draws the linked micromap using the `polygon` function in base R, with no need to call another drawing package, execution is very fast. Thus linked micromaps with different variables and display options can be quickly generated to identify the best plot to use for the purpose at hand. The disadvantage of this approach is that the user has less control of display features, although we have attempted to create publication-ready output without user modification and to offer many optional display modifications through parameter choices. The ease of use combined with speed of execution makes this an excellent tool for teaching, exploring and communicating spatial patterns and generating hypotheses for further analysis.

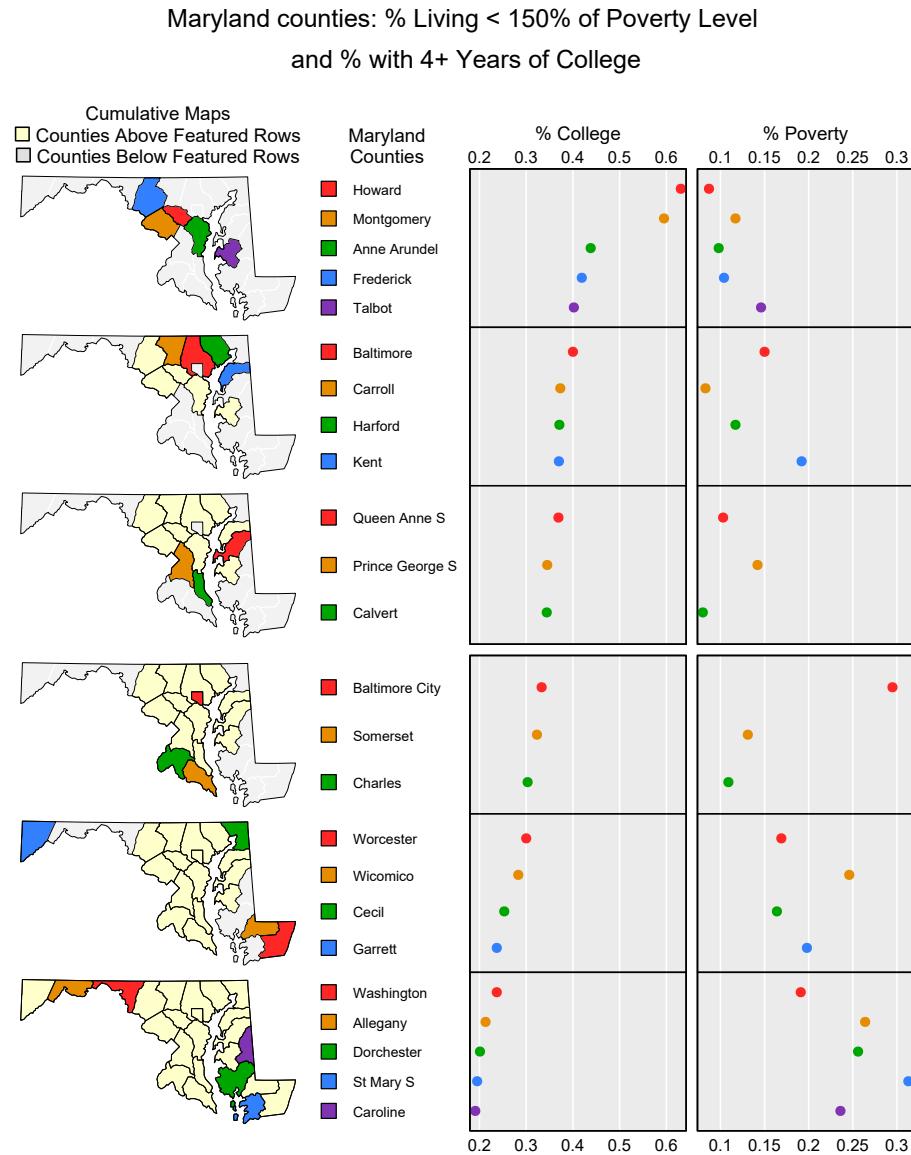


FIGURE 3.7 Percent of Maryland residents with income < 150% of the federal poverty level and percent with 4+ years of college by county, displayed in smaller perceptual groups.

The package **micromap** uses a different approach, requiring the user to know enough about R coding to take advantage of that package's greater flexibility in plot design.

4

Modifying Shapefiles for Use in Micromaps

BRADEN PROBST

In the past, modifying shapefiles from publicly available sources such as the Global Administrative Areas Database (GADM - <https://gadm.org/>) has been a challenge for all but the most advanced R programmers. Often, there exist relatively small geographic areas that have to be enlarged to be visible in micromaps and possibly have to be shifted to a different location to become better visible. Similarly, regions far away from the main geographic area (such as Alaska and Hawaii for the United States) have to be shifted to reduce the empty space on a map. A recently developed (but still unpublished) R package, LMshapemaker (<https://digitalcommons.usu.edu/etd/7751/>), developed by the author of this chapter, will be the basis for this chapter.

4.1 Introduction

As a reminder, see Chapter 1 for general style requirements for our *Micromap Plots in R* book. In particular, please do the following:

- Introduce meaningful labels for the sections, figures, and tables in your chapter.
 - Create index entries for all R packages (such as the **micromap** R package) and for all datasets (such as the *USstates* and *edPov* datasets) that are used in your chapter.
 - Include references for R packages and publications related to your chapter, such as for the **micromap** (Payton & Olsen, 2015) and **micromapST** (Carr & Pearson Jr., 2015) R packages and some micromap articles, book chapters, and books (Carr, 2001; Carr & Pickle, 2010; Symanzik & Carr, 2008).
 - Also create index entries for main topics such as linked micromap plots, conditioned choropleth maps, perceptual group, color blindness,, and quantile-quantile plot.
-

4.2 Outline

- overview of boundary files / regions that are part of the micromap R package
- use of external shapefiles
- thinning only (from within R)

- approach used for Latin American countries (from within R)
 - use of the LMshapemaker R package at the command level
 - the Shiny interface for LMshapemaker
 - shapefile modifications beyond the LMshapemaker R package
-

4.3 Main

Here goes the main content of your chapter. Introduce additional sections as needed.

For convenience, Figure 4.1 shows one linked micromap plot (which is the same as in Figure 1.6), but now formatted in a slightly more meaningful way.

```
library(micromap)

# initial example

data(USstates)
statePolys <- create_map_table(USstates, "ST")
data(edPov)

# basic figure 1
mmpplot(
  stat.data = edPov,
  map.data = statePolys,
  panel.types = c("labels", "dot", "dot", "map"),
  panel.data = list("state", "pov", "ed", NA),
  ord.by = "pov",
  grouping = 5,
  median.row = TRUE,
  plot.width = 2,
  plot.height = 6,
  map.link = c("StateAb", "ID")
))
```

4.4 Code Chunks China

Example moved from Chapter 2.

```
# VI. Use of external shapefiles

library(micromap)
```

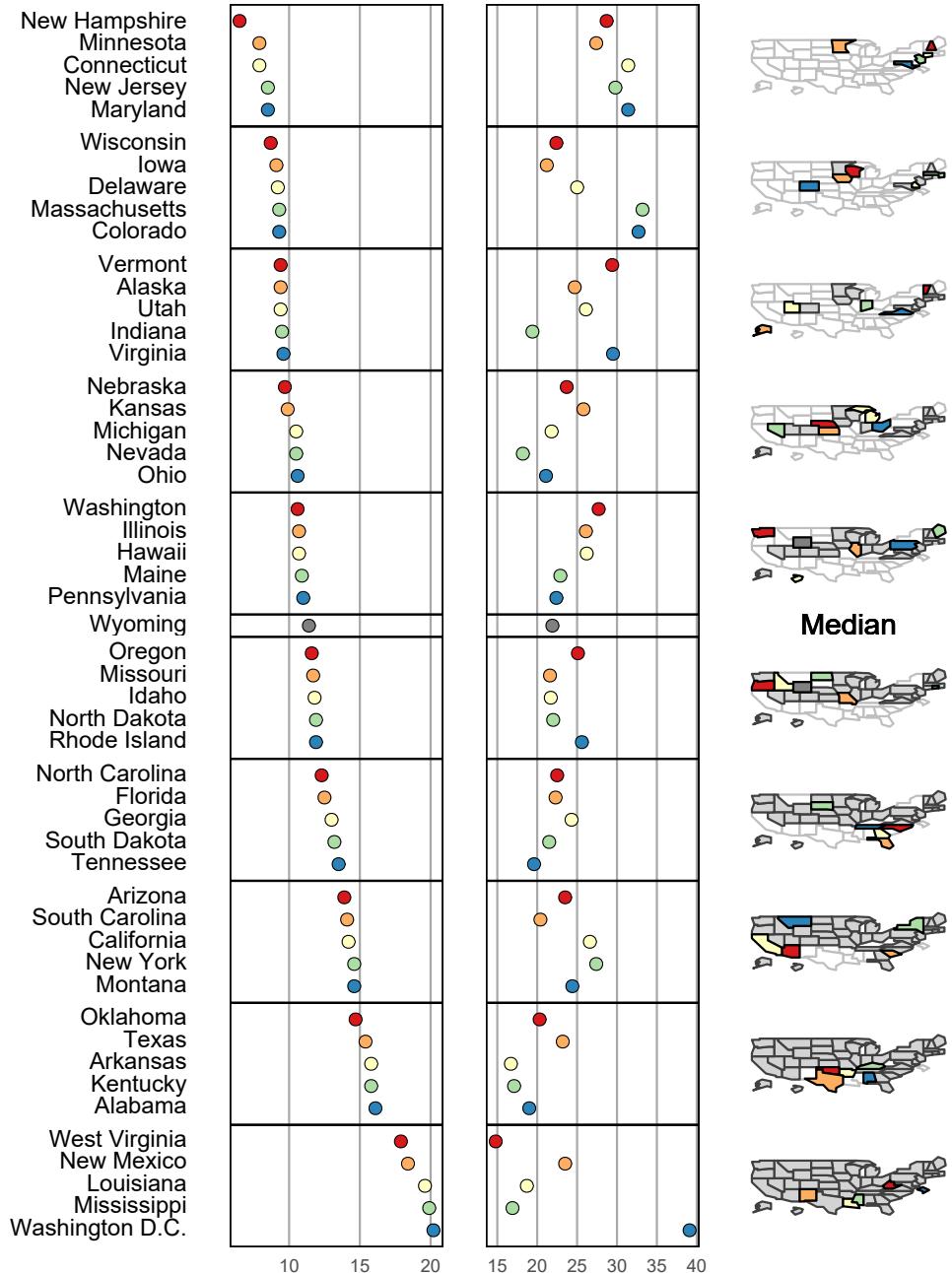


FIGURE 4.1 Here is a first micromap example for this chapter. Note that the figure is formatted in a slightly more meaningful way this time.

```
### Read in the shapefile and simplify the polygons (run this part of the code only once!)

#ChinaShapefile <- readOGR(
#  dsn = "data/China_Shapefiles",
#  layer = "export",
#  verbose = TRUE
#)

ChinaShapefile <- sf:::as_Spatial(sf:::st_read("data/China_Shapefiles/export.shp"))

## Reading layer `export' from data source
## `D:/Dropbox/JUE/Papers/2022_MicromapBook/MicromapRBookSource/data/China_Shapefiles/export.shp'
##   using driver 'ESRI Shapefile'
## Simple feature collection with 31 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: 8176000 ymin: 704800 xmax: 15040000 ymax: 7087000
## Projected CRS: Google_Mercator

plot(ChinaShapefile)

summary(ChinaShapefile)

## Object of class SpatialPolygonsDataFrame
## Coordinates:
##       min      max
## x 8176078 15037685
## y 704818 7086873
## Is projected: TRUE
## proj4string :
## [+proj=merc +lat_ts=0 +lon_0=0 +x_0=0 +y_0=0 +R=6378137 +units=m
## +no_defs]
## Data attributes:
##       chname      ename      gbcde      a107002
## Length:31      Length:31      Length:31      Min. : 1325371
## Class :character  Class :character  Class :character  1st Qu.:10806898
## Mode  :character  Mode  :character  Mode  :character  Median :18464477
##                                         Mean   :20654064
##                                         3rd Qu.:30710030
##                                         Max.  :47046599
##       a107003      a601002      a601003
## Min. : 1290958  Min. : 9431  Min. : 8787
## 1st Qu.:10084531 1st Qu.: 64698 1st Qu.: 48377
## Median :16986313 Median :118755 Median : 90900
## Mean   :19430202 Mean   :132470 Mean   :103436
## 3rd Qu.:28544378 3rd Qu.:187550 3rd Qu.:147096
## Max.  :44429729  Max.  :311596  Max.  :252049
```



FIGURE 4.2 Chunk 1.

```

# gIsValid(spgeom = ChinaShapefile, byid = TRUE, reason = TRUE)
#
# ChinaShapefileThin <- thinnedSpatialPoly(
#   SP = ChinaShapefile,
#   tolerance = 300,
#   # try 100, 300, 3000, 30000
#   minarea = 10000000000,
#   topologyPreserve = TRUE,
#   avoidGEOS = TRUE
# )
#
# ChinaShapefileThin <- gBuffer(
#   spgeom = ChinaShapefileThin,
#   width = 0,
#   byid = TRUE
# )
#
# gIsValid(spgeom = ChinaShapefileThin, byid = TRUE, reason = TRUE)

ChinaShapefileThin <- sf::st_read("data/China_Shapefiles/export.shp") %>%
  st_simplify(dTolerance = 2000) %>%
  as_Spatial()

## Reading layer `export' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\China_Shapefiles\export.shp'
## using driver `ESRI Shapefile'
## Simple feature collection with 31 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: 8176000 ymin: 704800 xmax: 15040000 ymax: 7087000
## Projected CRS: Google_Mercator

plot(ChinaShapefileThin)

### Change tolerance to 100, 3000, and 30000 and run the entire code again, starting with readOGR.
### Which of these is the best choice for tolerance? You may have to enlarge the map.

### If you plan to use LM plots for a research project, you may also want to
### enlarge small regions or shift far-away regions closer to the main region.
### Talk to me for further information.

### Read in the data

religion <- read.csv(
  file = "data/China_ReligionData.csv",
  header = TRUE
)

```



FIGURE 4.3 Chunk 1.

```

ChinaPolys <- create_map_table(
  tmp.map = ChinaShapefileThin,
  IDcolumn = "ename"
)

### Basic micromap plot

ChinaPlot <- mmplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c("labels", "dot", "map"),
  panel.data = list("Province", "Christianity", NA),
  ord.by = "Christianity",
  grouping = 5,
  median.row = TRUE,
  map.link = c("Province", "ID")
)

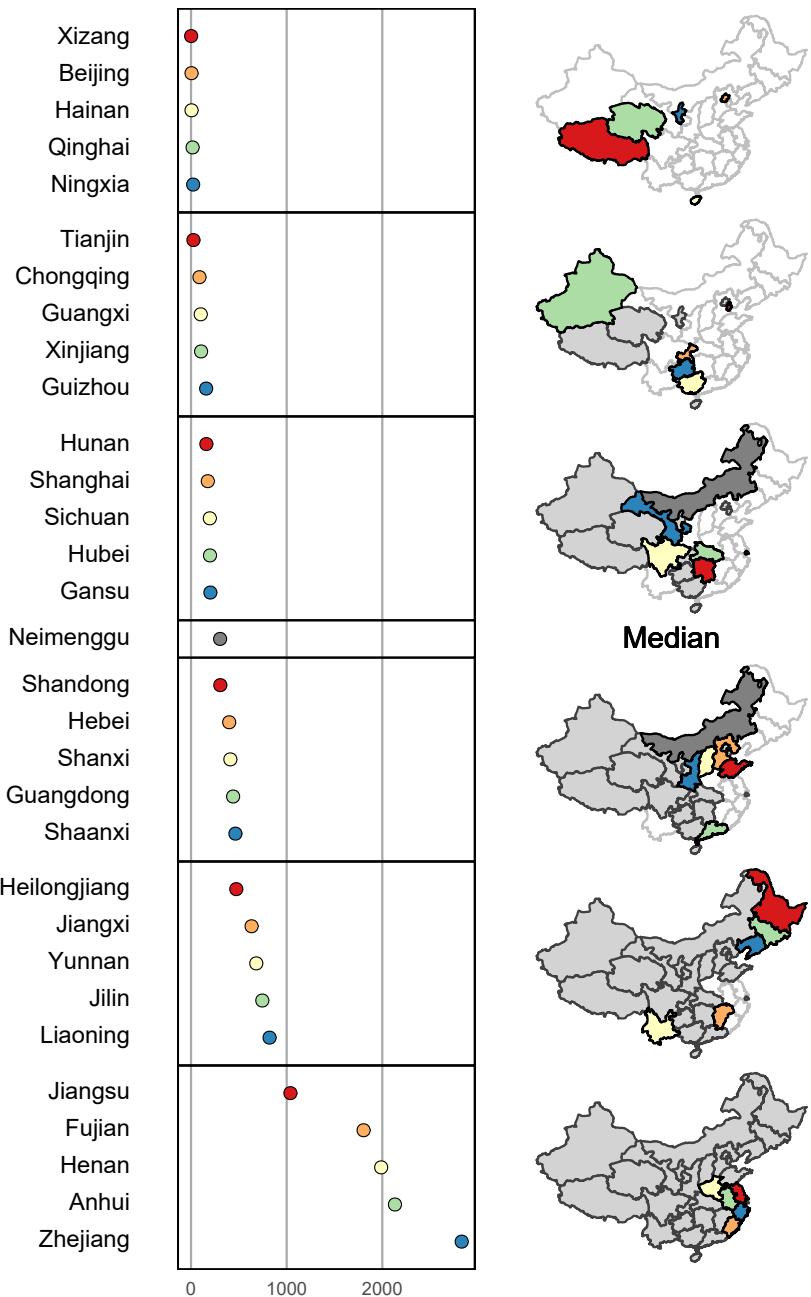
```

```

### Advanced micromap plot

ChinaReligion <- mmplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c(
    "map", "dot_legend", "labels",
    "dot", "dot", "dot", "dot"
  ),
  panel.data = list(
    NA,
    NA,
    "Province",
    "Christianity",
    "Buddhism",
    "Daoism",
    "Islam"
  ),
  map.link = c("Province", "ID"),
  ord.by = "Christianity",
  rev.order = TRUE,
  grouping = 5,
  median.row = TRUE,
  plot.height = 9,
  plot.width = 9,
  colors = c("red", "orange", "green", "blue", "purple"),
  two.ended.maps = TRUE,
  map.all = TRUE,
  map.color2 = "lightgray",
  plot.header = "Religion in China",
)

```

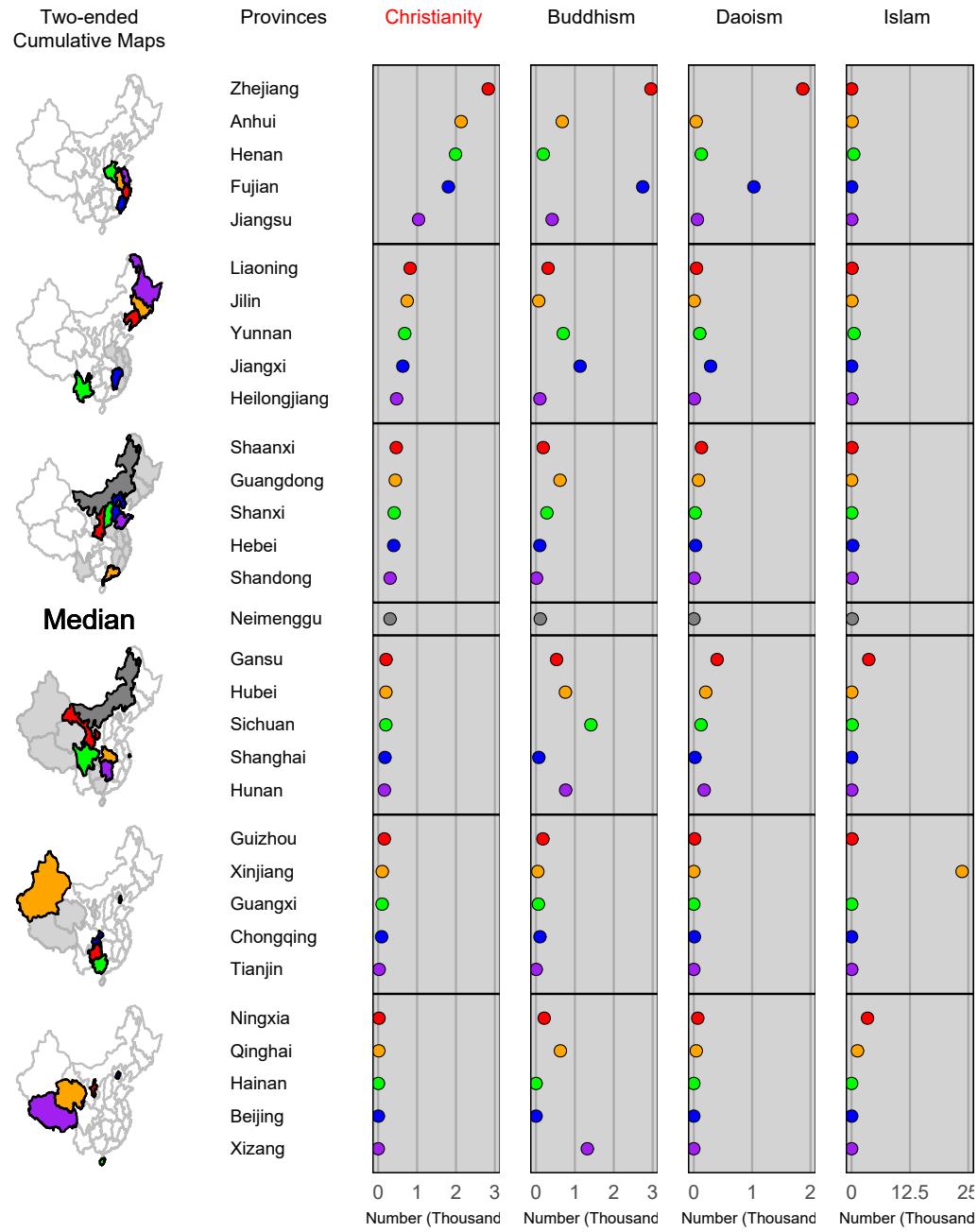
**FIGURE 4.4** Chunk 1.

```
plot.header.size = 2,
plot.header.color = "black",
plot.panel.spacing = 0,
panel.att = list(
  list(
    1,
    header = "Two-ended\nCumulative Maps",
    inactive.border.color = gray(.7),
    inactive.border.size = 1,
    panel.width = 1.2
  ),
  list(2,
    point.type = 20,
    point.size = 1.4
  ),
  list(
    3,
    header = "Provinces",
    panel.width = .9,
    align = "left",
    text.size = .8
  ),
  list(
    4,
    header = "Christianity",
    header.color = "red",
    graph.bgcolor = "lightgray",
    point.size = 1,
    xaxis.ticks = seq(0, 3000, by = 1000),
    xaxis.labels = seq(0, 3, by = 1),
    xaxis.title = "Number (Thousand)"
  ),
  list(
    5,
    header = "Buddhism",
    graph.bgcolor = "lightgray",
    point.size = 1,
    xaxis.ticks = seq(0, 3000, by = 1000),
    xaxis.labels = seq(0, 3, by = 1),
    xaxis.title = "Number (Thousand)"
  ),
  list(
    6,
    header = "Daoism",
    graph.bgcolor = "lightgray",
    point.size = 1,
    xaxis.ticks = seq(0, 2000, by = 1000),
    xaxis.labels = seq(0, 2, by = 1),
    xaxis.title = "Number (Thousand)"
  )
)
```

```
  ),
  list(
    7,
    header = "Islam",
    graph.bgcolor = "lightgray",
    point.size = 1,
    xaxis.ticks = seq(0, 25000, by = 12500),
    xaxis.labels = seq(0, 25, by = 12.5),
    xaxis.title = "Number (Thousand)"
  )
)
)
```

```
### Final advanced micromap plot with reduced margins
```

```
ChinaReligion <- mmplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c(
    "map", "dot_legend", "labels",
    "dot", "dot", "dot", "dot"
  ),
  panel.data = list(
    NA,
    NA,
    "Province",
    "Christianity",
    "Buddhism",
    "Daoism",
    "Islam"
  ),
  map.link = c("Province", "ID"),
  ord.by = "Christianity",
  rev.order = TRUE,
  grouping = 5,
  median.row = TRUE,
  plot.height = 9,
  plot.width = 9,
  colors = c("red", "orange", "green", "blue", "purple"),
  two.ended.maps = TRUE,
  map.all = TRUE,
  map.color2 = "lightgray",
  plot.header = "Religion in China",
  plot.header.size = 2,
  plot.header.color = "black",
  plot.panel.spacing = 0,
  panel.att = list(
    list(
      1,
```

**FIGURE 4.5** Chunk 1.

```
header = "Two-ended\nCumulative Maps",
inactive.border.color = gray(.7),
inactive.border.size = 1,
panel.width = 1.2
),
list(2,
  point.type = 20,
  point.size = 1.4
),
list(
  3,
  header = "Provinces",
  panel.width = .9,
  align = "left",
  text.size = .8
),
list(
  4,
  header = "Christianity",
  header.color = "red",
  graph.bgcolor = "lightgray",
  point.size = 1,
  right.margin = 0,
  left.margin = -0.5,
  xaxis.ticks = seq(0, 3000, by = 1000),
  xaxis.labels = seq(0, 3, by = 1),
  xaxis.title = "Number (Thousand)"
),
list(
  5,
  header = "Buddhism",
  graph.bgcolor = "lightgray",
  point.size = 1,
  right.margin = 0,
  left.margin = -0.5,
  xaxis.ticks = seq(0, 3000, by = 1000),
  xaxis.labels = seq(0, 3, by = 1),
  xaxis.title = "Number (Thousand)"
),
list(
  6,
  header = "Daoism",
  graph.bgcolor = "lightgray",
  point.size = 1,
  right.margin = 0,
  left.margin = -0.5,
  xaxis.ticks = seq(0, 2000, by = 1000),
  xaxis.labels = seq(0, 2, by = 1),
  xaxis.title = "Number (Thousand)"
```

```

),
list(
  7,
  header = "Islam",
  graph.bgcolor = "lightgray",
  point.size = 1,
  right.margin = 0.25,
  left.margin = -0.5,
  xaxis.ticks = seq(0, 25000, by = 12500),
  xaxis.labels = seq(0, 25, by = 12.5),
  xaxis.title = "Number (Thousand)"
)
)
)
)

```

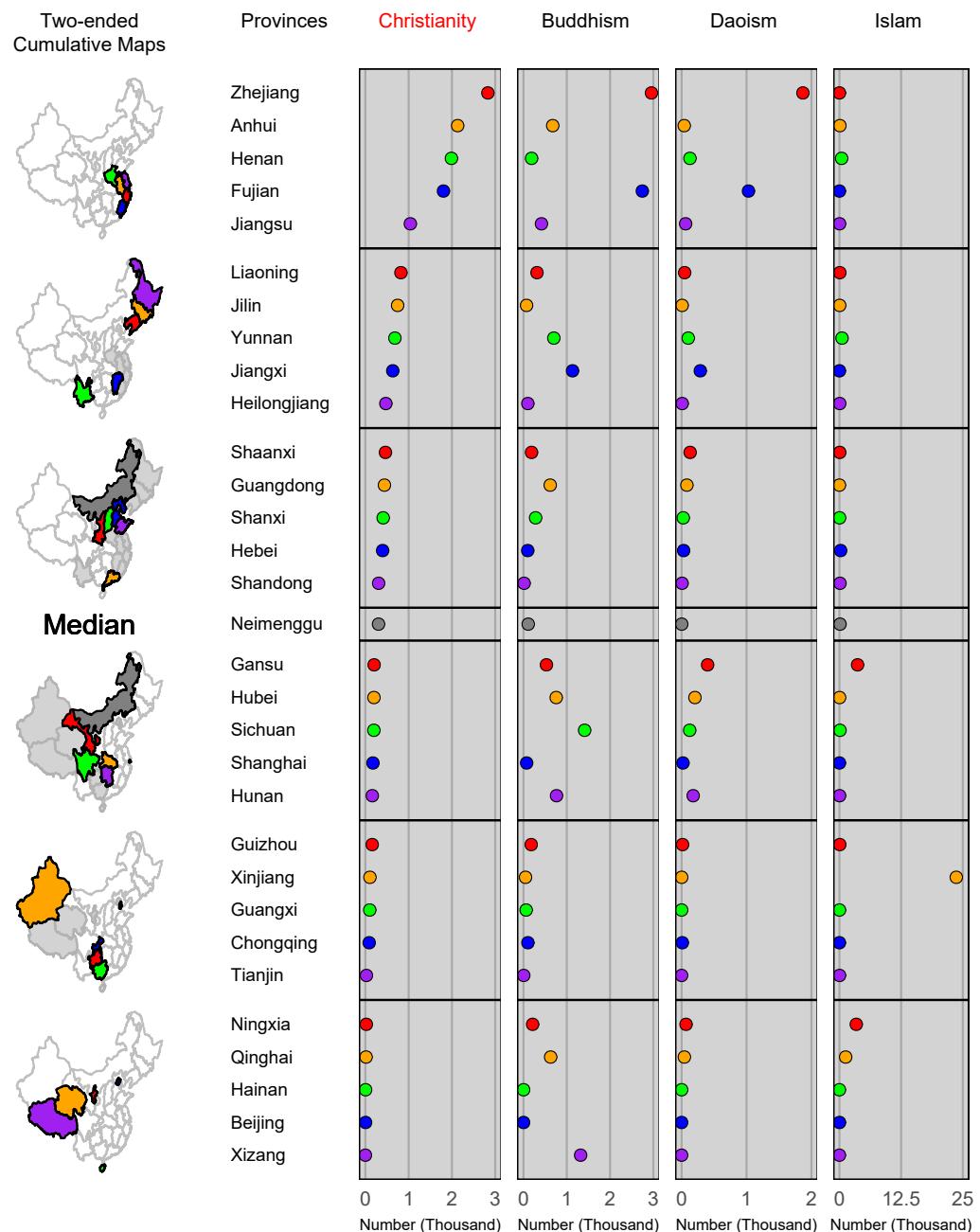
4.5 Summary and Further Reading

Introduce cross-references to other chapters, e.g., Chapter 1 and Chapter 2, where related work and further examples can be found in this book that match the content of this chapter, that follow up on this chapter, or that are a prerequisite of this chapter.

Also, do some scientific literature review here that is specific to your chapter. Where has this R package been introduced and used before, where have other plot types or different countries been used in micromaps, what were other applications of micromaps that are related to the title and content of your chapter, etc.?

References

- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.

**FIGURE 4.6** Chunk 1.

5

*Using Your Own Boundary Files in the **micromapST** R Package*

JAMES BLACKWOOD PEARSON, JR., LINDA WILLIAMS PICKLE, DANIEL B. CARR

Chapter 3 described how to create a simple linked micromap plot for U.S. states and Maryland counties using the **micromapST** R package (Carr & Pearson Jr., 2015), accessible at <https://cran.r-project.org/web/packages/micromapST/>. In this chapter, the reader will learn about other geographies available in the package and how to process their own shapefile to create a map file for new linked micromap plots. The latter option is an advanced feature in the package, best suited to those familiar with R programming. The directions here will sound complicated but need only be done once to produce future linked micromap plots for any geographic area.

5.1 Introduction

Although the original package was written to display U.S. states exclusively, over time other geographies have been added at the request of users. Starting in version 2.0 of **micromapST**, the process used to create these other boundary files has been codified into a new function to allow the user to create linked micromap plots with their geographic areas of choice.

The resulting map serves as the visual element linking graphed data to spatial aspects of the data, and so must be accurate enough to allow the user to identify the geographic space it represents. This is a challenge, as the maps in a linked micromap design are typically exceedingly small, about 1" x 1.5", but the overall map area and its sub-unit boundaries and fill colors need to be recognizable.

The micromap design includes multiple map images, one for each perceptual group of geographic units. These maps must be displayed as fast as possible to make the display process usable, especially for iterative data exploration. By using a minimal set of boundary points, **micromapST** can use a fundamental function in R (`polygon`) to quickly draw the multiple maps. For example, the Spatial Polygon Data frame and simple feature structures used for spatial objects in R are more complicated than are needed by **micromapST**. Using a more basic structure and not needing to call another plotting package helps **micromapST** provide the requested linked micromap plot almost instantaneously.

In addition to speed, by reducing the number of coordinates in the boundary file to only the essential ones, the boundary data can be greatly reduced in both size and complexity.

This enables the reader to better see the area borders and internal shading. For example, the U.S. Census Bureau 2000 shapefile of the boundaries for the U. S. states and DC is 2,360,011 bytes. After the simplification process described in this chapter, the U.S. state dataset is only 29,092 bytes, a 98.8% reduction.

The boundary data used for the default map of U.S. states in **micromapST** is based on Mark Monmonier’s “visibility map” (Monmonier, 1993), a caricature map of the U. S. states and DC. By highly smoothing the state boundaries and slightly enlarging several small states, these maps are not only fast to draw but minimize the boundary ink that can hamper identification of the interior color shading for small areas. This is especially a problem in coastal areas, where the complexity of coastlines can lead to black border lines folding in on themselves, resulting in the border ink masking the interior color. Of course, the degree of smoothing necessary depends on the size of the final map. As an example, the outline of the state in the linked micromap of Maryland counties (Figure 3.6) includes the Chesapeake Bay with some coastline detail whereas the boundary of Maryland within the larger U.S. map does not.

In the next section, we describe the boundary files included in the **micromapST** package. Then in following sections, the reader will learn how to read and process an external boundary file for use in **micromapST**.

5.2 Boundary Files Included in *micromapST*

We will refer to a collection of boundaries for a geographic entity as a “border group”, giving it a data frame name ending in “BG”. The following border groups were created by request over the years and were available beginning with version 2.0 of **micromapST**:

- USStatesBG, data from the original *micromapST* package for the 50 U.S. states and the District of Columbia;
- USSeerBG, data for the 21 cancer registry areas of the U.S. National Cancer Institute (National Cancer Institute, Surveillance, Epidemiology, and End Results Program, 2022a) and their state boundaries;
- KansasBG, data for the 105 counties in the state of Kansas;
- NewYorkBG, data for the 62 counties in the state of New York;
- MarylandBG, data for the 23 counties and Baltimore City in the state of Maryland;
- UtahBG, data for the 29 counties in the state of Utah;
- ChinaBG, 35 provinces, regions and municipalities in the country of China;
- UKIrelandBG, 219 administrative areas in UK, Ireland and Isle of Man;
- SeoulKoreaBG, 25 districts in the city of Seoul, S. Korea;
- AfricaBG, 52 countries on the African continent.

For simplicity, hereafter we will refer generically to U.S. states, even though DC is a district, not a state. Each of these border groups can be specified for use by including `bordGrp = "MarylandBG"`, for example, in the `micromapST` function call. An example is shown in Figure 3.6.

5.3 Creating Border Groups for Use in micromapST

5.3.1 Introduction

The experience of the developers when manually creating the border groups listed in Section 5.2 has been combined into a single function in the new **micromapST** package, `BuildBorderGroup`. The user provides a file of the location information (a Name Table) and a shapefile containing all of the area boundaries. The shapefile was originally developed by the Environmental Systems Research Institute, Inc. (ESRI) for use in their GIS packages but has since become a publicly-available spatial file format (Environmental Systems Research Institute, Inc., 1998).

To illustrate the process of building a border group, we will use the U.S. Census Bureau's state boundary files as defined in 2000 (United States Census Bureau, 2021). The resulting map will be compared to the map based on the `USStatesBG` border group, the familiar caricature which is the default map in the package. Noting differences between these maps can inform the user's decision about the degree of smoothing that is enough so that all area colors are visible but not so much that areas are unrecognizable.

The `BuildBorderGroup` function includes user options to display intermediate map images as the smoothing process proceeds, to aid in debugging and troubleshooting the function and to illustrate the impact of each step of the process on the final map image. These options are specified by `debug = x` in the `BuildBorderGroup` call. `debug` is a 16-bit integer, where each bit requests a different option if set to 1; the default is for each bit to be set to 0. Thus multiple options can be requested by setting multiple bits to 1, then specifying the integer value represented by that particular sequence of bits. Bits 8-11 are most relevant to `BuildBorderGroup`; see documentation for further details. Setting one of these bits to 1 requests the following option:

- bit 8 (value 128) sets the output image file type to PNG, instead of the default PDF;
- bit 9 (value 256) generates multiple small images of the map, each with 5 areas shaded, to show what the final map images in the linked micromap would look like;
- bit 10 (value 512) generates an approximately 4" by 4" image of the map at key processing stages: the raw (input) map, after smoothing by `rmapshaper`'s `simplify` function, after change requests in the name table have been applied and the final map;
- bit 11 (value 1024) is similar to bit 10, but only generates the raw and final map images.

For example, to illustrate multiple requests using the single integer, we can request saving the raw and final map images (bit 11, value = 1024) in PNG format (bit 8, value = 128) by using `debug = 1152` ($1024 + 128 = 1152$).

5.3.2 Challenges

The original Census boundaries for U.S. states illustrate challenges to clear map display common to many other mapping tasks, particularly when the final map will be very small, as in a linked micromap plot.

```
library(sf)
# assume a working directory has been set
shp_dir <- "data/Ch4b"
```

```

shp_file <- "st99_d00" # Census 2000 US state boundary shapefile
US_shp <- sf:::st_read(shp_dir, shp_file)

## Reading layer `st99_d00' from data source
##   'D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch4b'
##   using driver 'ESRI Shapefile'
## Simple feature collection with 273 features and 10 fields
## Geometry type: POLYGON
## Dimension:      XY
## Bounding box:  xmin: -179.1 ymin: 17.88 xmax: 179.8 ymax: 71.35
## CRS:           NA

par(mar = c(0, 0, 0, 0)) # put full extent of map in figure, no margins
plot(st_geometry(US_shp), axes = TRUE)
box(which = "figure") # draw a box around the entire figure

```

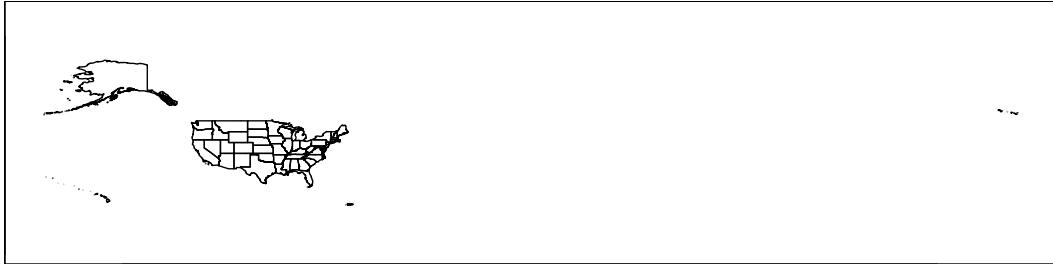


FIGURE 5.1 U.S. state boundaries plotted with latitude/longitude coordinates as provided by the U.S. Census Bureau (United States Census Bureau, 2021).

Several issues are visible in this initial map.

- Some of the Alaskan Aleutian Islands are in the Eastern Hemisphere (longitude = 0 to +180 degrees) while most of the state and the rest of the country are in the Western Hemisphere (longitude = 0 to -180 degrees). Therefore, as shown in Figure 5.1, the few Aleutian Islands that are in the Eastern Hemisphere plot to the far right of the map, while the Western Hemisphere islands plot at the far left, causing an extremely wide map image.
- Alaska and Hawaii are not close to the continental U. S. Plotting them in their correct locations requires a large mapping space, reducing the size of all areas, including the continental U.S., as seen above.
- Some of the areas are very small at this scale and any color shading will not be visible to the reader, such as in northeastern coastal areas in Figure 5.1. This is an example of too much border ink.
- Puerto Rico is included on the map, being a U.S. territory, but many datasets do not include data for it.

All of these issues need to be addressed to create a usable map for linked micromaps.

5.3.3 Steps in the Process

The steps implemented by `BuildBorderGroup` in **micromapST** are as follows:

- Create a Name Table that provides information about each area, including the geographic identifier that will link the data to be mapped to the boundary data. Steps include:
 - Validate that the Name Table has sufficient information for geoprocessing;
 - Add any additional information about each area, e.g., a region name;
 - Scale and move areas as specified by the user, e.g., Alaska and Hawaii in the U.S.;
 - Add labels for any areas that will be moved, if needed;
 - Read the input shapefile into R and process its polygons;
 - Validate that polygons are complete, shared boundaries align, etc.;
 - Simplify the polygon boundaries by applying `rmapshaper` (Teucher & Russell, 2022);
 - Link the Name Table to the simplified shapefile polygons;
 - Apply the Albers equal-area (default) or other specified projection to the polygons;
 - Create internal data frame tables required by **micromapST**.
-

5.4 Building a Border Group for **micromapST**

In this section, we will provide the details of each step, the necessary R code and the resulting images.

If a shapefile exists that is already appropriately smoothed for a linked micromap, e.g., one created by a GIS package, we still need to call `BuildBorderGroup` to read the shapefile and the Name Table in order to create the internal tables needed for **micromapST**. However, the option `ReducePC = 100` (Reduce Percentage) must be included so that there is no further smoothing of the boundaries (i.e., output is 100% of input).

`BuildBorderGroup` writes out a checkpoint shapefile just before building tables (after smoothing, etc.) that are needed to create the linked micromap plot. If further adjustments to the boundaries are needed, this checkpoint shapefile can be read into a GIS for further editing. Then restart `BuildBorderGroup` with option `checkPointReStart = TRUE` so that it will read in the revised shapefile, skip the simplification steps and just create the necessary internal tables for the **micromapST** function call.

Most of the time, we will need to smooth our shapefile to some extent so that it will work well as a small map in linked micromap plots. To illustrate the full process to build a border group, we will build a U.S. map from its original shapefile on the Census Bureau website, then apply each of the steps in Section 5.3.3. Comparison of the resulting map image with the familiar caricature map of the U.S. used in **micromapST** will illustrate the impact of each step in the process.

5.4.1 Build a U.S. Name Table

The starting point in building a border group is to construct its Name Table. This table can be constructed as a CSV text file or an Excel spreadsheet. Each column provides information to help `BuildBorderGroup` build the internal Name Table for the border group. The Name Table provides the location IDs supported in the border group and a linkage to the collection of polygons for each area in the shapefile in order for **micromapST** to draw the associated areas. The Name Table will provide the full name, the most common abbreviation and numerical identifier for each area. These columns must be named “Name”, “Abbr” and “ID”, respectively, in the Name Table. No duplicates are allowed in any location ID column. Each row in the Name Table represents one area, in this example a state. The `rowNames`

TABLE 5.1 Minimum information needed in the Name Table: First 15 U.S. states.

Abbr	Name	ID
AL	Alabama	1
AK	Alaska	2
AZ	Arizona	4
AR	Arkansas	5
CA	California	6
CO	Colorado	8
CT	Connecticut	9
DE	Delaware	10
DC	District of Columbia	11
FL	Florida	12
GA	Georgia	13
HI	Hawaii	15
ID	Idaho	16
IL	Illinois	17
IN	Indiana	18

call parameter specifies which location ID will be used in the statistical data when calling **micromapST**. Puerto Rico was excluded from our Name Table, so it will be excluded from the resulting map when the `BuildBorderGroup` function links the Name Table with the shapefile polygons. Table 5.1 shows the basic Name Table for the first 15 of the 50 U.S. states.

U.S. states have standardized names, abbreviations (postal codes), and numeric identifiers (Federal Information Processing Standard (FIPS) codes (United States Census Bureau, 2022c)). However, this is not always the case for other countries or for non-standard U.S. geographies, so **micromapST** allows the user to specify the primary abbreviation “Abbr” and an alternate abbreviation “Alt_Abbr” for the areas if there are two sets of commonly-acceptable abbreviations.

A more complicated issue is when the user only has a string name to identify each area, e.g., names of NCI cancer registries. In this case, **micromapST** can perform a wildcard match of a field in the Name Table, provided in the optional “Alias” column, against the user-provided area name. The user can request the wildcard matching by specifying `rowNames = "alias"`. The Alias column must be in the initial Name Table and must link to a single valid area. This is a very unusual situation and the authors do not expect it to be used by average users.

If the areas are organized by regions, such as Census regions comprised of U. S. states, then the “regID” and optional “regName” columns can be specified in the Name Table to associate the areas (e.g., states) with the larger regions. One option that uses these region definitions is `dataRegionsOnly = TRUE` in the **micromapST** function call. Including this option will cause only the regions with any data to be mapped.

For this U.S. border group, the states of Alaska and Hawaii need to be scaled and moved to a better location to reduce the overall size of the map. Alaska was scaled down to 30% of its original size and moved to just below California. It is still recognizable. Hawaii can be moved to just below New Mexico near Texas. The District of Columbia is too small to be seen in

the micromap, so it was moved to the right of its original position and enlarged to 400% of its original size. In the tests of the border group, Rhode Island did not show its internal colors very well and so for this demonstration was enlarged to 150% of its original size and moved slightly eastward. See Table 5.3 for the table entries corresponding to these changes. Rotation of areas is also possible but not needed in this example. The values specifying the degree of scaling, rotation and moving are included in the Name Table for areas in the border group that need to be modified. Areas that are unchanged will have these values set to NA in the table.

Having moved some areas, the user may have trouble recognizing them in their new positions. **micromapST** has a feature to add a short label for moved areas to help the user identify them. This information is provided in MapL, MapX, and MapY columns (the label, X and Y coordinates for the label, respectively) in the Name Table. Note that these labels only appear on the first of the multiple maps in the linked micromap plot. Labels should be used sparingly so as not to overly clutter the map display. Any area's row that does not require labeling or modification will have the values in those columns set to NA. Note that the location values used for the areas that are to be moved and for their labels are always in the units of the original shapefile projection, in most cases, longitude and latitude degrees. This feature was used on the U. S. States border group to scale DC, Rhode Island, Alaska and Hawaii and to shift DC, Alaska and Hawaii to a better location on the map.

The columns in the Name Table are shown in Table 5.2, with the final Name Table for the U.S. in Table 5.3.

TABLE 5.2 Contents of the Name Table

Column Name	Type (numeric/character)	Definition
Name	character	Full name of area
Abbr	character	Abbreviated name of area. Should be just a few letters
ID	numeric	A numerical code representing the area
Alt_Abbr	character	An alternate commonly acceptable abbreviated name of the area
Alias	character	A string used in a wildcard (*string*) match of the Location ID
Link	character	A string value to use to build a linkage between a row in the Name Table and the shapefile polygons for an area
regID	numeric or character	Code representing the region
regName	character	Name of region
Xoffset	numeric	Change in horizontal direction for a moved area (negative is left)
Yoffset	numeric	Change in vertical direction for a moved area (negative is down)
Scale	numeric	Relative size of modified area, where 1 = no change at all
Rotate	numeric	Amount to rotate the area (in degrees)
MapL	character	Label for modified area
MapX	numeric	X axis (horizontal) location for label
MapY	numeric	Y axis (vertical) location for label

TABLE 5.3 Final Name Table for the first 15 U.S. states.

Abbr	Name	ID	regID	regName	Xoffset	Yoffset	Scale	Rotate	MapL	MapX	MapY
AL	Alabama	1	2	South	NA	NA	NA	NA	NA	NA	NA
AK	Alaska	2	0	Offshore	36.5	-36.00	0.3	0	AK	-114.8	26.12
AZ	Arizona	4	4	West	NA	NA	NA	NA	NA	NA	NA
AR	Arkansas	5	2	South	NA	NA	NA	NA	NA	NA	NA
CA	California	6	4	West	NA	NA	NA	NA	NA	NA	NA
CO	Colorado	8	4	West	NA	NA	NA	NA	NA	NA	NA
CT	Connecticut	9	1	Northeast	NA	NA	NA	NA	NA	NA	NA
DE	Delaware	10	2	South	NA	NA	NA	NA	NA	NA	NA
DC	District of Columbia	11	2	South	3.8	-2.15	4.0	0	DC	-71.5	35.50
FL	Florida	12	2	South	NA	NA	NA	NA	NA	NA	NA
GA	Georgia	13	2	South	NA	NA	NA	NA	NA	NA	NA
HI	Hawaii	15	0	Offshore	52.0	6.00	1.0	0	HI	-101.2	25.63
ID	Idaho	16	4	West	NA	NA	NA	NA	NA	NA	NA
IL	Illinois	17	3	Midwest	NA	NA	NA	NA	NA	NA	NA
IN	Indiana	18	3	Midwest	NA	NA	NA	NA	NA	NA	NA

Features in the Name Table not used in this example are the additional location identifier types of alternate abbreviation (“alt_abbr”) and alias wildcard matching (“alias”).

When the Name Table is read in, its contents and columns are validated to ensure the resulting Name Table will function properly with **micromapST**. Any errors found are flagged and reported to the user.

5.4.2 Process the U.S. Shapefile

First, the user’s boundary file is read in using `sf:::st_read` which creates an `sf` data frame. This function will read many boundary file formats, but we use the commonly-available shapefile format (.SHP).

An `sf` data frame has information beyond a non-spatial data frame that is needed to process and plot the spatial object. These are organized as data frame columns and a geometry column with rows for each spatial entity. Included in the structure are:

- The Geometry Type
- Dimension: XY
- Bounding box: xmin: -179 ymin: 17 xmax: 179 ymax: 71
- CRS: NA
- Features - one per geometry include data and geometry.

The U.S. Census state boundary shapefile (st99_d00) was read in Section 5.3.2. We can inspect this shapefile and determine what variables will be available and if the shapefile variables need to be modified.

```
names(US_shp)
```

```
## [1] "AREA"          "PERIMETER"      "ST99_D00_"     "ST99_D00_I"    "STATE"
## [6] "NAME"          "LSAD"           "REGION"        "DIVISION"     "LSAD_TRANS"
## [11] "geometry"
```

```
US_shp_data <- sf:::st_drop_geometry(US_shp)
head(US_shp_data)
```

```
##   AREA PERIMETER ST99_D00_ ST99_D00_I STATE  NAME LSAD REGION DIVISION
## 1 2.713e+02  227.1714     2       1 02 Alaska  01     4     9
## 2 3.749e-03   0.3498     3       2 02 Alaska  01     4     9
## 3 1.499e-03   0.1535     4       3 02 Alaska  01     4     9
## 4 3.130e-02   0.7122     5       4 02 Alaska  01     4     9
## 5 8.893e-01   7.3603     6       5 02 Alaska  01     4     9
## 6 2.608e-02   0.8216     7       6 02 Alaska  01     4     9
##   LSAD_TRANS
## 1      <NA>
## 2      <NA>
## 3      <NA>
## 4      <NA>
## 5      <NA>
## 6      <NA>
```

This shapefile has 273 polygons, far more than the number of states, because most states

are made up of multiple land masses. For example, Alaska has 81 polygons and Hawaii has 27 polygons. By the end of this process, all areas with multiple polygons will be combined so there will be one element per area, possibly with multiple polygons in each element.

The shapefile has a variable “NAME” in the shapefile that can be used to link the polygons to the Name Table area rows. Alternatively, the “STATE” variable in the shapefile could be linked to “ID” in the Name Table (see Table 5.3). No additional Name Table column or shapefile variable needs to be added, but the `BuildBorderGroup` function call must contain `NameTableLink = "Name"` and `ShapeLinkName = "NAME"`, specifying the names of the linking variable in both sources, respectively.

The shapefile is expected to be in the standard ESRI shapefile structure (Environmental Systems Research Institute, Inc., 1998). The projection can be missing or should be specified as longitude/latitude, the projection most publicly available shapefiles use. All U.S. Census Bureau shapefiles are provided in longitude/latitude coordinates. If the shapefile’s projection is in longitude/latitude coordinates, the user can further specify the projection of the boundaries before they are converted to the **micromapST** format by using the `proj4` call parameter. This option was added because some states require that any maps of their state be presented in a particular projection. If no `proj4` parameter is present, the function will estimate an Albers equal area projection based on the centroid of the map with the extra latitudes set to $\frac{1}{4}$ of the height above and below the center latitude. This produces a good projection for use with linked micromaps and ensures that the land areas are not distorted.

Once the shapefile is read, the contents are validated using tools in several R packages: `sf` (Pebesma, 2018), `sp` (Bivand et al., 2013) and `spdep` (Bivand, 2022). This assures that polygons are complete, i.e., without gaps in their perimeter, shared boundaries do not contain gaps or overlaps, and areas (polygons) do not overlap. Any of these problems can cause processing errors later if not repaired at this point.

The shapefile is then passed through the `rmapshaper` R package (Teucher & Russell, 2022) to reduce its complexity to a specified percentage of the original size. We recommend the default 1.25% as a starting point, but this value may need to be adjusted to create a border group suitable for a particular application. `rmapshaper` is set to not allow any areas to be eliminated but it may reduce an area so much that its overall shape changes. For example, DC may be changed to a triangular area instead of maintaining its original rectangular shape. Check the area boundaries to make sure all the areas are still recognizable. If there are problems getting a good representation with the `BuildBorderGroup` function, simplify the shapefile with an external package (GIS program or MapShaper website). Then bring it back to the `BuildBorderGroup` function and set the `ReducePC` call parameter to 100 to keep the function from trying to simplify the boundaries again.

5.4.3 Link the Name Table and Shapefile

Now that the Name Table and shapefile are complete, the goal is to link the one unique row for each area in the Name Table to the (usually) multiple polygons per area in the shapefile. If the shapefile has a variable that matches the Name, Abbr or ID variable in the Name Table and accurately tags the polygons that belong to that area, all that needs to be done is specification of that variable in the `BuildBorderGroup` call, e.g., `ShapeLinkName = "NAME"` and `NameTableLink = "Name"`, to allow the function to compare and link the polygons to the Name Table entry for the area. If the shapefile has a variable other than Name, Abbr and ID that accurately identifies each area, then the unique values of this variable can be copied to the Name Table, preferably with the name “Link”. The actual linking of the Name

Table and the shapefile polygons does not occur until the shapefile is processed and all the polygons for a single area are gathered under one element in the SpatialPolygons structure.

5.4.4 Apply **BuildBorderGroup** to Create Internal Data Frames

When the Name Table is finished, the shapefile data are simplified and the two are linked, the information used to create the border group dataset consists of 6 data frame tables:

- areaParms - Containing specific operational information related to this border group.
- areaNamesAbbrsIDs - The Name Table for the border group.
- areaVisBorders - The boundary data for each area in the maps as a simple data frame containing the x and y coordinates of a point, a key for the associated area, and a flag as to whether the polygon is a hole. This information is needed so that the “hole” in a larger area (the “donut”) is properly shaded.
- RegVisBorders - The boundary data for each “region” defined in the map in the same format as the areaVisBorders data frame. In the U. S. States border groups, the information was provided to group each area/state into its assigned U. S. Census region. The region boundary data provides an outline for all of the areas/states within that region. For more information on how to sub-divide the areas drawn in a linked micromap by region, see the **micromapST** documentation.
- L2VisBorders - The boundary data in the same format as the areaVisBorders data frame for each “level 2” region in the map. This is an optional set of boundaries that can be used to accent an area or set of areas by using a thicker outline.
- L3VisBorders - The boundary data to outline the entire geographic space of all of the areas included in the border group.

The L2, Reg, and L3 VisBorders options are used to draw an enhanced border around the areas, using increasingly thicker outlines.

The areaParms data frame provides *micromapST* with information specific to this border group. On the **BuildBorderGroup** function call, the caller can specify the header labels for the map and ID columns of the linked micromap plot by using the MapHdr and IDHdr Header titles. LabelCex may be added to the function call to specify a font size multiplier for the text labels of any moved areas on the first map. In addition, two additional parameters can be provided here by the caller: MapMinH and MapMaxH, which specify the minimum and maximum allowed height of a map, respectively, when **micromapST** calculates the available space to draw the linked micromap.

The plot space calculated for each micromap depends on the number of perceptual groups of areas and the number of glyph columns to be plotted on the output page. This plot space typically works out to be rectangular and wider than it is high. If a particular border group requires a larger space than the package calculates, then the user can increase the maximum map height (MapMaxH) from its default of 1.5 to 2 inches or more. This may be necessary when a map is higher than it is wide, the reverse of the plot space orientation. For example, Massachusetts fits well within the default map plot space whereas California does not because of its north-south orientation. Similarly, MapMinH can be used to force the map plot area to be at least a specified height, such as for California. This may be necessary to allow the boundaries and colors on the micromaps to be visible in the small plot space. It may take some trial and error to balance map visibility and the wish to fit as many micromaps on a page as possible.

5.4.5 Compare U.S. Results to Default Caricature Map

Now let's apply these steps to the original U.S. state boundaries and compare the results to the default U.S. state map in **micromapST**. The output from the `BuildBorderGroup` function extends over several pages and is not shown here.

```
library(micromapST)

shape_file_dir_US <- "data/Ch4b"
name_table_dir_US <- "data/Ch4b"
output_dir_US <- "output/Ch4b/US"

BuildBorderGroup(
  ShapeFile = "st99_d00", # Base filename of shapefile without sf extension
  ShapeFileDir = shape_file_dir_US, # Directory containing the shapefile
  ShapeLinkName = "NAME", # Variable name containing link to NameTable
  NameTableDir = name_table_dir_US, # Directory containing the NameTable
  NameTableFile = "USNameTable.xlsx", # NameTable file
  NameTableLink = "Name", # The column in the NameTable to use to link the sf to the Name Table
  BorderGroupName = "USstBG", # Name of the Border Group (BordGrp)
  BorderGroupDir = output_dir_US, # Output directory for resulting files
  MapHdr = c("", "States"),
  MapMinH = 0.7, # Minimum Height for micromap drawing (inches)
  MapMaxH = 2, # Maximum Height for micromap drawing (inches)
  IDHdr = "States", # 1 of 2 header lines for ID Glyph column (Max 12 chars)
  ReducePC = 0.1, # reduce vertices to 0.1% of original number
  debug = 640 # generate 4 intermediate PNG map plots
)
```

After some trial and error, we chose to reduce the polygon vertices to 0.1% of the original number. Including the `debug = 640` option within `BuildBorderGroup` generates intermediate map files in PNG format as the multiple steps of simplification are applied. These are shown in the next figure, omitting the original raw image.

The progression from the original Census shapefile (Figure 5.1) to the final image created by `BuildBorderGroup` is shown in Figure 5.2. The color shading is random to illustrate how a shaded map might look. The first intermediate map (A) is smoothed from the original but still has the problem of a very small continental U.S. The second step (B), moving and scaling Alaska and Hawaii, improved this. Except for a difference in map projections used, the final map (C) looks quite similar to the caricature map used by **micromapST** (D) and would be a suitable substitute for it. This illustrates the trade-off: a caricature-type map might be preferred for a particular application but `BuildBorderGroup` can provide a reasonable alternative without the effort to hand draw a better map.

Some readers may prefer a less smoothed map or may need guidance as to the degree of smoothing needed for their own shapefile. To this end, we repeated the `BuildBorderGroup` call with various levels of smoothing. The function call is exactly as shown above except that only the final map image is saved (`debug = 1152`) and `ReducePC` was changed to specify smoothing levels from none (`ReducePC = 100`) to 99.9% reduction (`ReducePC = 0.1`).

We recommend starting with a reduction to 2.5% of the original vertices. For our U.S. state map, reduction to this level produces a map (B) that appears barely different from the

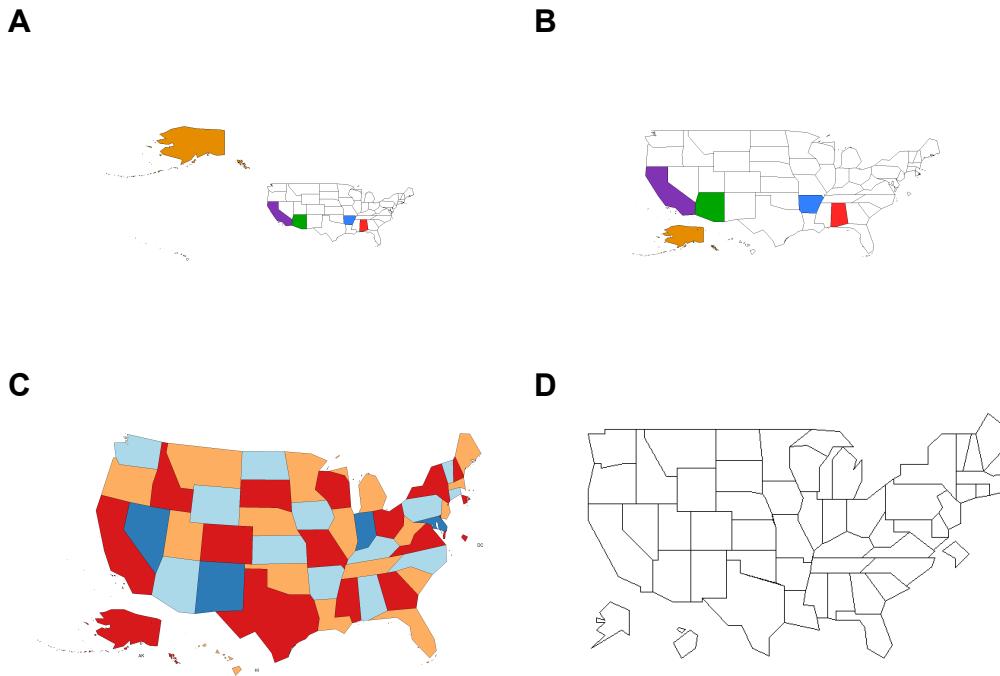


FIGURE 5.2 U.S. states maps: Simplified (A), after moving and scaling (B), final (C) and micromapST default map (D).

original (A). Changes are evident in the maps as the vertices are further reduced to 0.1% of the original (F), especially notable along coastlines and rivers that form state boundaries. The border group created by `BuildBorderGroup` can be used for subsequent linked micromaps by specifying `bordGrp = "filename"` in the `micromapST` call, where “filename” is defined by `BorderGroupName = "filename"` in `BuildBorderGroup` call. This will be illustrated in the next sections.

5.5 Linked Micromap Plots for a Large Number of Areas

Sometimes the geographic area to be displayed has too many sub-areas to fit easily on a single-page linked micromap plot. What can be done? The output can be directed to a device that allows a longer display, such as a PDF or PNG file, which can be printed on multiple sheets of paper or displayed on a computer monitor, using scrolling to view one section of the plot at a time. Neither of these solutions is ideal, since one purpose of the linked micromap design is to facilitate pattern recognition across all of the geographic areas at once. An alternative approach, if possible, is to aggregate the original areas into a smaller number of meaningful units. We illustrate each of these approaches in this section.

The maximum number of areas that can be easily visualized in a linked micromap plot depends on the size of the graphic space available. Recommended limits to ensure visibility and ease of use are about 75 areas for an 8.5" x 11" page (letter size), 100 for an 8.5" x 14"

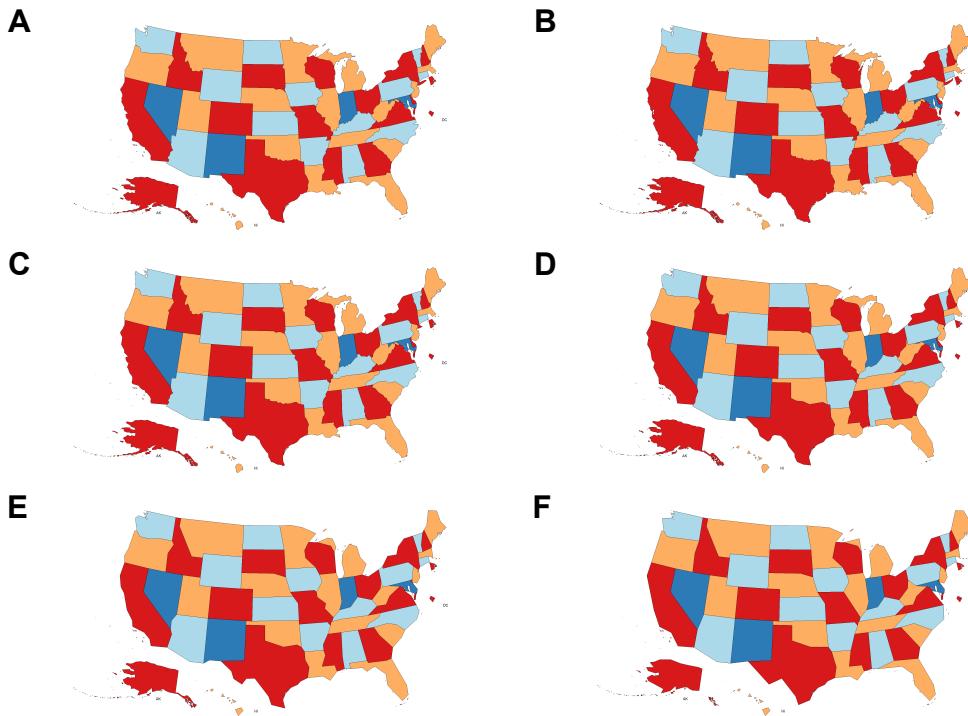


FIGURE 5.3 U.S. states maps smoothed to retain varying percents of vertices from the original shapefile: 100% (No smoothing) (A), 2.5% (B), 1% (C), 0.5% (D), 0.2% (E), 0.1% (F).

page (legal size) and as many as 240 on an 11" x 17" page (tabloid size). By using PDF files, a long linked micromap plot can be displayed, zooming in to see details and scrolling up and down to view the plot in sections.

5.5.1 Creating Long Linked Micromap Plots

We will use cancer data from the state of Kentucky, with 120 counties, to illustrate a long linked micromap plot display. Kentucky had the highest invasive cervical cancer rate among women in the U.S. for the period 2014-2018 (University of Kentucky, Markey Cancer Center, Kentucky Cancer Consortium, Kentucky Cancer Program, 2022). There is a test to detect this cancer at an early stage and a vaccine to prevent infections that can lead to cancer later (National Cancer Institute, 2021) but implementation of preventive measures is hampered by high poverty and low educational attainment in many parts of the state. In an effort to reduce these high cancer rates, the state public health department examines geospatial patterns within the state by county and by county aggregations when data are sparse. A linked micromap plot of cervical cancer incidence rates can display these patterns and help the state target their scarce resources to where they are needed most, e.g., implementing preventive measures in the high-rate eastern counties. Examining their cancer patterns in this way, the state health department has made some progress recently toward reducing their cervical cancer rates.

After reading the Kentucky county shapefile, we need to call `BuildBorderGroup` to create the

internal tables needed for **micromapST**. We will request some smoothing of the boundaries and plot the resulting map.

```

library(micromapST)
library(readxl)
library(sf)

shape_file_dir_KY <- "data/Ch4b" # Base Directory
name_table_dir_KY <- "data/Ch4b"
output_dir_KY <- "output/Ch4b/KY"

# Read in Kentucky 2000 census county boundaries shapefile.
KY_shp_file <- "co21_d00" # Census 2000 files Kentucky counties.
KY_sf <- st_read(shape_file_dir_KY, KY_shp_file)
KY_sf_data <- st_drop_geometry(KY_sf)
sf:::st_crs(KY_sf) <- st_crs("+proj=lonlat")

# shapefile has "NAME" field in data.frame that can be used to match
# "Name" in the Name Table
BuildBorderGroup(
  ShapeFile = KY_sf,
  ShapeFileDir = output_dir_KY,
  ShapeLinkName = "NAME", # name in shapefile to link to Name Table
  NameTableDir = name_table_dir_KY, # directory where all files are
  NameTableFile = "KY_Co_To_ADD.xlsx", # Name Table file name
  NameTableLink = "Name", # column name in Name Table for linking
  BorderGroupName = "KY_countiesBG", # name of new border group
  BorderGroupDir = output_dir_KY,
  MapHdr = c("", "KY counties"), # title over map column
  IDHdr = c("County"), # title over ID column
  ReducePC = 0.25, # percentage of vertices to keep after simplifying
  debug = 640 # generate 4 intermediate PNG map plots
)

```

```

load(paste0(output_dir_KY, "/KY_countiesBG.rda")) # this loads new border group

par(mar = c(0, 0, 0, 0))
Plot_Vis(VisB = areaVisBorders, xLwd = 1)

```

Note that the smoothing did not eliminate the small non-contiguous segment of Fulton County at the far western end of the state. This is not an island but land separated from the rest of the county by the Kentucky Bend of the Mississippi River.

Now we set up the linked micromap plot to display the county rates with their 95% confidence intervals, sorted in descending order. The data are invasive cervical cancer incidence rates, age-adjusted to the 2000 standard million population, downloaded from the Kentucky Cancer Registry website (<https://www.kcr.uky.edu/>) In order to have sufficient vertical space for the micromaps, we specify the figure dimensions as 7" by 15".

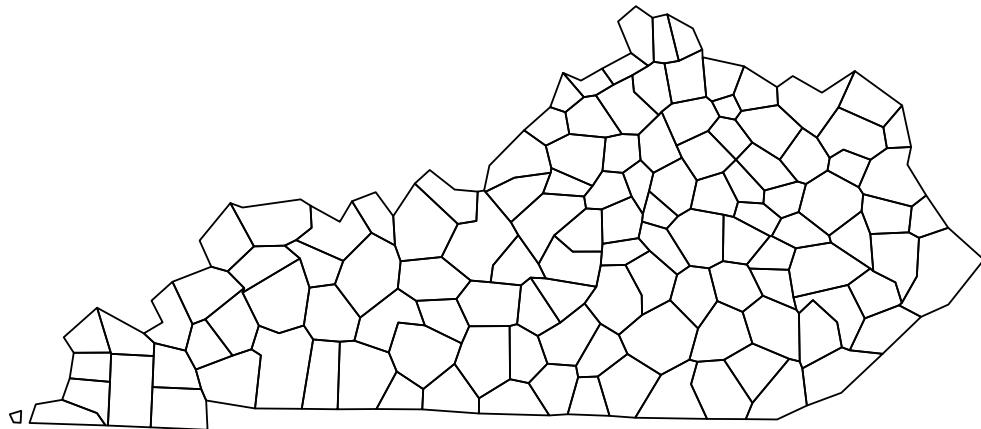


FIGURE 5.4 Kentucky counties, smoothed by `BuildBorderGroup`.

```
# Read in KY cervical cancer incidence rate data (skip 1st 2 header rows)
KY_cervca <- read.csv(
  "data/Ch4b/Invasive-Cancer-Incidence-Rates-by-County-in-Kentucky-Cervix-Uteri-2014-2018.csv",
  skip = 2
)
head(KY_cervca)

##   county_name population cases adj_rate confint_lower confint_upper
## 1      Adair     48654    NA      NA          NA          NA
## 2      Allen     52469     7    13.8        5.4       28.7
## 3  Anderson     56803    NA      NA          NA          NA
## 4    Ballard     20328    NA      NA          NA          NA
## 5    Barren    112550     9     7.2        3.2       14.1
## 6      Bath     30973    NA      NA          NA          NA

panel_desc_KY <- data.frame(
  type = c("mapcum", "id", "dotconf"),
  lab1 = c("", "", "Adj. Rate"),
  lab2 = c("", "", "per 100,000"),
  col1 = c(NA, NA, "adj_rate"),
  col2 = c(NA, NA, "confint_lower"),
  col3 = c(NA, NA, "confint_upper"),
  refVals = c(NA, NA, 7.7),
  refTexts = c(NA, NA, "US rate")
)

micromapST(
  statsDFrame = KY_cervca,
  panelDesc = panel_desc_KY,
```

```

sortVar = c("adj_rate"),
ascend = FALSE,
title = "Cervical Cancer Incidence Rates for Kentucky Counties, 2014-2018",
rowNames = "full",
rowNamesCol = "county_name",
bordGrp = "KY_countiesBG",
bordDir = output_dir_KY,
plotNames = "full"
)

## ***0432 PANEL panelLayout - Info:The calculated GrpRow Height is: 0.503125 inches. The mininum size limit is:

```

Clearly this tests the limits of map visibility, but it does work, although the package warns that the calculated panel height is about 0.5 inches, less than the minimum recommended height of 1 inch. However, there are more serious problems with the data than with the small map images. Many of the counties had unstable rates as evidenced by very wide confidence intervals. Furthermore, nearly half of the counties had fewer than five cases and so their data were suppressed (or shown as 0) on the plot, making it difficult to discern any geospatial pattern.

5.5.2 Aggregating counties to Area Development Districts in Kentucky

When the data are unstable due to small numbers, aggregation over some dimension of the data can help. The original Kentucky cancer data were already aggregated over time (5 years) so perhaps we can aggregate over space. Geographic aggregation will also help by reducing the number of areas to display on the micromap. Many political entities have defined aggregations of smaller areas that are preferred for reports of their data by geographic area.

Kentucky defined Area Development Districts (ADDs) (Kentucky Legislative Research Commission, 1972; Northern Kentucky Area Development District, 2024) 50 years ago to link federal and state resources to the needs of local Kentucky communities. These 15 districts foster collaboration across local governments in the state, aid in regional planning and resource sharing and now are used as the geographic units of choice in many governmental reports (Huang et al., 2018). We will use Kentucky ADDs to illustrate how one can create aggregated geographic boundaries which then can be used in **micromapST**.

To build a border group of the ADDs, the first step is to aggregate the county polygons of the state into ADD polygons outside of the **micromapST** package. Then this new shapefile (or SpatialPolygons), along with a Name Table, can be converted into a border group for use by the package.

The state of Kentucky's website provides the ADD name for each county (Northern Kentucky Area Development District, 2024). The U.S. 2000 Census Bureau County boundary data for Kentucky provides the boundaries and the shapefile provides the county ID ("COUNTY") and name ("NAME").

The code below is one example of how one can create a SpatialPolygonsDataFrame for the Kentucky ADDs.

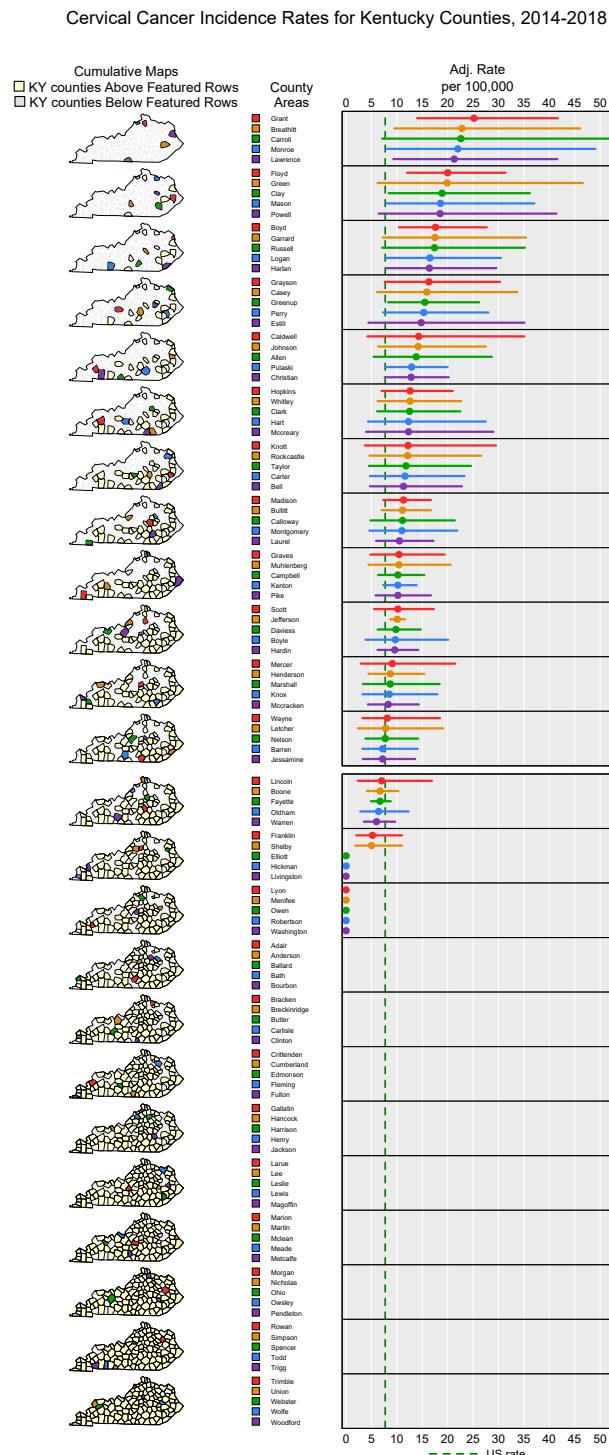


FIGURE 5.5 Invasive cervical cancer incidence rates in Kentucky Counties, 2014-2018, age-adjusted to 2000 standard million population.

```
# Read in *.xlsx table of counties and ADDs and make it a data frame
CoToADD <- read_xlsx("data/Ch4b/KY_Co_To_ADD.xlsx")
CoToADD <- as.data.frame(CoToADD, stringsAsFactors = FALSE)
head(CoToADD, n = 6L)
```

	ADD	Name	ID
## 1	Lake Cumberland	Adair	72
## 2	Barren River	Allen	8
## 3	Bluegrass	Anderson	16
## 4	Purchase	Ballard	115
## 5	Barren River	Barren	7
## 6	Gateway	Bath	52

This table has 120 rows, one per county, and provides the ADD name for each county. We can see from this short example that Allen and Barren counties are in an ADD called Barren River.

The first step is to aggregate boundaries for counties into the larger ADDs based on the county-to-ADD translation table. County names will be matched between the files (“NAME” in the shapefile, “Name” in the above table) to identify the ADD to which each county belongs. Then the county polygons are merged into ADD polygons and the new ADD shapefile is written out.

```
shape_file_dir_KYADD <- "data/Ch4b" # Base Directory
name_table_dir_KYADD <- "data/Ch4b"
output_dir_KYADD <- "output/Ch4b/KYADD"

# Match on county name, then merge polygons based on ADD
Merge_inx <- match(KY_sf$NAME, CoToADD$name)
Merge_list <- CoToADD$ADD[Merge_inx] # Convert county list into ADD list for merge

# define aggregate function for merge
aggFun <- function(z) {
  ifelse(methods::is(z, "character"), z[1], sum(z))
}

KY_ADD_sf <- aggregate(KY_sf, by = list(Merge_list), FUN = aggFun)
names(KY_ADD_sf)[1] <- "addName" # rename first column - modified by aggregate function
row.names(KY_ADD_sf) <- KY_ADD_sf$addName # use ADD names for row.names of each geometry (area)
KY_ADD_sf2 <- st_cast(KY_ADD_sf, "MULTIPOLYGON") # make geometries uniform

# save sf data.frame as shapefile
sf::st_write(
  obj = KY_ADD_sf,
  dsn = output_dir_KYADD,
  layer = "KYADD",
  driver = "ESRI Shapefile",
  delete_layer = TRUE # delete old file
)
```

```
## Deleting layer `KYADD' using driver `ESRI Shapefile'
## Writing layer `KYADD' to data source `output/Ch4b/KYADD' using driver `ESRI Shapefile'
## Writing 15 features with 10 fields and geometry type Unknown (any).
```

5.5.3 Building the New Border Group for the Aggregated Kentucky ADDs

Now we can build a new border group for use in **micromapST**. Since there were no modifications (e.g., scaling or moving) of the boundaries, the basic Name Table is all that is needed:

```
temp_name_table <- as.data.frame(read_xlsx("data/Ch4b/KYADD_NameTable-Basic.xlsx"))
head(temp_name_table, n = 6L)
```

	Name	Abbr	ID	Alias
## 1	Bluegrass	BG	2101	Blueg
## 2	Barren River	BR	2102	Barren
## 3	Big Sandy	BS	2103	Sandy
## 4	Buffalo Trace	BT	2104	Buffalo
## 5	Cumberland Valley	CV	2105	Valley
## 6	Fivco	FI	2106	Fiv

Next, `BuildBorderGroup` is called using this Name Table and the ADD border group created in the previous section (5.5.2). Again, the output from the `BuildBorderGroup` function extends over several pages and is not shown here.

```
# BuildBorderGroup function call

BuildBorderGroup(
  ShapeFile = "KYADD",
  ShapeFileDir = output_dir_KYADD,
  ShapeLinkName = "addName", # name in shapefile to link to Name Table
  NameTableDir = name_table_dir_KYADD, # directory where all files are
  NameTableFile = "KYADD_NameTable-Basic.xlsx", # Name Table file name
  NameTableLink = "Name", # column name in Name Table for linking
  BorderGroupName = "KYADD2BG", # name of border group to use
  BorderGroupDir = output_dir_KYADD,
  MapHdr = c("", "KY ADD"), # title over map column
  IDHdr = c("ADD"), # title over ID column
  ReducePC = 0.75, # percentage of vertices to keep after simplifying
  debug = 1152 # generate final PNG map plot only
)
```

Save and print the resulting image:

The original Kentucky ADD map didn't require any scaling or moving, so probably would be adequate, but this smoothed version will give a better color visualization in a very small linked micromap plot.

Now we can create the same linked micromap plot as in Figure 5.5. `panelDesc` is exactly the same as before but is shown here for completeness:

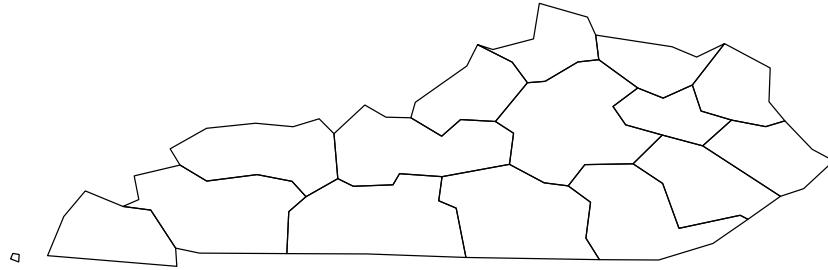


FIGURE 5.6 Kentucky smoothed Area Development Districts.

```
# Read in KY cervical cancer incidence rate data (skip 1st 2 header rows)
KYADD_cervca <- read.csv(
  "data/Ch4b/Invasive-Cancer-Incidence-Rates-by-ADD-in-Kentucky-Cervix-Uteri-2014-2018.csv",
  skip = 2
)
head(KYADD_cervca)

##          ADD_name population cases adj_rate confint_lower confint_upper
## 1           Fivco     338028    51   15.3        11.2       20.4
## 2   Kentucky River    270621    40   14.3        10.0       19.6
## 3      Big Sandy    365367    52   13.1         9.7       17.5
## 4  Lake Cumberland   526773    65   12.2         9.3       15.8
## 5  Cumberland Valley  590997    73   12.2         9.5       15.4
## 6    Buffalo Trace    141150    15   11.4         6.2       19.0

# panelDesc is the same as before; repeated here for clarity
panel_desc_KY <- data.frame(
  type = c("mapcum", "id", "dotconf"),
  lab1 = c("", "", "Adj. Rate"),
  lab2 = c("", "", "per 100,000"),
  col1 = c(NA, NA, "adj_rate"),
  col2 = c(NA, NA, "confint_lower"),
  col3 = c(NA, NA, "confint_upper"),
  refVals = c(NA, NA, 7.7),
  refTexts = c(NA, NA, "US rate")
)

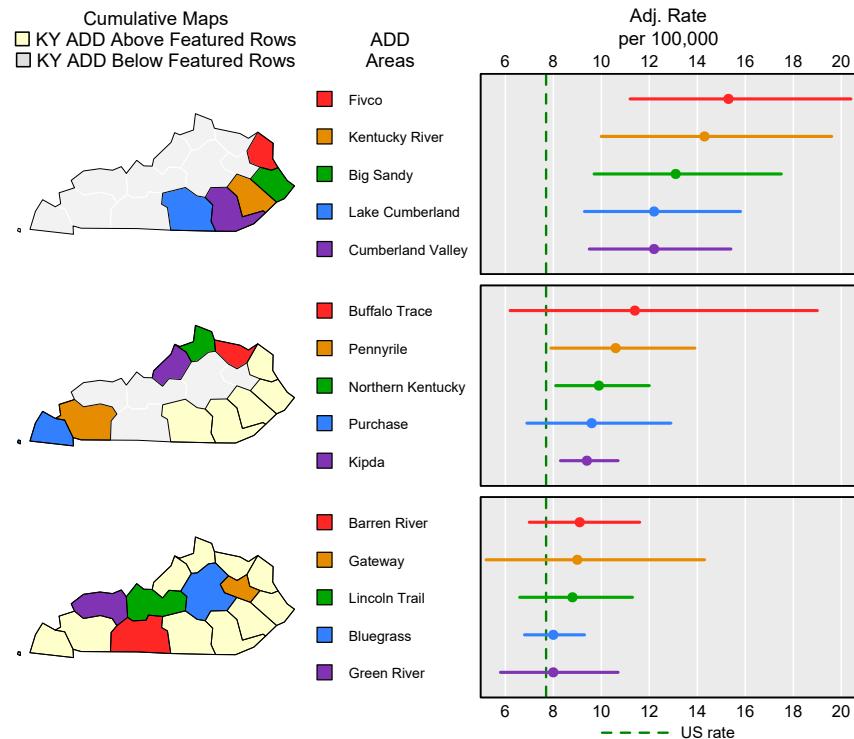
micromapST(
  statsDFrame = KYADD_cervca,
```

```

panelDesc = panel_desc_KY,
sortVar = "adj_rate",
ascend = FALSE,
title = "Cervical Cancer Incidence Rates for Kentucky ADDs, 2014-2018",
rowNames = "full",
rowNamesCol = "ADD_name",
bordGrp = "KYADD2BG",
bordDir = output_dir_KYADD,
plotNames = "full"
)

```

Cervical Cancer Incidence Rates for Kentucky ADDs, 2014-2018

**FIGURE 5.7** Invasive cervical cancer incidence rates in Kentucky Area Development Districts, 2014-2018, age-adjusted to 2000 standard million population.

This is much clearer. All ADDs have sufficient data to display. Eight of the highest 10 rates have confidence intervals that do not include the U.S. rate, while all five of the lowest rates

do include it. Buffalo Trace's confidence is very wide because it has the lowest population of any ADD. We also can see a clear geospatial pattern, with the highest rates occurring in what is known to be the poorest part of the state.

5.6 Summary and Further Reading

The **micromapST** package is quite flexible and can generate many types of graphics to explore and communicate data. However, its use was at first limited to U.S states. More recent versions included other built-in boundary files, including counties within a few U.S. states and other countries, but the latest version allows the user to create linked micromap plots using their own boundary file. The new `BuildBorderGroup` function has options to read in shapefiles directly and link them to a user-provided data table. Because the linked micromaps are very small, this function can smooth the original borders to better display the interior shading at that scale. This new capability widens the scope of linked micromap plots that can be produced by **micromapST**.

Further information about shapefiles is available at (Environmental Systems Research Institute, Inc., 1998) and more details about some of the spatial functions used to process them is available in (Pebesma & Bivand, 2023).

References

- Bivand, R. S. (2022). R Packages for Analyzing Spatial Data: A Comparative Case Study with Areal Data [<https://doi.org/10.1111/gean.12319>]. *Geographical Analysis*, 54(3), 488–518.
- Bivand, R. S., Pebesma, E., & Gómez-Rubio, V. (2013). *Applied Spatial Data Analysis with R, Second Edition* [<https://doi.org/10.1007/978-1-4614-7618-4> and <https://asdar-book.org/>]. Springer, New York, NY.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Environmental Systems Research Institute, Inc. (1998). ESRI Shapefile Technical Description [White Paper, <https://www.esri.com/Library/Whitepapers/Pdfs/Shapefile.pdf>].
- Huang, B., Pollock, E., Zhu, L., Athens, J. P., Gangnon, R., Feuer, E. J., & Tucker, T. C. (2018). Ranking Composite Cancer Burden Indices for Geographic Regions: Point and Interval Estimates [<https://doi.org/10.1007/s10552-018-1000-9>]. *Cancer Causes & Control*, 29(2), 279–287.
- Kentucky Legislative Research Commission. (1972). 147A.050 Area Development Districts Created [Web Page, <https://apps.legislature.ky.gov/law/statutes/statute.aspx?id=1652> (accessed September 27, 2022)].
- Monmonier, M. (1993). *Mapping It Out: Expository Cartography for the Humanities and Social Sciences*. University of Chicago Press, Chicago, IL.

- National Cancer Institute. (2021). Cervical Cancer Prevention (PDQ®)–Patient Version [Web Page, <https://www.cancer.gov/types/cervical/patient/cervical-prevention-pdq> (accessed October 3, 2022)].
- National Cancer Institute, Surveillance, Epidemiology, and End Results Program. (2022a). About the SEER Program [Web Page, <https://seer.cancer.gov/about/> (accessed September 8, 2022)].
- Northern Kentucky Area Development District. (2024). About Us [Web Page, <https://www.nkadd.org/about-us/> (accessed January 2, 2024)].
- Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data [<https://doi.org/10.32614/RJ-2018-009>]. *The R Journal*, 10(1), 439–446.
- Pebesma, E., & Bivand, R. S. (2023). *Spatial Data Science: With Applications in R* [<https://doi.org/10.1201/9780429459016> and <https://r-spatial.org/book/>]. Chapman & Hall / CRC, Boca Raton, FL.
- Teucher, A., & Russell, K. (2022). *rmapshaper: Client for 'mapshaper' for 'Geospatial' Operations* [R package version 0.4.6 (<http://CRAN.R-project.org/package=rmapshaper>)].
- United States Census Bureau. (2021). Cartographic Boundary Files - Shapefile [Web Page, <https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.2000.html> (accessed September 8, 2022)].
- United States Census Bureau. (2022c). American National Standards Institute (ANSI) and Federal Information Processing Series (FIPS) Codes [Web Page, <https://www.census.gov/library/reference/code-lists/ansi.html> (accessed September 8, 2022)].
- University of Kentucky, Markey Cancer Center, Kentucky Cancer Consortium, Kentucky Cancer Program. (2022). Kentucky Cancer Registry: The Population-Based Central Cancer Registry for the Commonwealth of Kentucky [Web Page, <https://www.kcr.uky.edu/> (accessed September 27, 2022)].

6

Adding New Plot Types to Linked Micromap Plots (such as Arrow Plots, Line Charts, and Scatterplots)

SARAH SCHWARTZ, JÜRGEN SYMANZIK

The primary graph types in the statistical panels that are supported by the micromap R package are dot plots, dot plots with confidence intervals, bar charts, and bar charts with confidence intervals. The reader will learn how to add new plot types to the repertoire of plot types, starting with arrow plots that show differences of two variables and scatterplots for two quantitative variables.

6.1 Introduction

As a reminder, see Chapter 1 for general style requirements for our *Micromap Plots in R* book. In particular, please do the following:

- Introduce meaningful labels for the sections, figures, and tables in your chapter.
 - Create index entries for all R packages (such as the **micromap** R package) and for all datasets (such as the *USstates* and *edPov* datasets) that are used in your chapter.
 - Include references for R packages and publications related to your chapter, such as for the **micromap** (Payton & Olsen, 2015) and **micromapST** (Carr & Pearson Jr., 2015) R packages and some micromap articles, book chapters, and books (Carr, 2001; Carr & Pickle, 2010; Symanzik & Carr, 2008).
 - Also create index entries for main topics such as linked micromap plots, conditioned choropleth maps, perceptual group, color blindness,, and quantile-quantile plot.
-

6.2 Outline

- overview of graph types that are part of the micromap R package

panel.types (required) - a vector specifying the panels of the plot. Note: each panel.type (e.g. `map`, `labels`, `dot_cl`, etc.) is the name of a function that will be called to create that panel. Therefore a user can create a new panel type (e.g. `new.graph.type`) and the `mmpplot` function will automatically go look for and call that function just by changing the entry here. See the section [Creating a New Panel Type](#).

panel.types from Chapter 2: dot_legend, labels, dot, map, box_summary, dot_cl.

also bar_cl from vignette

Existing build functions from <https://github.com/USEPA/micromap/blob/master/R/PanelBuilding.r>

labels_build, bar_build, bar_cl_build, box_summary_build, dot_build, dot_cl_build, ranks_build.

Figure 6.1 shows one linked micromap plot with the all pre-defined plot types of the **micromap** R package (Payton & Olsen, 2015). This figure is an extension of Figures 2.15 and 2.16 from Chapter 2 that previously dealt with the WV_Watershed dataset.

There are 41 variables in the underlying data frame. Similar to Chapter 2 we will focus on the specific conductance variables in the following linked micromap plots.

Variable names beginning with an uppercase letter, e.g., Cond_med, Cond_LCB95, Cond_UCB95, are the population estimates, representing the median, the lower 95% confidence bound, and the upper 95% confidence bound of a variable (here, specific conductance) in each of the 25 watersheds, respectively. These variables will be used for the construction of dotplots, dotplots with confidence bounds, bar charts, and bar charts with confidence bounds.

Variable names starting with a lowercase letter, e.g., cond_min, condq1, cond_med1, cond_q3, cond_max, are the descriptive statistics, representing the minimum, first quartile, median, third quartile, and maximum of a variable (here, specific conductance) in each of the 25 watersheds, respectively. These will be used for the construction of boxplots.

The main purpose of the linked micromap plot shown in Figure 6.1 is to introduce all pre-defined panel and plot types of the **micromap** R package. Therefore, we have omitted a meaningful header for all columns in this figure and rather titled each column with the type of the information shown in that column and the corresponding entry for the panel.types list. We have also further shortened the abbreviations for the subregions shown in this linked micromap plot. Readers interested in the full names of these subregions are referred to Section 2.4.4. Also, from a practical perspective, it makes little sense to include dotplots, dotplots with confidence bounds, bar charts, and bar charts with confidence bounds of the same statistical variable in one figure. Typically, just a dotplot with confidence bounds or a bar chart with confidence bounds would have been sufficient to show the underlying statistical data.

```
library(micromap)

wv_watershed_sf <- sf::st_read(
  dsn = "data/WV_Watershed/RandomWatershed2_stats_smooth.shp",
  quiet = TRUE
)

wv_watershed_sf$short_name <- as.factor(wv_watershed_sf$Random_Wat)

wv_watershed_short <- list(
  "SPotom" = "South Branch Potomac",
  "NPotom" = "North Branch Potomac",
  "Cacapo" = "Cacapon/Shenandoah Hardy",
  "Potom" = "Potomac Direct Drains/Shenandoah Jefferson",
```

```
"UNew" = "Upper New/James",
"Tygart" = "Tygart Valley",
"WFork" = "West Fork",
"Monong" = "Monongahela/Dunkard",
"Cheat" = "Cheat/Youghiogheny",
"UOhio" = "Upper Ohio North/Upper Ohio South",
"MOhioN" = "Middle Ohio North",
"MOhioS" = "Middle Ohio South",
"LKana" = "Little Kanawha",
"Greenb" = "Greenbrier",
"LNew" = "Lower New",
"Gauley" = "Gauley",
"UKana" = "Upper Kanawha",
"Elk" = "Elk",
"LKana" = "Lower Kanawha",
"Coal" = "Coal",
"UGuya" = "Upper Guyandotte",
"LGuya" = "Lower Guyandotte",
"TugFork" = "Tug Fork",
"BigSan" = "Big Sandy/Lower Ohio",
"Twelvep" = "Twelvepole"
)

levels(wv_watershed_sf$short_name) <- wv_watershed_short

mmpplot(
  map.data = wv_watershed_sf,
  panel.types = c(
    "dot_legend",
    "ranks",
    "labels",
    "dot",
    "dot_cl",
    "bar",
    "bar_cl",
    "box_summary",
    "map"
  ),
  panel.data = list(
    NA,
    NA,
    "short_name",
    "Cond_med",
    list("Cond_med", "Cond_LCB95", "Cond_UCB95"),
    "Cond_med",
    list("Cond_med", "Cond_LCB95", "Cond_UCB95"),
    list("cond_min", "condq1", "cond_med1", "condq3", "cond_max"),
    NA
  ),
)
```

```
ord.by = "Cond_med",
rev.order = TRUE,
grouping = 5,
plot.pGrp.spacing = 1.2,
colors = RColorBrewer::brewer.pal(n = 5, name = "YlGnBu")[5:1],
panel.att = list(
  list(
    1,
    header = "Symbol\n(dot_legend)",
    panel.width = 5.5,
    point.type = 15,
    point.size = 0.9,
    point.border = TRUE,
    xaxis.labels.size = 0.8,
    right.margin = 0.4
  ),
  list(
    2,
    header = "Rank\n(ranks)",
    panel.width = 0.8,
    align = "center",
    text.size = 0.2,
    xaxis.labels.size = 0.8,
    right.margin = 0.4
  ),
  list(
    3,
    header = "Name\n(labels)",
    panel.width = 0.8,
    align = "left",
    text.size = 0.75,
    xaxis.labels.size = 0.8
  ),
  list(
    4,
    header = "Dotplot\n(dot)",
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(0, 250, 500),
    xaxis.labels = list(0, 250, 500),
    xaxis.labels.size = 0.8,
    xaxis.title = "[\u03BCS/cm]",
    left.margin = -0.5,
    right.margin = 0.3
  ),
  list(
    5,
    header = "Dotplot & CI\n(dot_ci)",
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(0, 250, 500, 750),
```

```
xaxis.labels = list(0, 250, 500, 750),
xaxis.labels.size = 0.8,
xaxis.title = "[\u03BCS/cm]",
left.margin = -0.5,
right.margin = 0.3
),
list(
  6,
  header = "Barchart\n(bar)",
  graph.bgcolor = "lightgray",
  xaxis.ticks = list(0, 250, 500),
  xaxis.labels = list(0, 250, 500),
  xaxis.labels.size = 0.8,
  xaxis.title = "[\u03BCS/cm]",
  left.margin = -0.5,
  right.margin = 0.3
),
list(
  7,
  header = "Barchart & CI\n(bar_cl)",
  graph.bgcolor = "lightgray",
  xaxis.ticks = list(0, 250, 500, 750),
  xaxis.labels = list(0, 250, 500, 750),
  xaxis.labels.size = 0.8,
  xaxis.title = "[\u03BCS/cm]",
  left.margin = -0.5,
  right.margin = 0.3
),
list(
  8,
  header = "Boxplot\n(box_summary)",
  graph.bgcolor = "lightgray",
  graph.bar.size = 0.7,
  xaxis.ticks = c(0, 1000, 2000, 3000, 4000, 5000),
  xaxis.labels = c(0, 1, 2, 3, 4, 5),
  xaxis.labels.size = 0.8,
  xaxis.title = "[1,000 \u03BCS/cm]",
  panel.width = 1.2,
  left.margin = -0.5,
  right.margin = 0.3
),
list(
  9,
  header = "Maps\n(map)",
  inactive.border.color = gray(0.7),
  inactive.border.size = 1.5,
  xaxis.labels.size = 0.8
)
```

```
)  
)
```

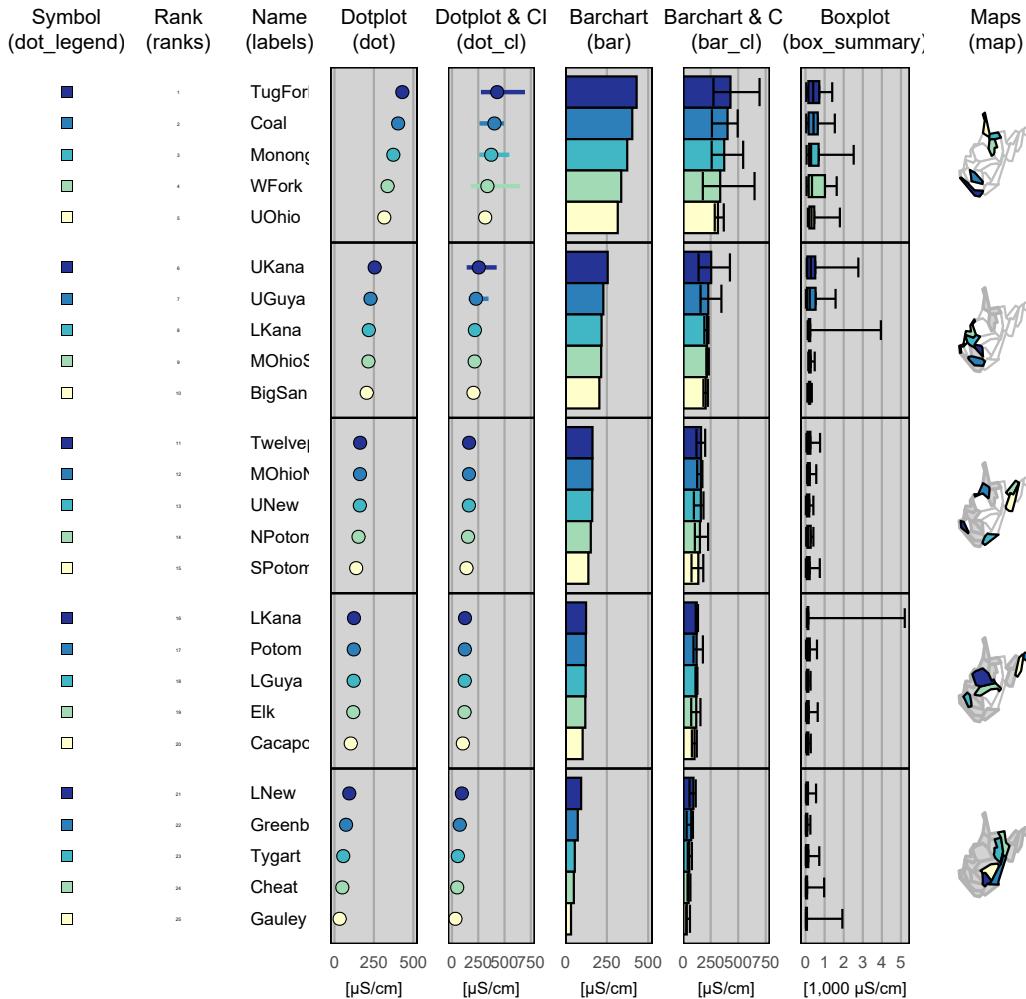


FIGURE 6.1 Linked micromap plot, based on the West Virginia watershed dataset, that shows all pre-defined plot types of the **micromap** R package (Payton & Olsen, 2015).

- general principles of adding new graph types

Examples 6.4 and 6.5 have been adapted from Symanzik et al. (2014) where they have been introduced originally.

6.3 Example 1: Adding Vignette Example

```

library(micromap)
library(ggplot2)
library(grid)
library(XML)
library(dplyr)

data("USstates")
statePolys <- create_map_table(USstates, IDcolumn = "ST")

data(lungMort)
myStats <- lungMort
head(myStats)

##      StateAb Rate_95 Count_95 Lower_95 Upper_95    Pop_95 StdErr_95 Rate_00
## AK      AK  50.0     298    44.2    56.3 1089123     3.1   46.8
## AL      AL  40.2    4095    39.0    41.4 8124753     0.6   43.4
## AR      AR  43.8    3079    42.3    45.4 5479988     0.8   48.3
## AZ      AZ  38.4    4794    37.3    39.5 10557561     0.6   38.5
## CA      CA  41.5   26931    41.0    42.0 64354973     0.3   39.2
## CO      CO  31.5    2723    30.3    32.7 9245273     0.6   33.9
##      Count_00 Lower_00 Upper_00    Pop_00 StdErr_00       State
## AK      350    41.8    52.2 1122525     2.6   Alaska
## AL     4630    42.2    44.7 8245919     0.6  Alabama
## AR     3568    46.7    49.9 5661547     0.8 Arkansas
## AZ      5482    37.4    39.5 12066024     0.5 Arizona
## CA     27406   38.7    39.6 68478617     0.2 California
## CO     3265    32.7    35.1 10159130     0.6 Colorado

myStats <- subset(myStats, !StateAb == "DC")

myNewStats <- create_DF_rank(myStats, ord.by = "Rate_00", group = 5)
head(myNewStats)

##      pGrp StateAb Rate_95 Count_95 Lower_95 Upper_95    Pop_95 StdErr_95 Rate_00
## 1 1      UT  17.6     685    16.3    18.9 5036638     0.7   16.9
## 2 1      ND  30.6     574    28.1    33.3 1527853     1.3   31.4
## 3 1      NM  31.8    1293    30.1    33.6 3899455     0.9   31.5
## 4 1      SD  30.5     659    28.2    33.1 1710003     1.2   32.9
## 5 1      CO  31.5    2723    30.3    32.7 9245273     0.6   33.9
## 6 2      ID  33.0     981    31.0    35.2 2976963     1.1   35.0
##      Count_00 Lower_00 Upper_00    Pop_00 StdErr_00       State rank median addOrd
## 1      738    15.7    18.2 5488475     0.6      Utah  1 FALSE   0
## 2      608    28.9    34.1 1480915     1.3 North Dakota  2 FALSE   0
## 3     1420    29.9    33.2 4038163     0.8 New Mexico  3 FALSE   0
## 4      736    30.5    35.5 1716683     1.2 South Dakota  4 FALSE   0

```

1426 Adding New Plot Types to Linked Micromap Plots (such as Arrow Plots, Line Charts, and Scatterplots)

```
## 5     3265     32.7     35.1 10159130      0.6    Colorado  5 FALSE   0
## 6     1158     33.0     37.1 3230513      1.0    Idaho    6 FALSE   0
##   pGrpRank pGrpOrd color
## 1         1       1     1
## 2         2       2     2
## 3         3       3     3
## 4         4       4     4
## 5         5       5     5
## 6         1       1     1
```

```
ggplot(myNewStats) +
  geom_segment(
    aes(
      x = Rate_95,
      y = -pGrpOrd,
      xend = Rate_00,
      yend = -pGrpOrd,
      colour = factor(color)
    ),
    arrow = arrow(length = unit(0.1, "cm"))
  ) +
  facet_grid(pGrp ~ .,
             scales = "free_y"
  ) +
  scale_colour_manual(
    values = c("red", "orange", "green", "blue", "purple"),
    guide = "none"
  )
```

```
myAtts <- sample_att()
myNumber <- 1
myAtts$colors <- c("red", "orange", "green", "blue", "purple")
myAtts[[myNumber]]$panel.data <- c("Rate_95", "Rate_00")

myColors <- myAtts$colors
# pulls color out of the plot level
# section of the "myAtts" attributes list

myColumns <- myAtts[[myNumber]]$panel.data
# looks in the panel level section numbered
# "myNumber" of the "myAtts" attributes list

myNewStats$data1 <- myNewStats[, myColumns[1]]
myNewStats$data2 <- myNewStats[, myColumns[2]]
myPanel <- ggplot(myNewStats) +
  geom_segment(
    aes(
      x = data1,
      y = -pGrpOrd,
```

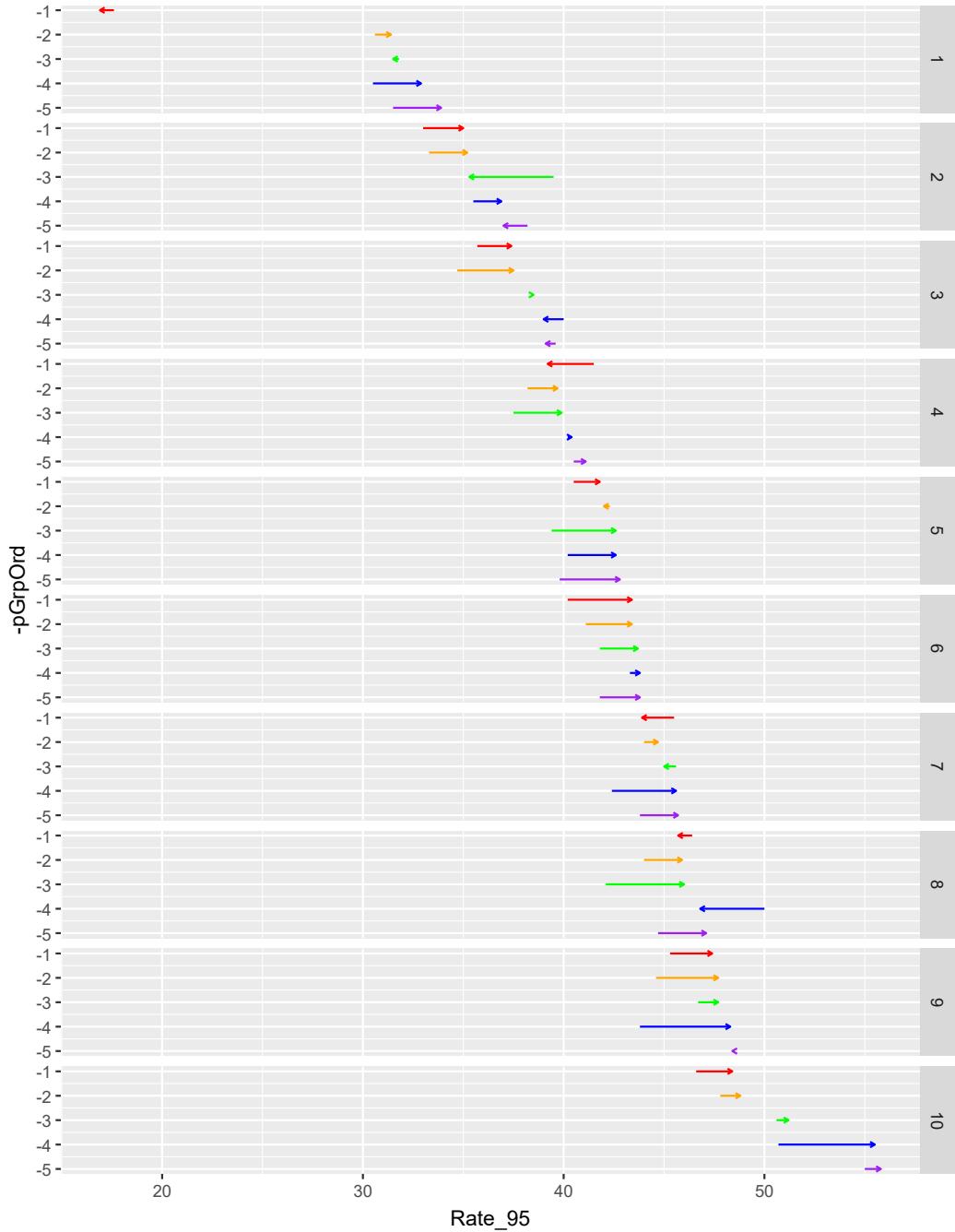


FIGURE 6.2 Arrow plot from vignette.

```

    xend = data2,
    yend = -pGrpOrd,
    colour = factor(color)
  ),
  arrow = arrow(length = unit(0.1, "cm"))
) +
facet_grid(pGrp ~ .) +
scale_colour_manual(
  values = myColors,
  guide = "none"
)
myPanel

```

```
assimilatePlot(myPanel, myNumber, myAtts)
```

```

arrow_plot_build <- function(myPanel, myNumber, myNewStats, myAtts) {
  myColors <- myAtts$colors
  myColumns <- myAtts[[myNumber]]$panel.data
  myNewStats$data1 <- myNewStats[, myColumns[1]]
  myNewStats$data2 <- myNewStats[, myColumns[2]]
  myPanel <- ggplot(myNewStats) +
    geom_segment(
      aes(
        x = data1,
        y = -pGrpOrd,
        xend = data2,
        yend = -pGrpOrd,
        colour = factor(color)
      ),
      arrow = arrow(length = unit(0.1, "cm"))
    ) +
    facet_grid(pGrp ~ .) +
    scale_colour_manual(
      values = myColors,
      guide = "none"
    )
  myPanel <- assimilatePlot(myPanel, myNumber, myAtts)
}
myPanel

```

```

arrow_plot_build <- function(myPanel, myNumber, myNewStats, myAtts) {
  myColors <- myAtts$colors
  myColumns <- myAtts[[myNumber]]$panel.data
  myNewStats$data1 <- myNewStats[, myColumns[1]]
  myNewStats$data2 <- myNewStats[, myColumns[2]]
  myNewStats <- alterForMedian(myNewStats, myAtts)
  myPanel <- ggplot(myNewStats) +

```

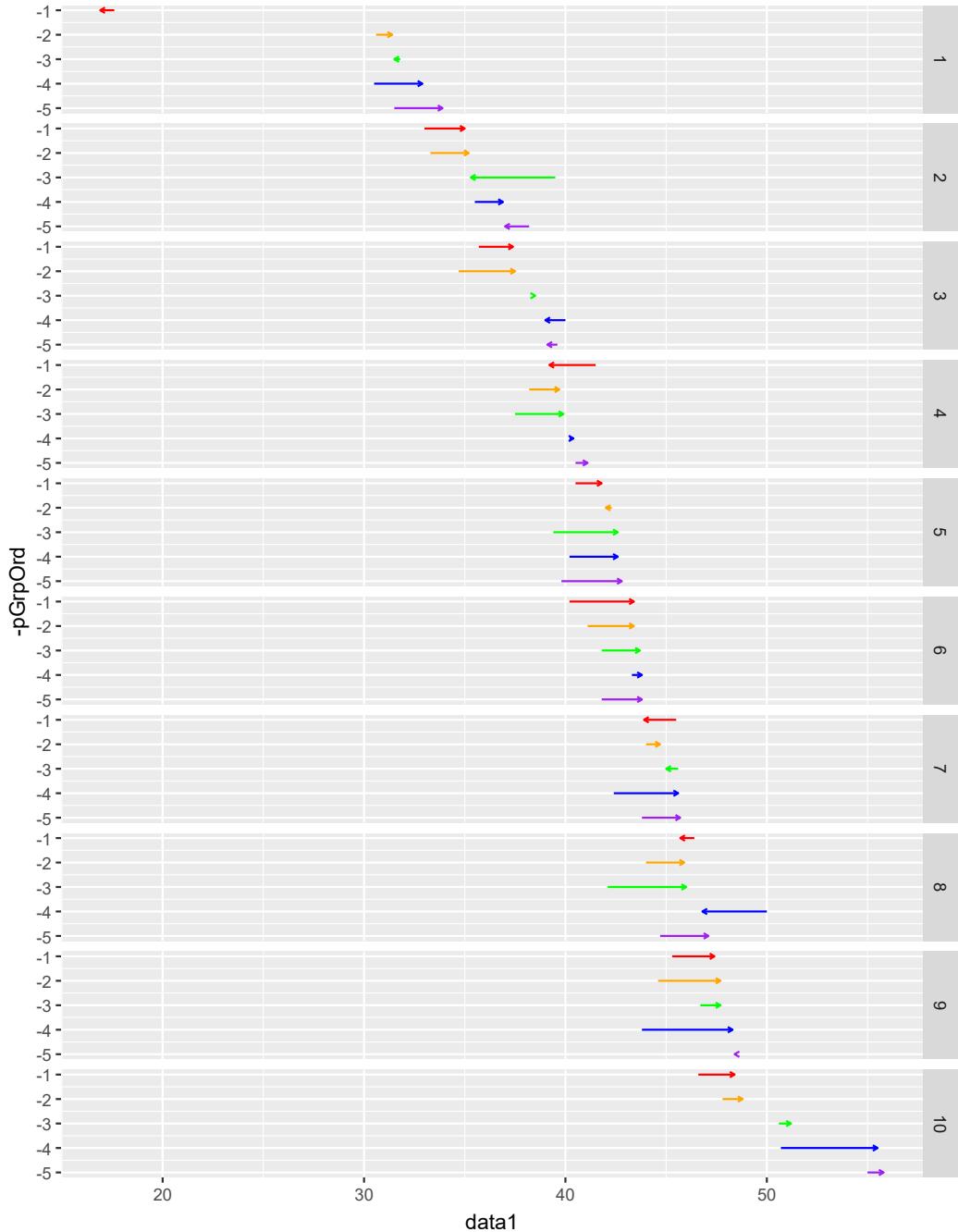


FIGURE 6.3 Arrow plot from vignette.

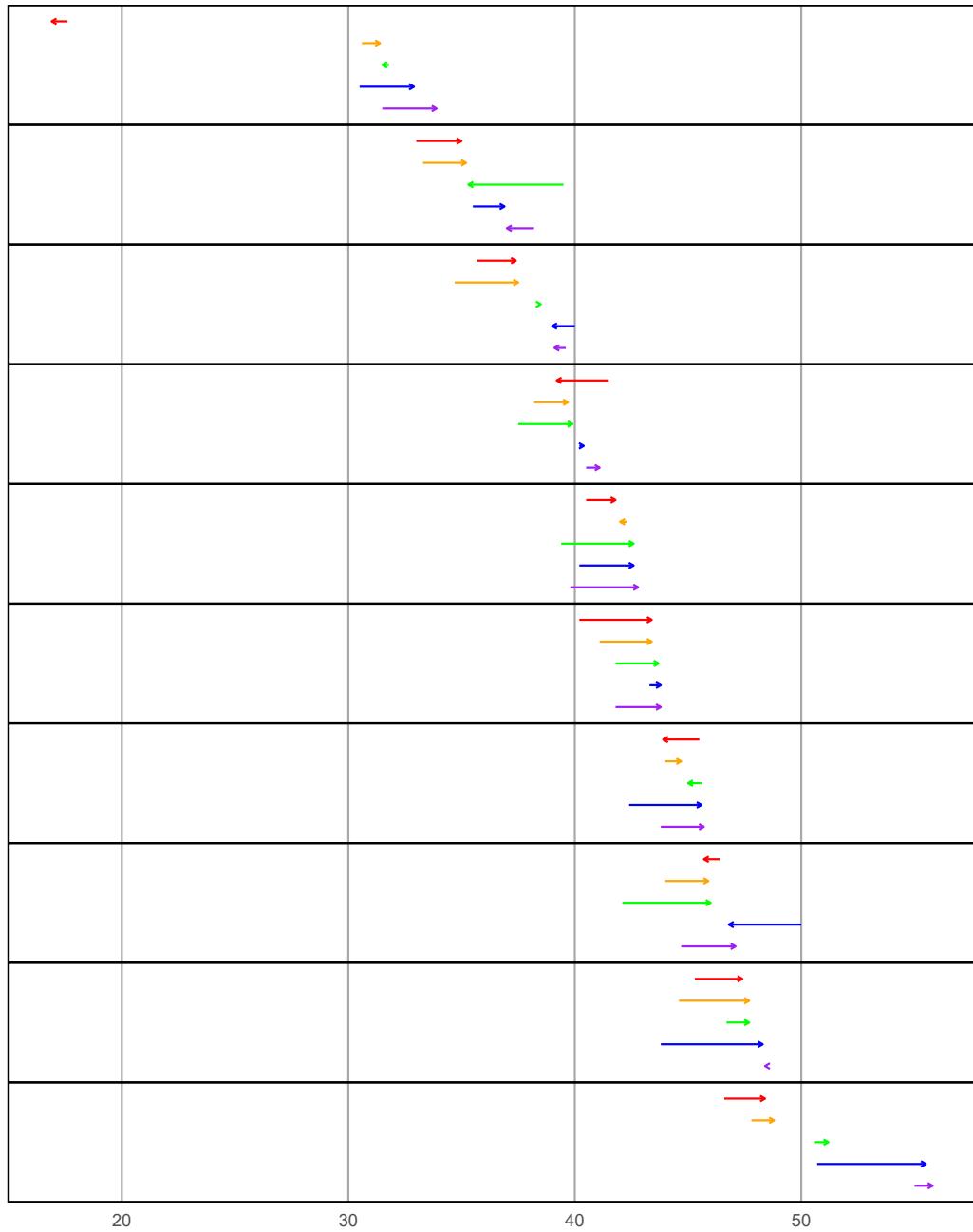


FIGURE 6.4 Arrow plot from vignette.

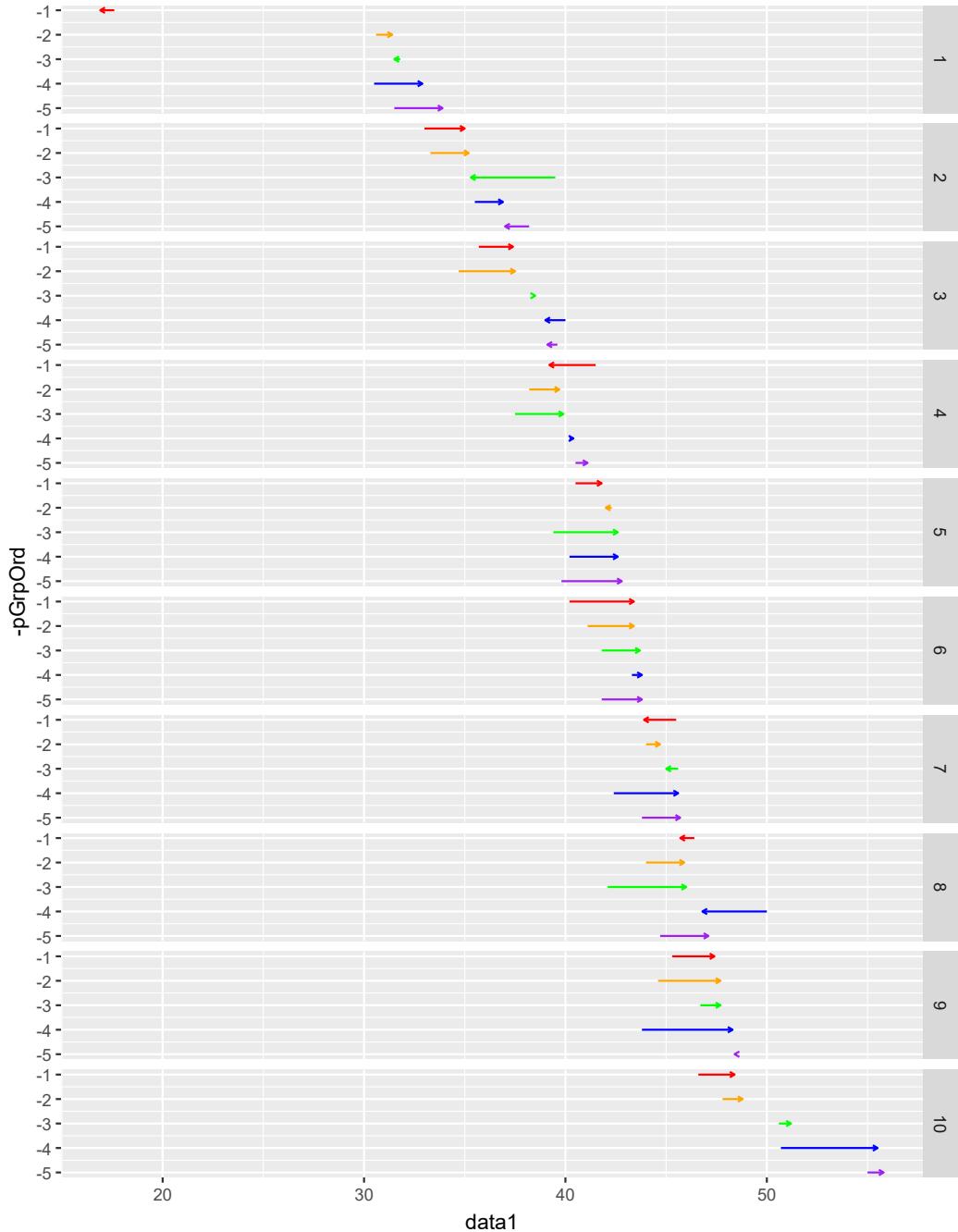


FIGURE 6.5 Arrow plot from vignette.

```

geom_segment(
  aes(
    x = data1,
    y = -pGrpOrd,
    xend = data2,
    yend = -pGrpOrd,
    colour = factor(color)
  ),
  arrow = arrow(length = unit(0.1, "cm"))
) +
facet_grid(pGrp ~ .,
  space = "free",
  scales = "free_y"
) +
scale_colour_manual(
  values = myColors,
  guide = "none"
)
myPanel <- assimilatePlot(myPanel, myNumber, myAtts)
}
myPanel

```

```

# print(myAtts)

myPanelAtts <- standard_att()
myPanelAtts <- append(
  myPanelAtts,
  list(line.width = 1, tip.length = 1)
)

arrow_plot_att <- function() {
  myPanelAtts <- standard_att()
  myPanelAtts <- append(
    myPanelAtts,
    list(line.width = 1, tip.length = 1)
  )
}

arrow_plot_build <- function(myPanel, myNumber, myNewStats, myAtts) {
  myColors <- myAtts$colors
  myColumns <- myAtts[[myNumber]]$panel.data
  myLineWidth <- myAtts[[myNumber]]$line.width
  # Again, note that these are stored in the panel level section of the
  myTipLength <- myAtts[[myNumber]]$tip.length # attributes object
  myNewStats$data1 <- myNewStats[, myColumns[1]]
  myNewStats$data2 <- myNewStats[, myColumns[2]]
  myNewStats <- alterForMedian(myNewStats, myAtts)
  myPanel <- ggplot(myNewStats) +

```

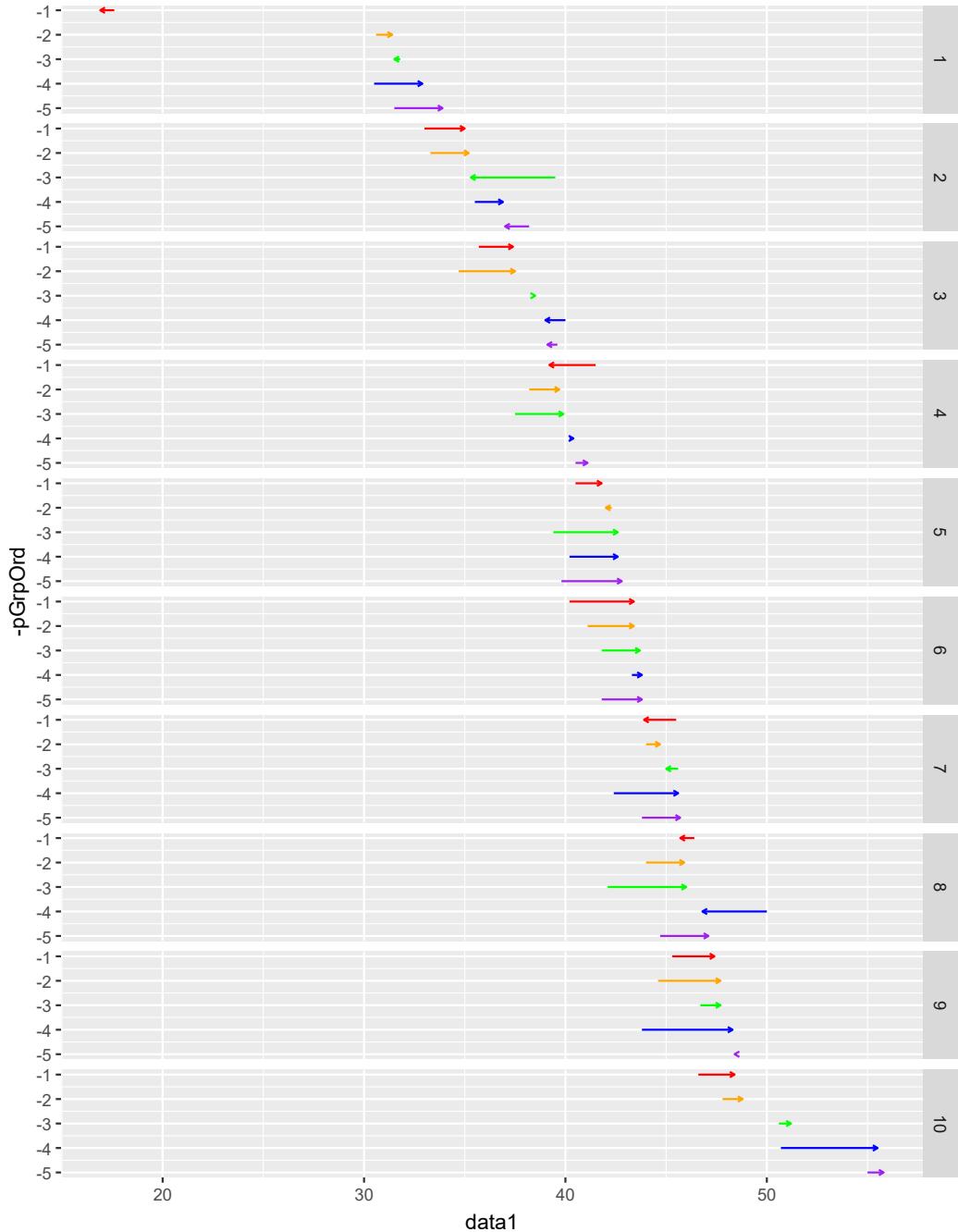


FIGURE 6.6 Arrow plot from vignette.

1506 Adding New Plot Types to Linked Micromap Plots (such as Arrow Plots, Line Charts, and Scatterplots)

```
geom_segment(
  aes(
    x = data1,
    y = -pGrpOrd,
    xend = data2,
    yend = -pGrpOrd,
    colour = factor(color)
  ),
  arrow = arrow(length = unit(0.1 * myTipLength, "cm")),
  size = myLineWidth
) +
facet_grid(pGrp ~ .,
  space = "free",
  scales = "free_y"
) +
scale_colour_manual(
  values = myColors,
  guide = "none"
)
myPanel <- assimilatePlot(myPanel, myNumber, myAtts)
}
myPanel
```

```
mmpplot(
  stat.data = myStats,
  map.data = statePolys,
  panel.types = c("map", "labels", "arrow_plot"),
  panel.data = list(NA, "State", list("Rate_95", "Rate_00")),
  ord.by = "Rate_00",
  grouping = 5,
  map.link = c("StateAb", "ID"),
  panel.att =
  list(
    list(
      3,
      line.width = 1.25,
      tip.length = 1.5
    )
  )
)
```

```
data(lungMort)
myStats <- lungMort

mmpplot(
  stat.data = myStats,
  map.data = statePolys,
```

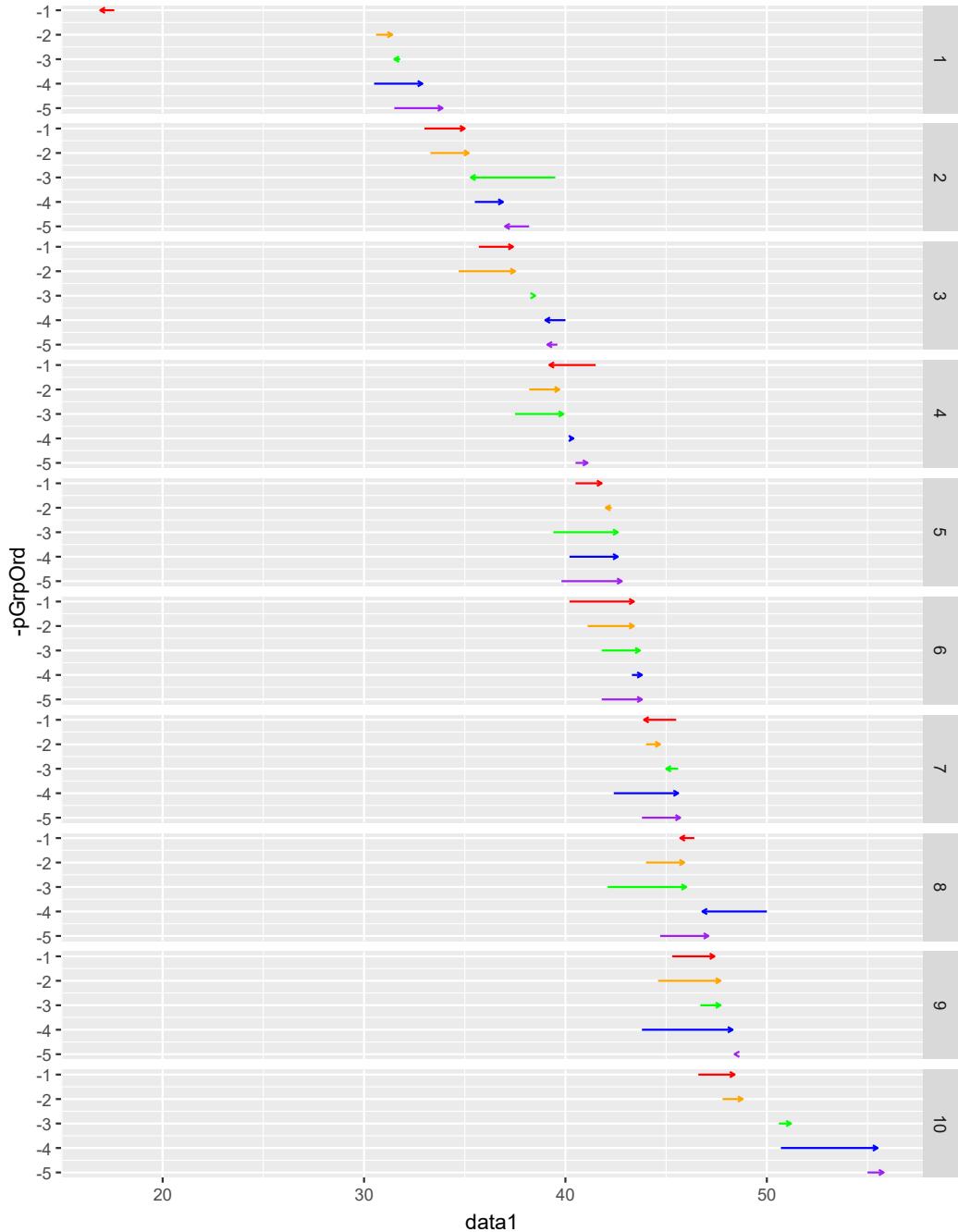
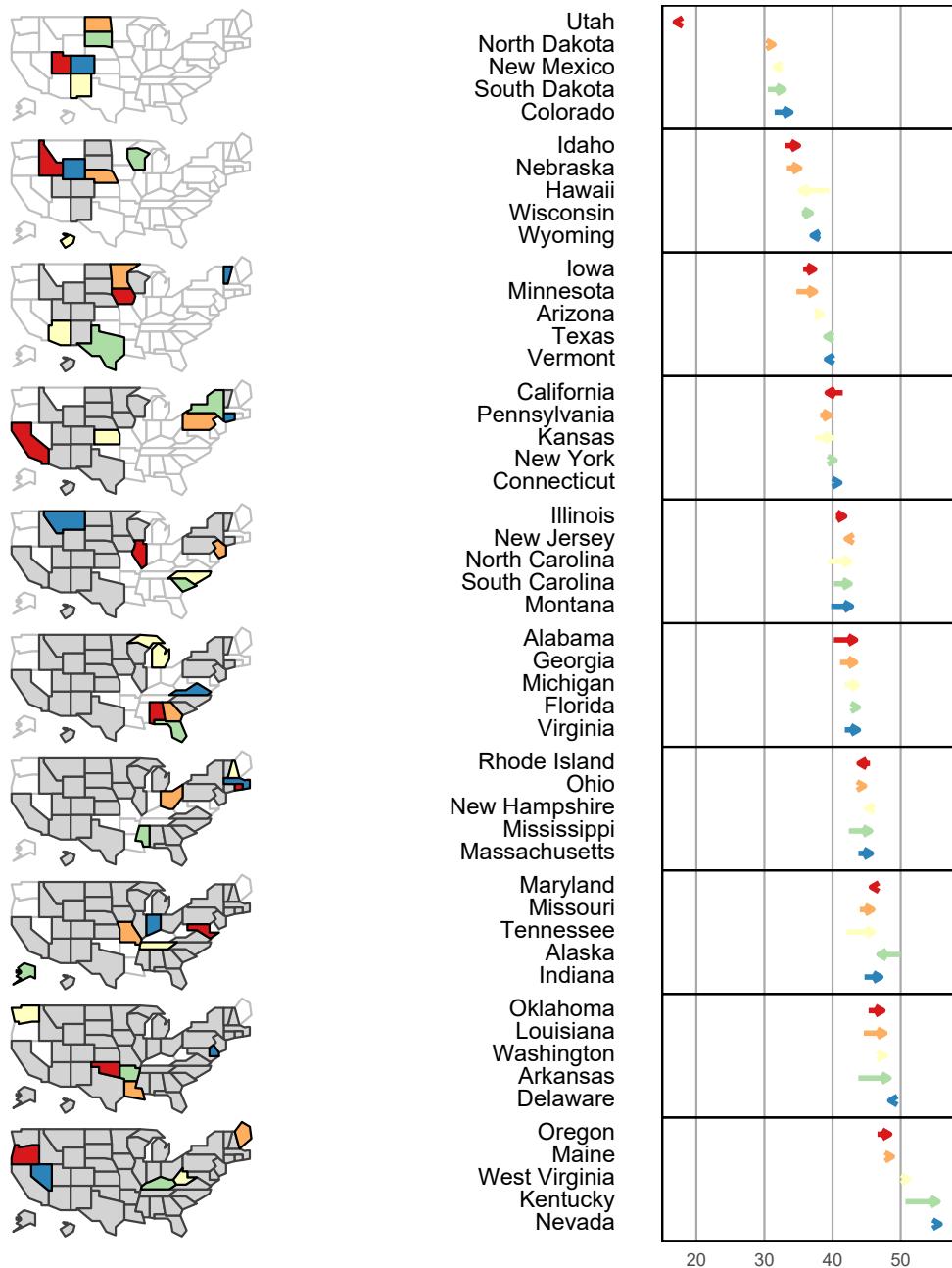


FIGURE 6.7 Arrow plot from vignette.

**FIGURE 6.8** Arrow plot from vignette.

```
panel.types = c("map", "dot_legend", "labels", "dot_cl", "arrow_plot"),
panel.data = list(
  NA,
  "points",
  "State",
  list("Rate_00", "Lower_00", "Upper_00"),
  list("Rate_95", "Rate_00")
),
ord.by = "Rate_00",
grouping = 5,
median.row = TRUE,
map.link = c("StateAb", "ID"),
plot.height = 10,
colors = c("red", "orange", "green", "blue", "purple"),
panel.att = list(
  list(
    1,
    header = "Light Gray Means\nHighlighted Above",
    map.all = TRUE,
    fill.regions = "two ended",
    inactive.fill = "lightgray",
    inactive.border.color = gray(.7),
    inactive.border.size = 2,
    panel.width = 1
  ),
  list(
    2,
    point.type = 20,
    point.border = TRUE
  ),
  list(
    3,
    header = "U.S. \nStates ",
    panel.width = .8,
    align = "left",
    text.size = .9
  ),
  list(
    4,
    header = "State 2000\n Rate and 95% CI",
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(20, 30, 40, 50),
    xaxis.labels = list(20, 30, 40, 50),
    xaxis.title = "Deaths per 100,000"
  ),
  list(
    5,
    header = "State Rate Change\n 1995-99 to 2000-04",
    line.width = 1.25,
```

```

    tip.length = 1.5,
    graph.bgcolor = "lightgray",
    xaxis.ticks = list(20, 30, 40, 50),
    xaxis.labels = list(20, 30, 40, 50),
    xaxis.title = "Deaths per 100,000"
)
)
)

```

6.4 Example 2: Adding Bar Charts as a New Plot Type

```
ArgShapefile <- as_Spatial(st_read("data/ARG_adm/ARG_adm1.shp"))
```

```

## Reading layer `ARG_adm1' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\ARG_adm\ARG_adm1.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 24 features and 16 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box: xmin: -73.58 ymin: -55.06 xmax: -53.59 ymax: -21.78
## Geodetic CRS:  WGS 84

```

```
names(ArgShapefile)
```

```

## [1] "ID_0"        "ISO"         "NAME_0"       "ID_1"        "NAME_1"
## [6] "VARNAME_1"   "NL_NAME_1"   "HASC_1"       "CC_1"        "TYPE_1"
## [11] "ENGTYPE_1"   "VALIDFR_1"   "VALIDTO_1"   "REMARKS_1"  "Shape_Leng"
## [16] "Shape_Area"

```

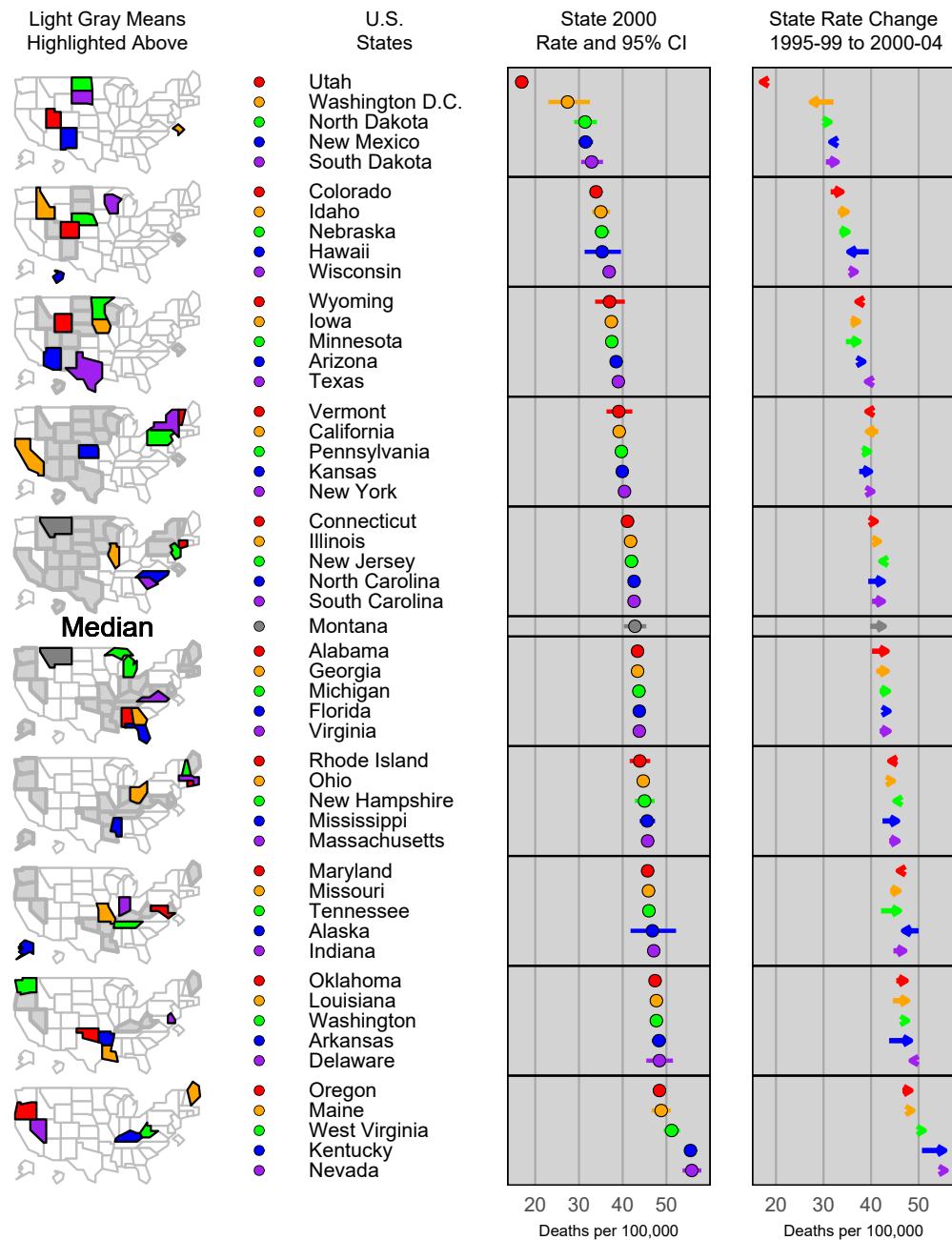
```
ArgShapefile$NAME_1
```

```

## [1] "Buenos Aires"          "Catamarca"           "Chaco"
## [4] "Chubut"                "Ciudad de Buenos Aires" "Córdoba"
## [7] "Corrientes"            "Entre Ríos"           "Formosa"
## [10] "Jujuy"                 "La Pampa"             "La Rioja"
## [13] "Mendoza"               "Misiones"             "Neuquén"
## [16] "Río Negro"              "Salta"                "San Juan"
## [19] "San Luis"               "Santa Cruz"           "Santa Fe"
## [22] "Santiago del Estero"  "Tierra del Fuego"      "Tucumán"

```

```
ArgShapefileThin <- st_as_sf(ArgShapefile) %>%
  st_simplify(
    dTolerance = 1000,
```

**FIGURE 6.9** Arrow plot from vignette.

```

  preserveTopology = TRUE
) %>%
as_Spatial()

# manipulate population dataset
Argfile <- "data/List of Argentine provinces by population - Wikipedia, the free encyclopedia.htm"
Argtable <- readHTMLTable(Argfile)[[1]]

Argtable <- Argtable[, -3]
Argtable <- apply(Argtable, 2, function(x) gsub(".*00000", "", x))
Argtable <- apply(Argtable, 2, function(x) gsub(", ", "", x))
Argtable <- as.data.frame(Argtable)
Argtable[, 1] <- as.character(Argtable[, 1])
Argtable[, 2] <- as.character(Argtable[, 2])
Argtable[, 3] <- as.numeric(as.character(Argtable[, 3]))
Argtable[, 4] <- as.numeric(as.character(Argtable[, 4]))
Argtable[4, 2] <- "Ciudad de Buenos Aires"
provincematch <- gregexpr("\\<[:alpha:][:space:]*", Argtable[, 2])
Argtable[, 2] <- unlist(regmatches(Argtable[, 2], provincematch))
colnames(Argtable) <- c("Rank", "provinces", "2010 Population", "2001 Population")
Argtable[, "change"] <- Argtable[, 3] - Argtable[, 4]
provinces <- as.character(sort(Argtable[, 2]))
ArgShapefileThin@data <- cbind(ArgShapefileThin@data, provinces)
Argtable[, 3:5] <- sapply(Argtable[, 3:5], function(x) x / 1000000)
Argtable[, "ratio"] <- Argtable[, 3] / Argtable[, 4]

# arg1_1 - figure not used in accompanying article

ArgPolys <- create_map_table(ArgShapefileThin, "provinces")

myTable <- create_DF_rank(Argtable,
  ord.by = "2010 Population",
  group = c(5, 5, 4, 5, 5), rev.order = TRUE
)
myTable[11:14, "pGrpOrd"] <- myTable[11:14, "pGrpOrd"] + 0.5
myAtts <- sample_att()
myNumber <- 1
myAtts$colors <- c("red", "orange", "green", "blue", "purple")
myAtts[[myNumber]]$panel.data <- "change"
myColors <- myAtts$colors
myColumns <- myAtts[[myNumber]]$panel.data
myTable$data1 <- myTable[, myColumns]
myPanel <- ggplot(myTable) +
  geom_segment(aes(
    x = 0, y = -pGrpOrd, xend = data1, yend = -pGrpOrd,
    colour = factor(color)
  )) +
  facet_grid(pGrp ~ .) +

```

```

scale_colour_manual(
  values = myColors,
  guide = "none"
)
assimilatePlot(myPanel, myNumber, myAtts)

myPanelAtts <- standard_att()
myPanelAtts <- append(myPanelAtts, list(line.width = 4))

bar_plot_att <- function() {
  myPanelAtts <- standard_att()
  myPanelAtts <- append(myPanelAtts, list(line.width = 4))
}

bar_plot_build <- function(myPanel, myNumber, myNewStats, myAtts) {
  myColors <- myAtts$colors
  myColumns <- myAtts[[myNumber]]$panel.data
  myLineWidth <- myAtts[[myNumber]]$line.width
  myTipLength <- myAtts[[myNumber]]$tip.length
  myTable$data1 <- myNewStats[, myColumns]
  myPanel <- ggplot(myTable) +
    geom_segment(
      aes(
        x = 0, y = -pGrpOrd,
        xend = data1, yend = -pGrpOrd,
        colour = factor(color)
      ),
      size = myLineWidth
    ) +
    facet_grid(pGrp ~ ., space = "free", scales = "free_y") +
    scale_colour_manual(values = myColors, guide = "none")
  myPanel <- assimilatePlot(myPanel, myNumber, myAtts)
}

Argtable$ProvinceTemp <- as.character(Argtable$provinces)

# arg1_2 - Figure 3
mmpplot(
  stat.data = Argtable,
  map.data = ArgPolys,
  panel.types = c(
    "map", "dot_legend", "labels",
    "dot", "bar_plot", "dot"
  ),
  panel.data = list(
    NA, "points", "ProvinceTemp",
    "2010 Population", "change", "ratio"
  ),
  ord.by = "2010 Population",
)

```

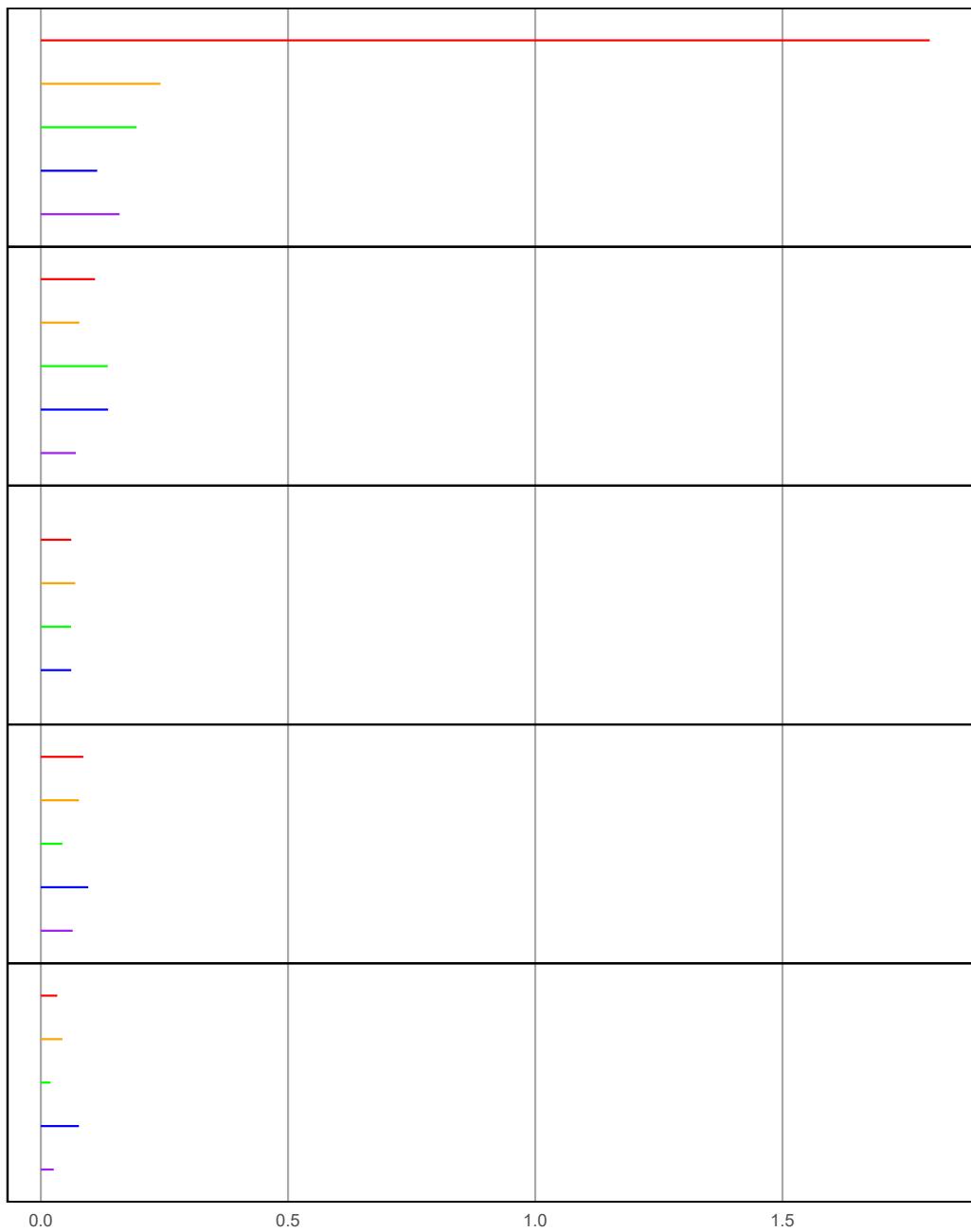


FIGURE 6.10 Bar chart.

```
rev.order = TRUE,
grouping = c(5, 5, 4, 5, 5),
vertical.align = "center",
map.link = c("provinces", "ID"),
colors = brewer.pal(6, "Greys")[6:1],
map.color2 = gray(0.95),
plot.width = 10,
panel.att = list(
  list(
    1,
    header = "Maps",
    panel.width = 0.6,
    active.border.color = "black",
    active.border.size = 0.5,
    inactive.border.color = gray(0.7),
    inactive.border.size = 0.5
  ),
  list(
    2,
    point.size = 0.8
  ),
  list(
    3,
    header = "Provinces",
    align = "left",
    text.size = 1.0,
    panel.width = 1.5
  ),
  list(
    4,
    header = "2010 Population",
    graph.bgcolor = gray(0.95),
    xaxis.ticks = list(0, 4, 8, 12, 16),
    xaxis.labels = list(0, 4, 8, 12, 16),
    xaxis.title = "People (Millions)",
    panel.width = 1.8
  ),
  list(
    5,
    header = "Population Increase\nfrom 2001 to 2010",
    graph.bgcolor = gray(0.95),
    xaxis.ticks = list(0, 0.9, 1.8),
    xaxis.labels = list(0, 0.9, 1.8),
    xaxis.title = "People (Millions)",
    panel.width = 1.8
  ),
  list(
    6,
    header = "Ratio of 2010 Population\n\tto 2001 Population",
```

```
graph.bgcolor = gray(0.95),
xaxis.ticks = list(1, 1.2, 1.4),
xaxis.labels = list(1, 1.2, 1.4),
panel.width = 1.8
)
)
)
```

6.5 Example 3: Adding Arrow Plots for Differences as a New Plot Type

```
library(micromap)
library(ggplot2)
library(XML)
library(dplyr)

BraShapefile <- sf:::as_Spatial(sf:::st_read("data/BRA_adm/BRA_adm1.shp"))

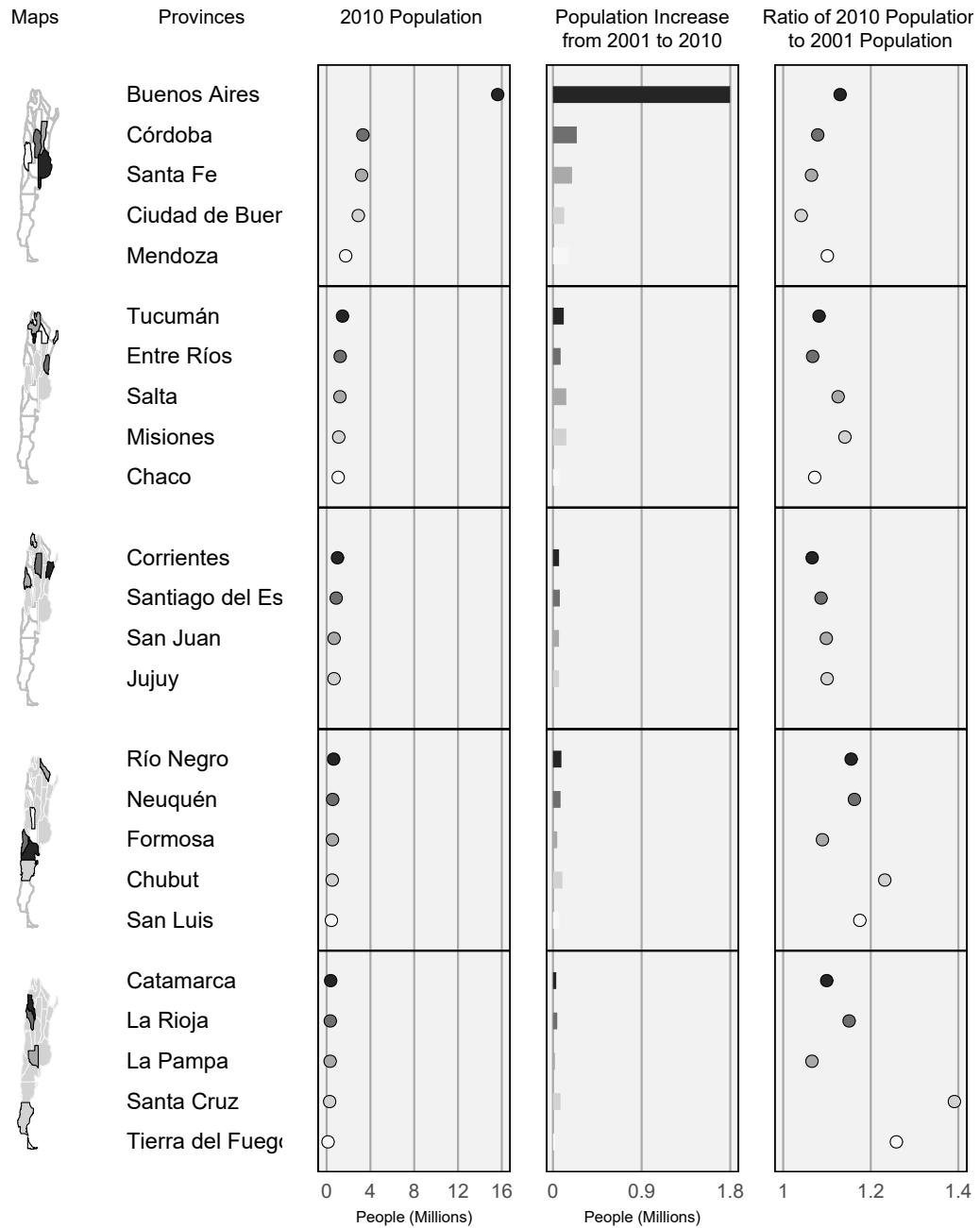
## Reading layer `BRA_adm1' from data source
##   `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\BRA_adm\BRA_adm1.shp'
##     using driver `ESRI Shapefile'
## Simple feature collection with 27 features and 16 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: -73.99 ymin: -33.75 xmax: -28.85 ymax: 5.265
## Geodetic CRS:  WGS 84

BraShapefileThin <- st_as_sf(BraShapefile) %>%
  st_simplify(
    dTolerance = 1000,
    preserveTopology = TRUE
  ) %>%
  as_Spatial()

brazil4 <- BraShapefileThin

Brafile <- "data/States of Brazil - Wikipedia, the free encyclopedia.htm"
Bratable <- readHTMLTable(Brafile)[[12]]

Bratable <- apply(Blatable, 2, function(x) gsub("[().*]", "", x))
Bratable <- apply(Blatable, 2, function(x) gsub(".*00000", "", x))
Bratable <- apply(Blatable, 2, function(x) gsub("[,]", "", x))
Bratable <- apply(Blatable, 2, function(x) gsub("[%]", "", x))
```

**FIGURE 6.11** Bar chart.

```

Bratable[, 12] <- substr(Bratable[, 12], 1, nchar(Bratable[, 12]) - 1)
Bratable <- as.data.frame(Bratable)
Bratable <- data.frame(
  Bratable[, 1:4],
  sapply(Bratable[, 5:13], function(x) as.numeric(as.character(x)))
)
State <- as.character(sort(Bratable[, 2]))

StateReordered <- State[c(16, 18:27, 15, 17)]
State[15:27] <- StateReordered

brazil4@data <- cbind(brazil4@data, State)
Bratable[, 6] <- Bratable[, 6] / 1000000

BrazilPolys <- create_map_table(brazil4, "State")

# bra2 - Figure 6

Brafile2 <- "data/List of Brazilian states by murder rate - Wikipedia, the free encyclopedia.htm"
Bratable2 <- readHTMLTable(Brafile2)[[1]]

Bratable2 <- Bratable2[-c(8, 18, 23, 27, 32, 33), ]
Bratable2[, 2:13] <- sapply(Bratable2[, 2:13], function(x) as.numeric(as.character(x)))
Bratable2[, 1] <- as.character(Bratable2[, 1])
Bratable2[, 1] <- substring(Bratable2[, 1], 2)
Bratable2[, "change"] <- Bratable2[, 12] - Bratable2[, 2]
Bratable2[, "zero"] <- 0
Bratable2 <- Bratable2[order(Bratable2[, 1]), ]
myTable <- create_DF_rank(Bratable2,
  ord.by = "2008",
  group = c(4, 4, 4, 3, 4, 4, 4), rev.order = TRUE
)
myTable[13:15, "pGrpOrd"] <- myTable[13:15, "pGrpOrd"] + 0.6
names(Bratable2)[1] <- "states"
names(myTable)[2] <- "states"

myAtts <- sample_att()
myNumber <- 1
myAtts$colors <- c("red", "orange", "green", "blue")
myAtts[[myNumber]]$panel.data <- c("zero", "change")
myColors <- myAtts$colors
myColumns <- myAtts[[myNumber]]$panel.data
myTable$data1 <- myTable[, myColumns[1]]
myTable$data2 <- myTable[, myColumns[2]]
myPanel <- ggplot(myTable) +
  geom_segment(
    aes(
      x = data1, y = -pGrpOrd,

```

```

    xend = data2, yend = -pGrpOrd, colour = factor(color)
),
arrow = arrow(length = unit(0.1, "cm"))
) +
facet_grid(pGrp ~ .) +
scale_colour_manual(values = myColors, guide = "none")

assimilatePlot(myPanel, myNumber, myAtts)

```

```

myPanelAtts <- standard_att()
myPanelAtts <- append(
  myPanelAtts,
  list(line.width = 1, tip.length = 1)
)

arrow_plot_att <- function() {
  myPanelAtts <- standard_att()
  myPanelAtts <- append(
    myPanelAtts,
    list(line.width = 1, tip.length = 1)
  )
}

arrow_plot_build <- function(myPanel, myNumber, myNewStats, myAtts) {
  myColors <- myAtts$colors
  myColumns <- myAtts[[myNumber]]$panel.data
  myLineWidth <- myAtts[[myNumber]]$line.width
  myTipLength <- myAtts[[myNumber]]$tip.length
  myTable$data1 <- myNewStats[, myColumns[1]]
  myTable$data2 <- myNewStats[, myColumns[2]]
  myPanel <- ggplot(myTable) +
    geom_segment(
      aes(
        x = data1, y = -pGrpOrd,
        xend = data2, yend = -pGrpOrd,
        colour = factor(color)
      ),
      arrow = arrow(length = unit(0.1 * myTipLength, "cm")),
      size = myLineWidth
    ) +
    facet_grid(pGrp ~ ., space = "free", scales = "free_y") +
    scale_colour_manual(values = myColors, guide = "none")
  myPanel <- assimilatePlot(myPanel, myNumber, myAtts)
}

Bratable2$StateMod <- as.character(Bratable2$states)

mmpplot(
  stat.data = Bratable2,

```

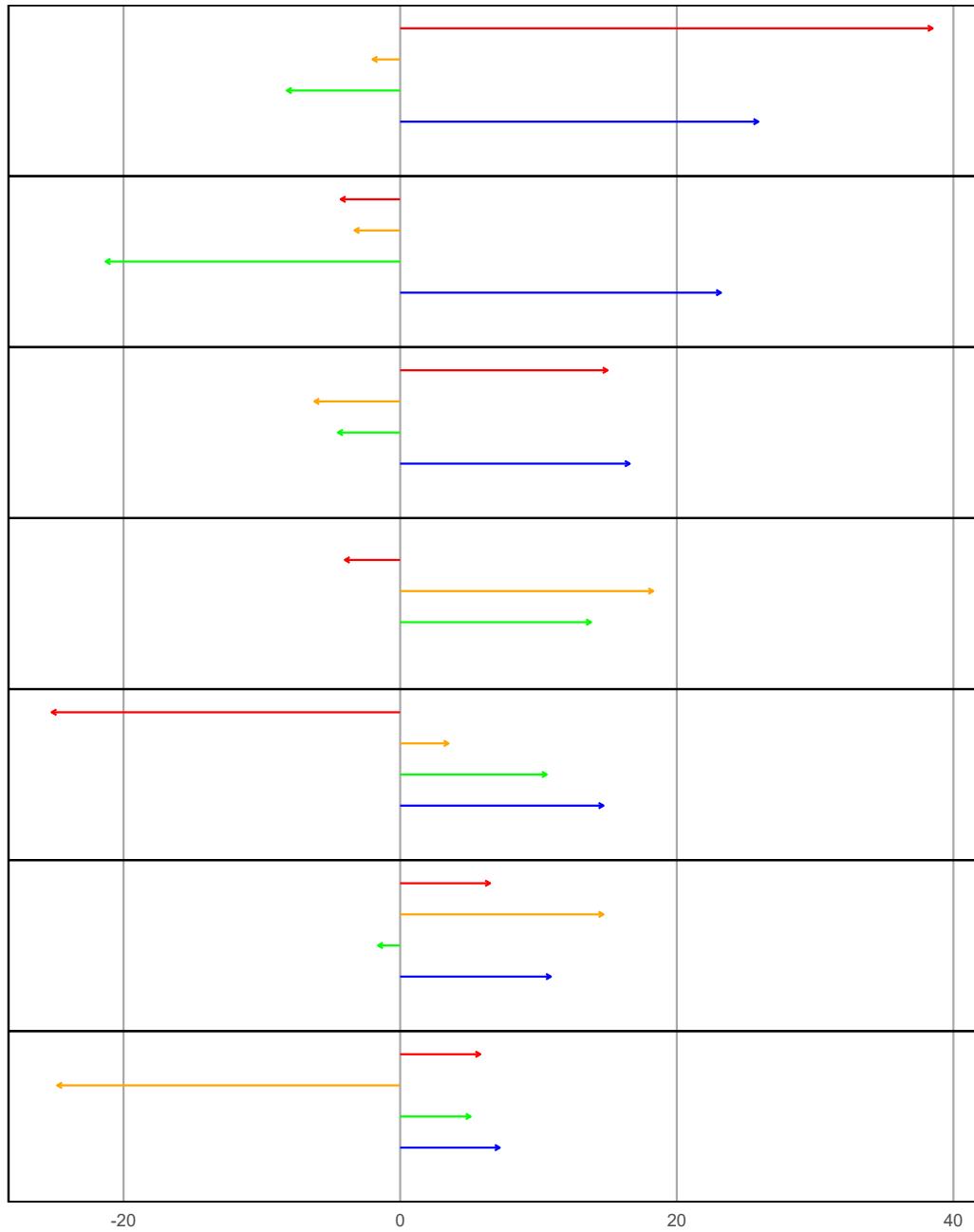
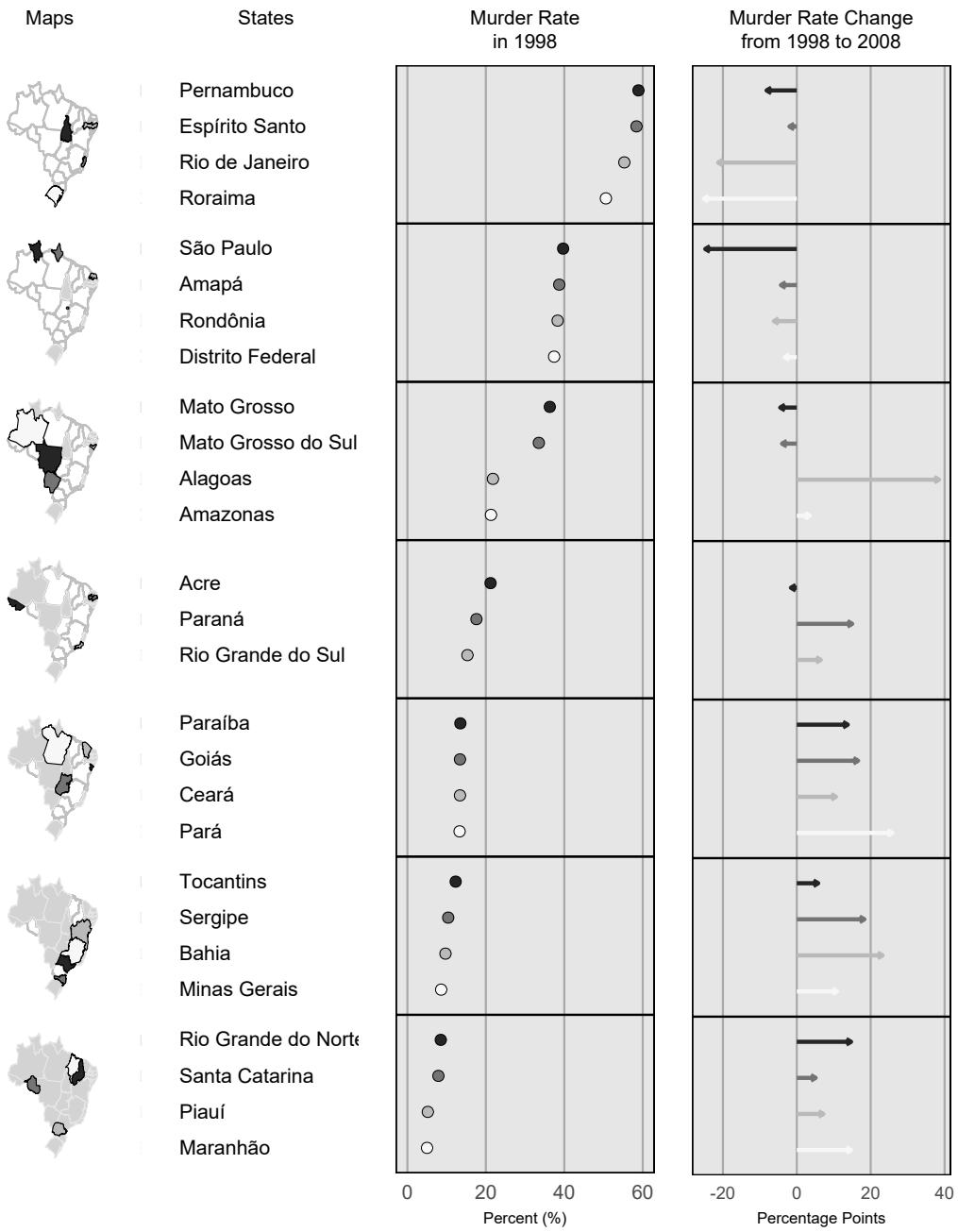


FIGURE 6.12 Arrow plot for differences.

```
map.data = BrazilPolys,
panel.types = c("map", "dot_legend", "labels", "dot", "arrow_plot"),
panel.data = list(NA, "points", "StateMod", "1998", c("zero", "change")),
ord.by = "1998",
rev.order = TRUE,
grouping = c(4, 4, 4, 3, 4, 4, 4),
vertical.align = "center",
map.link = c("states", "ID"),
colors = brewer.pal(5, "Greys")[5:1],
map.color2 = gray(0.9),
panel.att = list(
  list(
    1,
    header = "Maps",
    panel.width = 0.8,
    active.border.color = "black",
    active.border.size = 0.5,
    inactive.border.color = gray(0.7),
    inactive.border.size = 0.5
  ),
  list(
    2,
    point.size = 0.8
  ),
  list(
    3,
    header = "States",
    align = "left",
    text.size = 0.9,
    panel.width = 1.3
  ),
  list(
    4,
    header = "Murder Rate\nin 1998",
    point.type = 20,
    point.size = 1.4,
    graph.bgcolor = gray(0.9),
    xaxis.ticks = list(0, 20, 40, 60),
    xaxis.labels = list(0, 20, 40, 60),
    xaxis.title = "Percent (%)",
    panel.width = 1.8
  ),
  list(
    5,
    header = "Murder Rate Change\nfrom 1998 to 2008",
    graph.bgcolor = gray(0.9),
    xaxis.title = "Percentage Points",
    panel.width = 1.8
  )
)
```

```
)  
)
```

**FIGURE 6.13** Arrow plot for differences.

6.6 Example 4: Adding Scatterplots as a New Plot Type

```

#### load packages
library(micromap)
library(ggplot2)
library(grid)
# library(rgdal)

#### read code to print at end
# read_chunk("scatdot_build.R")
# read_chunk("scatdot_att.R")
# read_chunk("axis_opts_grid.R")

#### source the functions in to the global environment
# source("scatdot_build.R")

scatdot_build <- function(pl, p, DF, att) {
  # define the data to use
  DF$tmp.data1 <- DF[, unlist(att[[p]]$panel.data[1])]
  DF$tmp.data2 <- DF[, unlist(att[[p]]$panel.data[2])]

  DF <- alterForMedian(DF, att)

  # set the x and y axis limits
  # start with the ranges of the data
  tmp.limsx <- range(DF[, unlist(att[[p]]$panel.data[1])], na.rm = T)
  tmp.limsy <- range(DF[, unlist(att[[p]]$panel.data[2])], na.rm = T)

  # expand if the ticks cover a wider range
  if (any(!is.na(att[[p]]$xaxis.ticks))) {
    tmp.limsx <- range(c(tmp.limsx, att[[p]]$xaxis.ticks))
  }
  if (any(!is.na(att[[p]]$yaxis.ticks))) {
    tmp.limsy <- range(c(tmp.limsy, att[[p]]$yaxis.ticks))
  }

  # expand a bit more so things aren't bumping against the edges
  tmp.limsx <- tmp.limsx + c(-1, 1) * diff(tmp.limsx) * 0.05
  tmp.limsy <- tmp.limsy + c(-1, 1) * diff(tmp.limsy) * 0.05

  # ensure the range isn't zero
  if (diff(tmp.limsx) == 0) {
    tmp.limsx <- tmp.limsx + c(-0.5, 0.5)
  }
  if (diff(tmp.limsy) == 0) {
    tmp.limsy <- tmp.limsy + c(-0.5, 0.5)
  }
}

```

```

# median row margins, if needed
tmp.median.limsy <- NULL

if (att$median.row) {
  if (!any(DF$median)) {
    DF <- rbind(DF, transform(DF[1, ],
      pGrpOrd = 1,
      pGrp = att$m.pGrp,
      median = TRUE,
      rank = "",
      tmp.data = median(DF$tmp.data)
    ))
  }
  tmp.median.limsy <- c(-0.5, -1.5)
}

# create a new dataframe to hold the full set of points
# 'n' for every group, except the median
if (att[[p]]$annotate.all) {
  fullDF <- tmpDF <- DF
  list <- unique(DF$pGrp[!DF$median])
  for (j in 1:length(list)) {
    tmpDF$pGrp <- list[j]
    tmpDF$color <- att$grouping + 2
    tmpDF$median <- FALSE
    fullDF <- merge(fullDF, tmpDF, all = TRUE)
  }
} else {
  fullDF <- DF
}

fullDF$ann <- (fullDF$color == att$grouping + 2) + 1

if (att$median.row) {
  if (fullDF[fullDF$median, ]$pGrp %in% 1:max(fullDF$pGrp)) {
    fullDF[fullDF$median, ]$pGrp <- fullDF[fullDF$median, ]$pGrp - 1
    fullDF[fullDF$median, ]$color <- att$grouping + 1

    copyMed <- fullDF[fullDF$median, ]
    copyMed$pGrp <- copyMed$pGrp + 2

    fullDF <- merge(fullDF, copyMed, all = TRUE)
  } else {
    fullDF <- fullDF[!fullDF$median, ]
  }
}

att$colors <- c(

```

```
att$colors[1:att$grouping],  
att[[p]]$median.point.color  
)  
  
# start the panel  
pl <- ggplot(fullDF)  
  
# plot points  
pl <- pl + geom_point(aes(  
  x = tmp.data1,  
  y = tmp.data2,  
  color = as.factor(color),  
  size = as.factor(ann),  
  pch = as.factor(ann),  
  alpha = as.factor(ann)  
(  
))  
  
# use the correct colors, no legend needed  
pl <- pl + scale_colour_manual(  
  values = c(  
    att$colors,  
    att[[p]]$annotate.color  
(,  
  guide = "none"  
)  
pl <- pl + scale_size_manual(  
  values = c(  
    att[[p]]$point.size * 2,  
    att[[p]]$annotate.size * 2  
(,  
  guide = "none"  
)  
pl <- pl + scale_shape_manual(  
  values = c(  
    att[[p]]$point.type,  
    att[[p]]$annotate.type  
(,  
  guide = "none"  
)  
pl <- pl + scale_alpha_manual(  
  values = c(  
    att[[p]]$point.alpha,  
    att[[p]]$annotate.alpha  
(,  
  guide = "none"  
)  
  
# use facet to split up the groups
```

```

pl <- pl + facet_grid(pGrp ~ .,
  space = "free",
  scales = "free_y",
  drop = TRUE
)

# let the 3 package functions alter some basics
pl <- plot_opts(p, pl, att)
pl <- graph_opts(p, pl, att)
pl <- axis_opts_grid(p, pl, att,
  limsx = tmp.limsx,
  limsy = c(tmp.limsy, tmp.median.limsy),
  border = att[[p]]$border
)
pl
}

# source("scatdot_att.R")

scatdot_att <- function(show = FALSE) {
  tmp.att <- list(
    panel.header = NA,
    panel.header.size = as.numeric(1),
    panel.header.color = "black",
    panel.header.face = "plain",
    panel.header.font = NA,
    panel.header.lineheight = as.numeric(1),
    panel.width = as.numeric(1),
    panel.bgcolor = NA,
    left.margin = NA,
    right.margin = NA,
    panel.margins = c(1, -0.25, 1, -1.5),
    graph.grid.major = as.logical(TRUE),
    graph.grid.major.lty = as.numeric(1),
    graph.grid.major.col = "dark gray",
    graph.grid.minor = as.logical(TRUE),
    graph.grid.minor.lty = as.numeric(3),
    graph.grid.minor.col = "light gray",
    graph.bgcolor = NA,
    graph.border.color = "black",
    xaxis.title = "",
    xaxis.ticks = NA,
    xaxis.labels = NA,
    xaxis.line.display = as.logical(TRUE),
    xaxis.ticks.display = as.logical(TRUE),
    xaxis.text.display = as.logical(TRUE),
    xaxis.labels.size = as.numeric(1),
    xaxis.labels.angle = NULL,
    xaxis.labels.hjust = NULL,
  )
}

```

```
xaxis.labels.vjust = NULL,
xaxis.title.size = as.numeric(1),
xaxis.title.color = "black",
xaxis.title.face = "plain",
xaxis.title.font = NA,
xaxis.title.lineheight = as.numeric(1),
yaxis.title = "",
yaxis.ticks = NA,
yaxis.labels = NA,
yaxis.line.display = as.logical(TRUE),
yaxis.ticks.display = as.logical(TRUE),
yaxis.text.display = as.logical(TRUE),
yaxis.labels.size = as.numeric(1),
yaxis.labels.angle = NULL,
yaxis.labels.hjust = NULL,
yaxis.labels.vjust = NULL,
yaxis.title.size = as.numeric(1),
yaxis.title.color = "black",
yaxis.title.face = "plain",
yaxis.title.font = NA,
yaxis.title.lineheight = as.numeric(1),
panel.margins = c(1, -0.25, 1, 1.5),
point.size = as.numeric(2),
point.type = as.numeric(19),
point.alpha = as.numeric(0.9),
median.line = as.logical(FALSE),
median.line.col = "black",
median.line.typ = "longdash",
median.line.size = as.numeric(1),
median.point.color = "dark gray",
border = as.logical(TRUE),
graph.margin = as.numeric(1),
annotate.all = as.logical(TRUE),
annotate.size = as.numeric(0.5),
annotate.color = "gray",
annotate.type = as.numeric(19),
annotate.alpha = as.numeric(1)
)
if (show) {
  tmp.att
} else {
  invisible(tmp.att)
}
}

# source("axis_opts_grid.R")

axis_opts_grid <- function(i, pl, a,
                           limsx = NA,
```

```

limsy = NA,
border = TRUE,
expx = FALSE) {
# if missing a background color specification, default to white
bgcolor <- ifelse(!is.na(a[[i]]$panel.bgcolor),
a[[i]]$panel.bgcolor,
"white"
)

# there is only one axis label size
label.size <- as.numeric(max(all_atts(a, "xaxis.labels.size")))) * 10

#####
#### x-axis title
#####

# start off with everything set clean
x.breaks <- x.labels <- x.limits <- x.expand <- FALSE
xstr.title <- "!!!"

if (!all(is.na(all_atts(a, "xaxis.title")))) {
  tmp.size <- max(as.numeric(all_atts(
    a,
    "xaxis.title.size"
  )) * 8
  tmp.lineheight <- as.numeric(all_atts(
    a,
    "xaxis.title.lineheight"
  ))
  tmp.lineheight <- tmp.lineheight[which.max(abs(tmp.lineheight - 1))]

  tmp.titles <- lapply(
    all_atts(a, "xaxis.title"),
    function(t) {
      if (is.na(t) | t == "") {
        t <- " "
      } else {
        t
      }
    }
  )
  tmp.title <- tmp.titles[[i]]
  ns <- max(unlist(lapply(
    tmp.titles,
    function(x) {
      length(strsplit(x, "\n")[[1]])
    }
  ))) -
}

```

```

length(strsplit(tmp.title, "\n")[[1]])
if (ns > 0) {
  tmp.title <- paste(tmp.title, rep(" \n ", ns), sep = "")
}
xstr.title <- paste("", tmp.title, "", sep = "")

pl <- pl +
  theme(
    axis.title.x =
      element_text(
        family = a[[i]]$xaxis.title.font,
        face = a[[i]]$xaxis.title.face,
        colour = a[[i]]$xaxis.title.color,
        size = tmp.size,
        lineHeight = tmp.lineheight
      )
    )
}

if (all(is.na(all_atts(a, "xaxis.title")))) {
  pl <- pl + theme(axis.title.x = element_blank())
}

### -----
### 

if (!any(is.na(limsx))) {
  x.limits <- TRUE
}
if (!expx) {
  x.expand <- TRUE
  xstr.expand <- as.character("", expand=c(0,0))
}

xstr.limits <- as.character(paste(
  "c(",
  min(limsx),
  ",",
  max(limsx),
  ")"
))
xstr.limits <- paste("", limits="", xstr.limits)

### -----
### Axis line display
### ----- ###

if (!a[[i]]$xaxis.line.display & !a[[i]]$yaxis.line.display) {
  pl <- pl + theme(axis.line = element_line(colour = bgcolor))
}

```

```

} else {
  pl <- pl + theme(axis.line = element_line(colour = "black"))
}

pl <- pl + theme(axis.ticks = element_blank())

##### -----
##### x-axis text display & text, labels, ticks
##### ----- ####

if (!any(all_atts(a, "xaxis.text.display"))) {
  pl <- pl + theme(axis.text.x = element_blank())
} else if (!a[[i]]$xaxis.text.display) {
  pl <- pl + theme(axis.text.x = element_text(
    colour = bgcolor,
    size = label.size
  ))
} else if (!is.na(unlist(a[[i]]$xaxis.labels)[1]) &
  !is.na(unlist(a[[i]]$xaxis.ticks)[1])) {
  tmpTheme <- "theme(axis.text.x = element_text(size=label.size"
  if (!is.null(a[[i]]$xaxis.labels.angle)) {
    tmpTheme <- paste(
      tmpTheme, ",",
      angle = "",
      a[[i]]$xaxis.labels.angle
    )
  }
  if (!is.null(a[[i]]$xaxis.labels.hjust)) {
    tmpTheme <- paste(
      tmpTheme, ",",
      hjust ="",
      a[[i]]$xaxis.labels.hjust
    )
  }
  if (!is.null(a[[i]]$xaxis.labels.vjust)) {
    tmpTheme <- paste(
      tmpTheme, ",",
      vjust ="",
      a[[i]]$xaxis.labels.vjust
    )
  }
  tmpTheme <- paste(tmpTheme, ""))
}

pl <- pl + eval(parse(text = tmpTheme))

x.breaks <- x.labels <- TRUE
xstr.breaks <- paste("",

breaks=c(",

```

```

make.string(a[[i]]$xaxis.ticks),
")",
sep = """
)
xstr.labels <- paste(",
           labels=c(",
make.string(a[[i]]$xaxis.labels),
")",
sep = """
)
} else if (!is.na(unlist(a[[i]]$xaxis.labels)[1]) |
  is.na(unlist(a[[i]]$xaxis.ticks)[1])) {
  print("Warning: both axis labels AND tick location must be specified")
}

#####
### scale_x_continuous
#####

xstr <- paste("scale_x_continuous(", xstr.title)
if (x.expand) {
  xstr <- paste(xstr, xstr.expand)
}
if (x.breaks) {
  xstr <- paste(xstr, xstr.breaks)
}
if (x.labels) {
  xstr <- paste(xstr, xstr.labels)
}
if (x.limits) {
  xstr <- paste(xstr, xstr.limits)
}
xstr <- paste(xstr, ")")

pl <- pl + eval(parse(text = xstr))

#####
## y-axis title
#####

# start off with everything set clean
y.breaks <- y.labels <- y.limits <- y.expand <- FALSE
ystr.title <- "!!!"
if (!all(is.na(all_atts(a, "yaxis.title")))) {

```

```

tmp.size <- max(as.numeric(all_atts(
  a,
  "yaxis.title.size"
))) * 8
tmp.lineheight <- as.numeric(all_atts(
  a,
  "yaxis.title.lineheight"
))
tmp.lineheight <- tmp.lineheight[which.max(abs(tmp.lineheight - 1))]

tmp.titles <- lapply(
  all_atts(a, "yaxis.title"),
  function(t) {
    if (is.na(t) | t == "") {
      t <- " "
    } else {
      t
    }
  }
)
tmp.title <- tmp.titles[[i]]
ns <- max(unlist(lapply(
  tmp.titles,
  function(y) {
    length(strsplit(y, "\n")[[1]])
  }
)) -
  length(strsplit(tmp.title, "\n")[[1]]))
if (ns > 0) {
  tmp.title <- paste(tmp.title, rep(" \n ", ns), sep = "")
}
ystr.title <- paste("""", tmp.title, """", sep = "")

pl <- pl +
  theme(
    axis.title.y =
      element_text(
        family = a[[i]]$yaxis.title.font,
        face = a[[i]]$yaxis.title.face,
        colour = a[[i]]$yaxis.title.color,
        size = tmp.size,
        lineheight = tmp.lineheight
      )
  )
}
if (all(is.na(all_atts(a, "yaxis.title")))) {
  pl <- pl + theme(axis.title.y = element_blank())
}

```

```

##### -----
##if (!any(is.na(limsy))) {
#  y.limits <- TRUE
#}
##if (!expx) {
#  y.expand <- TRUE
#  ystr.expand <- as.character("", expand=c(0,0))
#}
ystr.limits <- as.character(paste(
  "c(",
  min(limsy),
  ",",
  max(limsy),
  ")"
))
ystr.limits <- paste("", limits="", ystr.limits)

#####
#### axis line display
#### -----
##if (!a[[i]]$xaxis.line.display & !a[[i]]$yaxis.line.display) {
#  pl <- pl + theme(axis.line = element_line(colour = bgcolor))
#} else {
#  pl <- pl + theme(axis.line = element_line(colour = "black"))
#}
pl <- pl + theme(axis.ticks = element_blank())

#####
## y-axis text display & text, labels, ticks
#### -----
##if (!any(all_attr(a, "yaxis.text.display"))) {
#  pl <- pl + theme(axis.text.y = element_blank())
#} else if (!a[[i]]$yaxis.text.display) {
#  pl <- pl + theme(
#    axis.text.y =
#      element_text(
#        colour = bgcolor,
#        size = label.size
#      )
#  )
#} else if (!is.na(unlist(a[[i]]$yaxis.labels)[1]) &
#           !is.na(unlist(a[[i]]$yaxis.ticks)[1])) {

```

```

tmpTheme <- "theme(axis.text.y = element_text(size=label.size"
if (!is.null(a[[i]]$yaxis.labels.angle)) {
  tmpTheme <- paste(
    tmpTheme, ",",
    angle = "",
    a[[i]]$yaxis.labels.angle
  )
}
if (!is.null(a[[i]]$yaxis.labels.hjust)) {
  tmpTheme <- paste(
    tmpTheme, ",",
    hjust ="",
    a[[i]]$yaxis.labels.hjust
  )
}
if (!is.null(a[[i]]$yaxis.labels.vjust)) {
  tmpTheme <- paste(
    tmpTheme, ",",
    vjust ="",
    a[[i]]$yaxis.labels.vjust
  )
}
tmpTheme <- paste(tmpTheme, "")")

pl <- pl + eval(parse(text = tmpTheme))

y.breaks <- y.labels <- TRUE
ystr.breaks <- paste("", breaks = c(",
  make.string(a[[i]]$yaxis.ticks),
")",
sep = ""
)
ystr.labels <- paste("", labels = c(",
  make.string(a[[i]]$yaxis.labels),
")",
sep = ""
)
} else if (!is.na(unlist(a[[i]]$yaxis.labels)[1]) |
  !is.na(unlist(a[[i]]$yaxis.ticks)[1])) {
  print("Warning: both axis labels AND tick location must be specified")
}

#####
### scale_y_continuous
#####

ystr <- paste("scale_y_continuous(", ystr.title)

```

```

if (y.expand) {
  ystr <- paste(ystr, ystr.expand)
}
if (y.breaks) {
  ystr <- paste(ystr, ystr.breaks)
}
if (y.labels) {
  ystr <- paste(ystr, ystr.labels)
}
if (y.limits) {
  ystr <- paste(ystr, ystr.limits)
}
ystr <- paste(ystr, ")")

pl <- pl + eval(parse(text = ystr))

#####
## Grid Lines & vertical gap between scatter plots
#####

# some specific theme items different than the basic
pl <- pl + theme(
  panel.grid.major =
    element_line(
      linetype =
        a[[i]]$graph.grid.major.lty,
      colour =
        a[[i]]$graph.grid.major.col
    ),
  panel.grid.minor =
    element_line(
      linetype =
        a[[i]]$graph.grid.minor.lty,
      colour =
        a[[i]]$graph.grid.minor.col
    ),
  panel.margin = unit(
    a[[i]]$graph.margin,
    "lines"
  )
)

#####
## Grid's border
#####

borderx <- range(limsx) + c(1, -1) * diff(range(limsx)) * 0.001

```

```

bordery <- range(limsy) + c(0, -1) * diff(range(limsy)) * 0.001

facetNum <- unique(pl$data$pGrp)
if (a$median.row) {
  facetNum <- facetNum[facetNum != a$m.pGrp]
}

tmp.border <- data.frame(
  pGrp = rep(facetNum, each = 2),
  ymin = bordery[1],
  ymax = bordery[2],
  xmin = borderx[1],
  xmax = borderx[2]
)
if (border) {
  border.color <- a[[i]]$graph.border.color
} else {
  border.color <- NA
}

# draw a rectangle instead of a border
pl <- pl + geom_rect(
  aes(
    xmin = xmin,
    xmax = xmax,
    ymin = ymin,
    ymax = ymax
  ),
  data = tmp.border,
  colour = border.color,
  fill = NA
)

# set the border off
pl <- pl + theme(axis.line = element_blank())

pl
}

### read in the statistical data
# religion <- read.csv("Main Document-ReligionData.csv",
#                      header = TRUE)

religion <- read.csv(
  file = "data/China_ReligionData.csv",
  header = TRUE
)

### read in the map data

```

```

# ChinaShape <- readOGR("China_Provinces_Thinned",
#                         "ChinaThinned",
#                         verbose = TRUE)

ChinaShapefile <- sf:::as_Spatial(sf:::st_read("data/China_Shapefiles/export.shp"))

## Reading layer `export' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\China_Shapefiles\export.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 31 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 8176000 ymin: 704800 xmax: 15040000 ymax: 7087000
## Projected CRS: Google_Mercator

ChinaShapefileThin <- sf:::st_read("data/China_Shapefiles/export.shp") %>%
  st_simplify(dTolerance = 2000) %>%
  as_Spatial()

## Reading layer `export' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\China_Shapefiles\export.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 31 features and 7 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 8176000 ymin: 704800 xmax: 15040000 ymax: 7087000
## Projected CRS: Google_Mercator

# gIsValid(ChinaShape, byid = TRUE, reason = TRUE)
ChinaPolys <- create_map_table(ChinaShapefileThin, "ename")



mmpplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c(
    "labels",
    "dot_legend",
    "dot",
    "scatdot",
    "map"
  ),
  panel.data = list(
    "Province",
    NA,
    "Daoism",
    c("Christianity", "Buddhism"),
    NA
  )
)

```

```

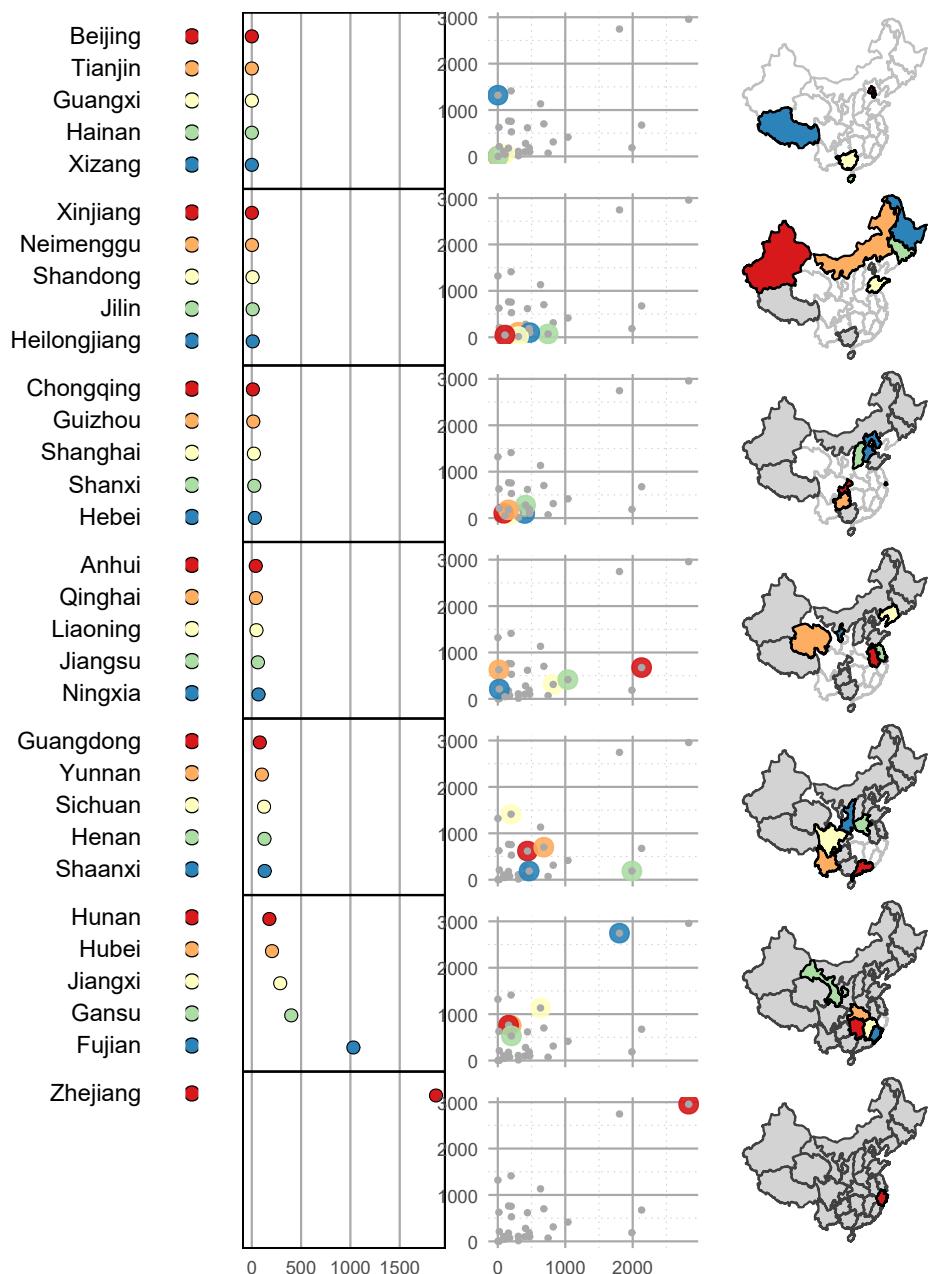
),
ord.by = "Daoism",
map.link = c("Province", "ID"),
grouping = 5
)

#### set tick mark sequence
Daoism.tick <- seq(from = 0, to = 2000, by = 1000)
Christ.tick <- seq(from = 0, to = 3000, by = 1000)
Buddhi.tick <- seq(from = 0, to = 3000, by = 1000)
Islam.tick <- seq(from = 0, to = 25000, by = 10000)
Total.tick <- seq(from = 0, to = 25000, by = 10000)

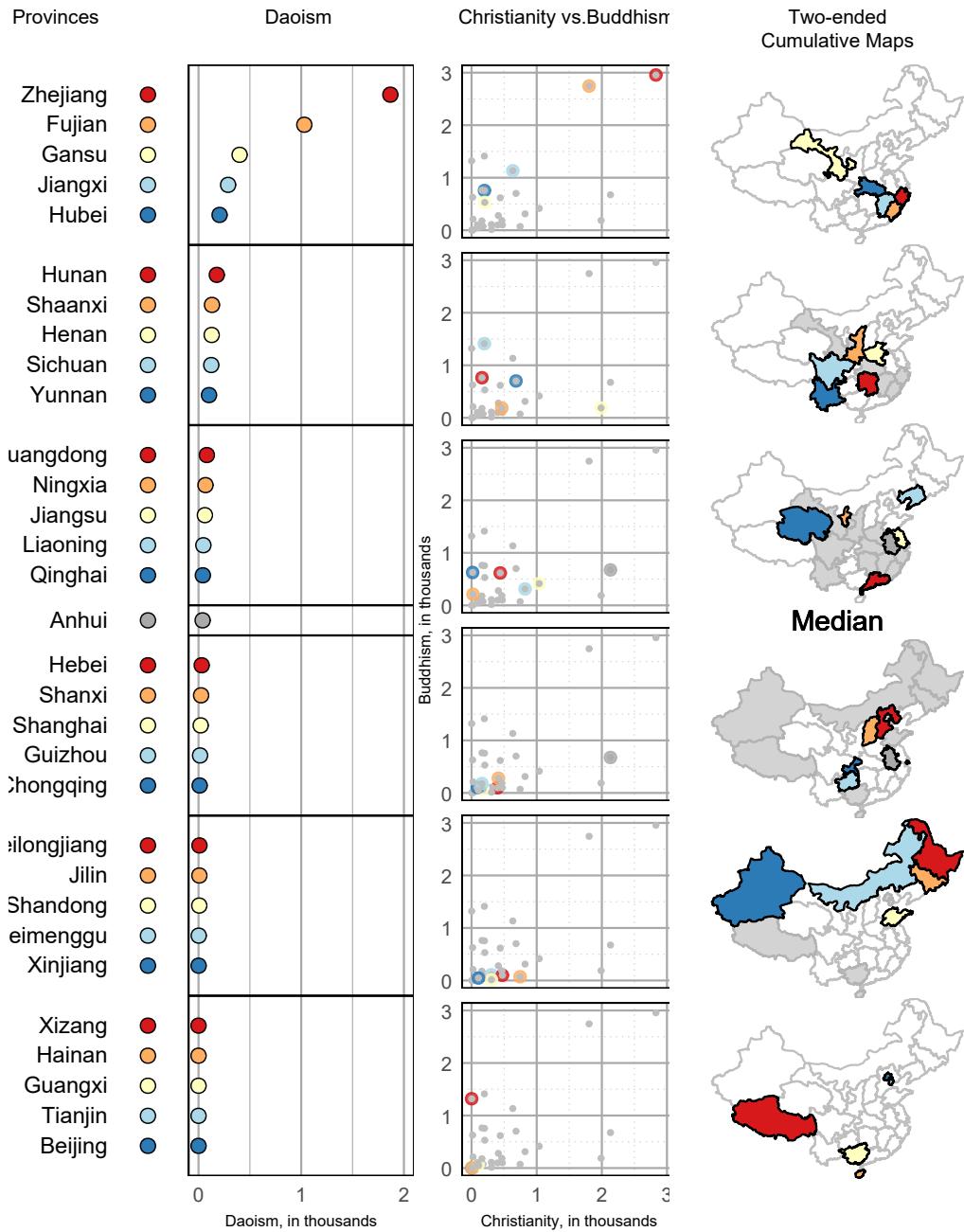
Total.tick.less <- seq(from = 0, to = 10000, by = 5000)
Islam.tick.less <- seq(from = 0, to = 4000, by = 1000)

mmpplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c(
    "labels",
    "dot_legend",
    "dot",
    "scatdot",
    "map"
  ),
  panel.data = list(
    "Province",
    NA,
    "Daoism",
    c("Christianity", "Buddhism"),
    NA
  ),
  ord.by = "Daoism",
  map.link = c("Province", "ID"),
  rev.ord = TRUE,
  grouping = 5,
  median.row = TRUE,
  median.color = "dark gray",
  colors = brewer.pal(5, "RdYlBu"),
  two.ended.maps = TRUE,
  map.all = TRUE,
  map.color2 = "lightgray",
  plot.header = "Religion in China",
  plot.header.size = 2,
  plot.header.color = "black",
  plot.panel.spacing = 0,
  plot.pGrp.spacing = 2,

```

**FIGURE 6.14** Scatterplot.

```
panel.att = list(
  list(
    1,
    header = "Provinces",
    panel.width = .5,
    align = "right",
    text.size = 1
  ),
  list(
    2,
    header = "",
    point.type = 20,
    point.size = 2
  ),
  list(
    3,
    header = "Daoism",
    point.type = 20,
    point.size = 2,
    graph.grid.minor = TRUE,
    xaxis.title = "Daoism, in thousands",
    xaxis.ticks = Daoism.tick,
    xaxis.labels = Daoism.tick / 1000
  ),
  list(
    4,
    header = "Christianity vs.Buddhism",
    point.type = 20,
    point.alpha = 0.8,
    left.margin = 1.6,
    xaxis.title = "Christianity, in thousands",
    xaxis.ticks = Christ.tick,
    xaxis.labels = Christ.tick / 1000,
    yaxis.title = "Buddhism, in thousands",
    yaxis.ticks = Buddhi.tick,
    yaxis.labels = Buddhi.tick / 1000,
    graph.margin = 0.5,
    border = TRUE
  ),
  list(
    5,
    header = "Two-ended\nCumulative Maps",
    inactive.border.color = gray(.7),
    inactive.border.size = 1,
    panel.width = 1.2
  )
)
```

**FIGURE 6.15** Scatterplot.

```

Paired5 <- brewer.pal(5, "Paired")[5:1]
Paired6 <- brewer.pal(6, "Paired")[6:1]
Divergent5 <- brewer.pal(5, "RdYlBu")

##### ===== ## 
##### Create a plot WITHOUT a median row, but TWO scatdot panels
##### ===== ## 

mmpplot(
  stat.data = religion[religion$Province != "Xinjiang", ],
  map.data = ChinaPolys,
  panel.types = c(
    "labels",
    "dot_legend",
    "dot",
    "scatdot",
    "scatdot",
    "map"
  ),
  panel.data = list(
    "Province",
    NA,
    "Total",
    c("Christianity", "Buddhism"),
    c("Daoism", "Islam"),
    NA
  ),
  ord.by = "Total",
  map.link = c("Province", "ID"),
  rev.ord = TRUE,
  grouping = 5,
  median.row = FALSE,
  # colors = Divergent5,
  colors = Paired5,
  plot.width = 10,
  plot.height = 10,
  two.ended.maps = TRUE,
  map.all = TRUE,
  map.color2 = "lightgray",
  plot.header = "Religion in China",
  plot.header.size = 2,
  plot.header.color = "black",
  plot.panel.spacing = 0,
  plot.pGrp.spacing = 2,
  panel.att = list(
    list(1,
         header = "Provinces",
         panel.width = .5,
         align = "right",

```

```
text.size = 1
),
list(2,
  header = "",
  point.type = 20,
  point.size = 2
),
list(3,
  header = "Total",
  panel.width = 0.75,
  point.type = 20,
  point.size = 2,
  graph.grid.minor = TRUE,
  xaxis.title = "Total, in thousands",
  xaxis.ticks = Total.tick.less,
  xaxis.labels = Total.tick.less / 1000
),
list(4,
  header = "Christianity vs. Buddhism",
  point.type = 20,
  # point.size = 3,
  point.alpha = 0.95,
  point.border = FALSE,
  left.margin = 1.5,
  # panel.width = 1.5,
  xaxis.title = "Christianity, in thousands",
  xaxis.ticks = Christ.tick,
  xaxis.labels = Christ.tick / 1000,
  yaxis.title = "Buddhism, in thousands",
  yaxis.ticks = Buddhi.tick,
  yaxis.labels = Buddhi.tick / 1000,
  graph.grid.major.col = "dark gray",
  graph.grid.minor.col = "dark gray",
  graph.margin = 0.5
),
list(5,
  header = "Daoism vs. Islam",
  point.type = 20,
  # point.size = 3,
  point.alpha = 0.95,
  point.border = FALSE,
  left.margin = 1.5,
  # panel.width = 1.5,
  xaxis.title = "Daoism, in thousands",
  xaxis.ticks = Daoism.tick,
  xaxis.labels = Daoism.tick / 1000,
  yaxis.title = "Islam, in thousands",
  yaxis.ticks = Islam.tick.less,
  yaxis.labels = Islam.tick.less / 1000,
```

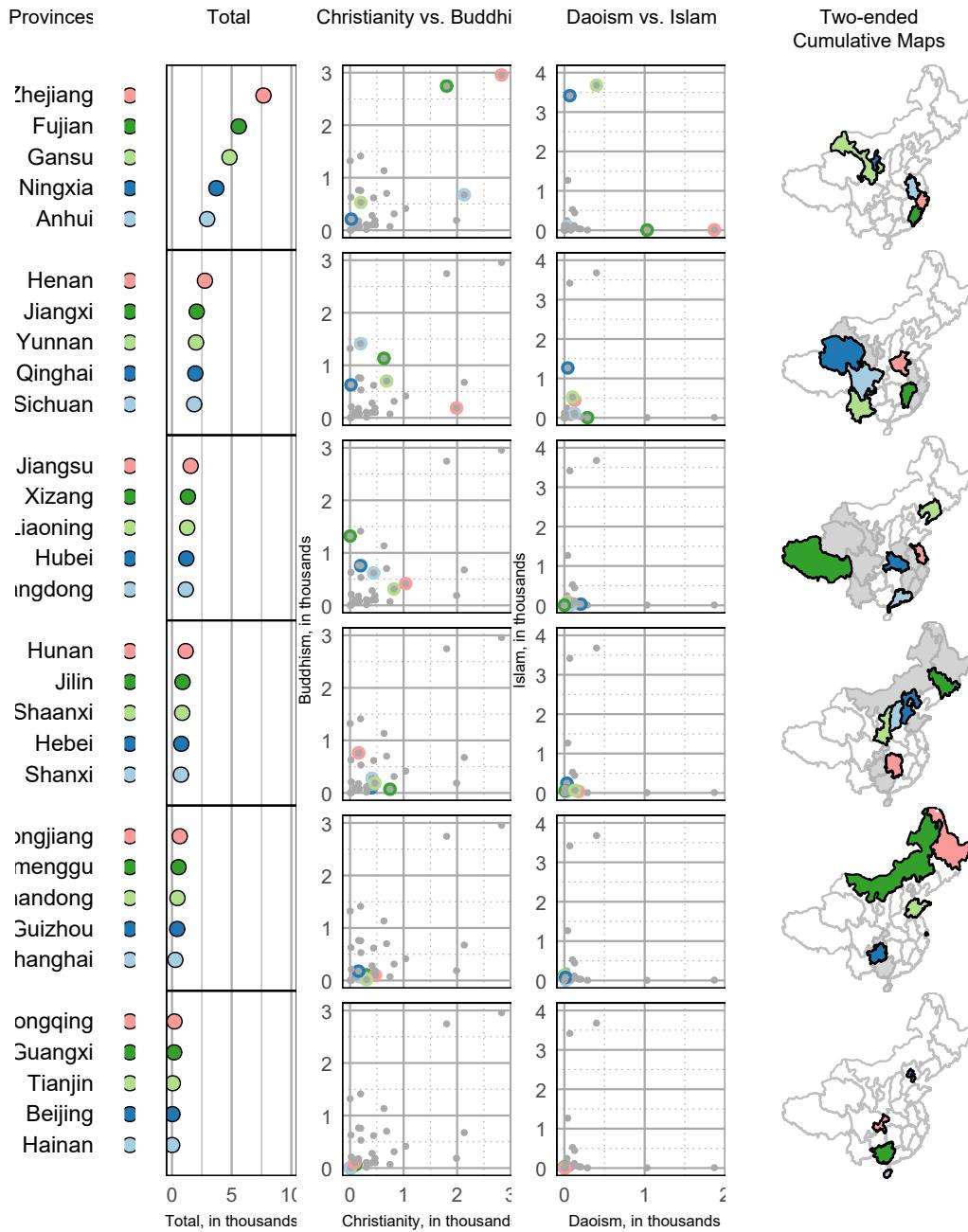
```

graph.grid.major.col = "dark gray",
graph.grid.minor.col = "dark gray",
graph.margin = 0.5
),
list(6,
  header = "Two-ended\nCumulative Maps",
  inactive.border.color = gray(.7),
  inactive.border.size = 1,
  panel.width = 1.2
)
)
)

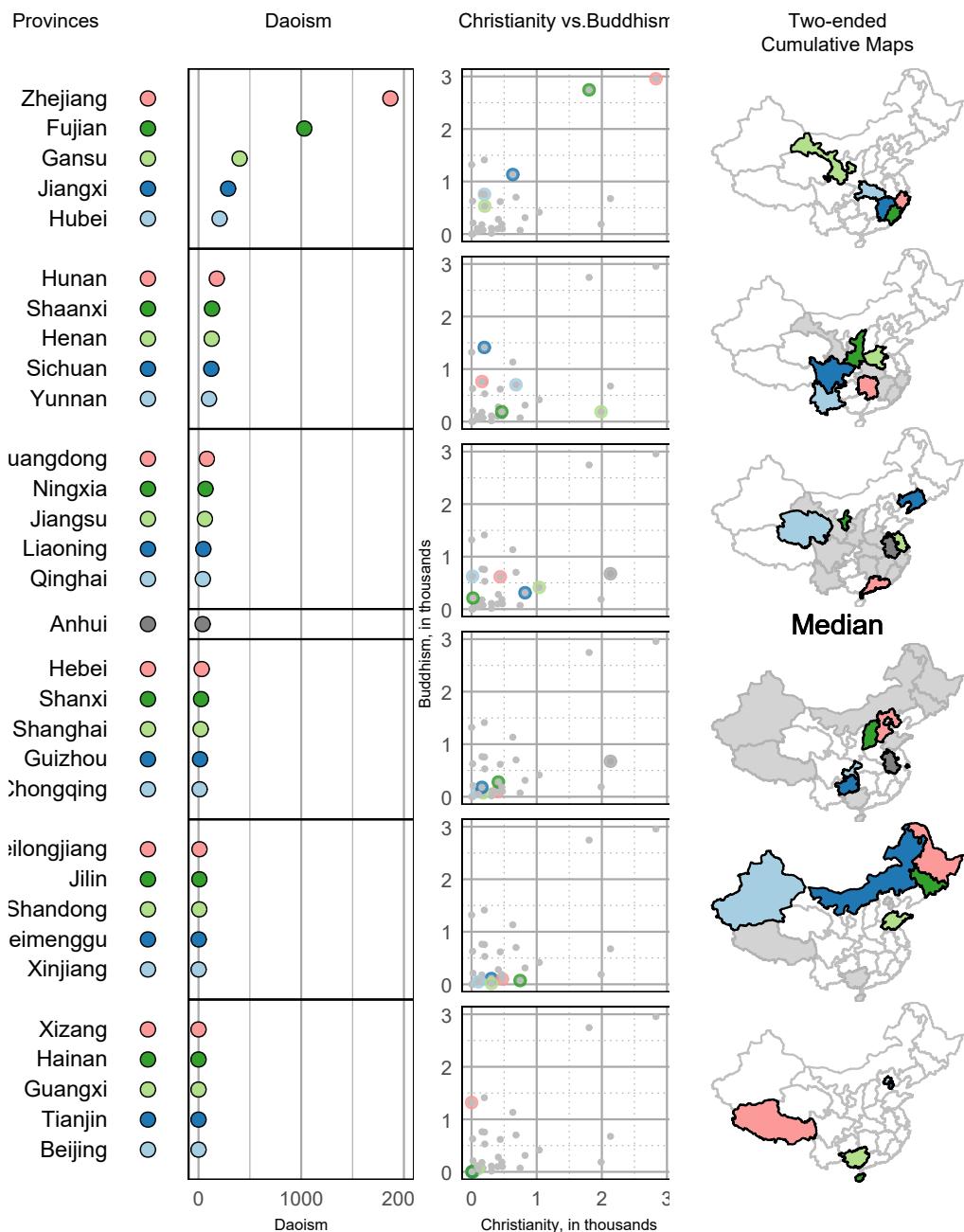
#### ===== #
#### Create a plot WITH a median row
#### ===== #

mmpplot(
  stat.data = religion,
  map.data = ChinaPolys,
  panel.types = c(
    "labels",
    "dot_legend",
    "dot",
    "scatdot",
    "map"
  ),
  panel.data = list(
    "Province",
    NA,
    "Daoism",
    c("Christianity", "Buddhism"),
    NA
  ),
  ord.by = "Daoism",
  map.link = c("Province", "ID"),
  rev.order = TRUE,
  grouping = 5,
  median.row = TRUE,
  colors = Paired5,
  plot.width = 10,
  plot.height = 10,
  two.ended.maps = TRUE,
  map.all = TRUE,
  map.color2 = "lightgray",
  plot.header = "Religion in China",
  plot.header.size = 2,
  plot.header.color = "black",
  plot.panel.spacing = 0,

```

**FIGURE 6.16** Scatterplot.

```
plot.pGrp.spacing = 2,
panel.att = list(
  list(1,
    header = "Provinces",
    panel.width = .5,
    align = "right",
    text.size = 1
  ),
  list(2,
    header = "",
    point.type = 20,
    point.size = 2
  ),
  list(3,
    header = "Daoism",
    point.type = 20,
    point.size = 2,
    graph.grid.minor = TRUE,
    xaxis.title = "Daoism",
    xaxis.ticks = Daoism.tick,
    xaxis.labels = Daoism.tick
  ),
  list(4,
    header = "Christianity vs.Buddhism",
    point.type = 20,
    # point.size = 2.5,
    point.alpha = 0.8,
    point.border = FALSE,
    left.margin = 1.6,
    # panel.width = 1.5,
    xaxis.title = "Christianity, in thousands",
    xaxis.ticks = Christ.tick,
    xaxis.labels = Christ.tick / 1000,
    yaxis.title = "Buddhism, in thousands",
    yaxis.ticks = Buddhi.tick,
    yaxis.labels = Buddhi.tick / 1000,
    graph.grid.major.col = "dark gray",
    graph.grid.minor.col = "dark gray",
    graph.margin = 0.5,
    border = TRUE
  ),
  list(5,
    header = "Two-ended\nCumulative Maps",
    inactive.border.color = gray(.7),
    inactive.border.size = 1,
    panel.width = 1.2
  )
)
```

**FIGURE 6.17** Scatterplot.

6.7 Main

TEXT

6.8 Summary and Further Reading

Introduce cross-references to other chapters, e.g., Chapter 1 and Chapter 2, where related work and further examples can be found in this book that match the content of this chapter, that follow up on this chapter, or that are a prerequisite of this chapter.

Also, do some scientific literature review here that is specific to your chapter. Where has this R package been introduced and used before, where have other plot types or different countries been used in micromaps, what were other applications of micromaps that are related to the title and content of your chapter, etc.?

References

- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.
- Symanzik, J., Dai, X., Weber, M. H., Payton, Q., & McManus, M. G. (2014). Linked Micromap Plots for South America — General Design Considerations and Specific Adjustments [<https://doi.org/10.15446/rce.v37n2spe.47949>]. *Revista Colombiana de Estadística*, 37(2), 451–469.

7

*Enhanced Glyphs Available in the **micromapST** R Package*

LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR., DANIEL B. CARR

The **micromapST** R package (Carr & Pearson Jr., 2015), accessible at <https://cran.r-project.org/web/packages/micromapST/index.html>, can be used for a large number of plot types that go beyond basic dot plots and bar charts. In this chapter the reader will learn how to create a linked micromap plot that makes use of these advanced plot types that are built into the **micromapST** R package itself.

7.1 Introduction

The meaning of the word glyph varies by discipline. In this statistical context, glyphs are graphical objects that encode statistics for visualization purposes. In Chapter 3, we showed how to create simple linked micromaps using the **micromapST** package . Glyphs used in those simple examples were dots, with and without confidence intervals , and horizontal bars. Another glyph used was a dashed vertical green line that encoded the U.S. value for a survey response. This reference line supports the common comparison of the individual area values with the overall one, e.g., state vs. U.S. percents.

This chapter introduces additional glyphs that are built into the **micromapST** package to support making a wider variety of plots. There are time series plots, arrow and scatter plots for encoding two variables, segmented bars showing multi-level survey responses and boxplots for comparing multiple distributions of values. These will be illustrated using the same data displayed in the Chapter 3 figures. In Table 7.1 all available glyphs are listed, along with the content of the various parameters required to define them.

TABLE 7.1 Glyphs available in micromapST and the specification of input data by column name, which indexes the input data frame. NA indicates that this column parameter is not used for that glyph.

Glyph Name	Meaning	col1	col2	col3	panelData
arrow	Arrow	Beginning values	Ending values (arrowhead)	NA	NA
bar	Horizontal bar	Bar end values	NA	NA	NA
segbar	Horizontal stacked bar	Values for first (leftmost) bar segment	Values for last (rightmost) bar segment	NA	NA
normbar	Horizontal stacked bar, normalized to total 100%	Same as segbar	Same as segbar	NA	NA
ctrbar	Horizontal stacked bar, centered on middle bar	Same as segbar	Same as segbar	NA	NA
boxplot	Horizontal box plot	NA	NA	NA	Name of output list from call to boxplot(..., plot = FALSE)
dot	Dot	Values for dots	NA	NA	NA
dotconf	Dot with confidence interval line	Values for dots	Values of lower limits	Values of upper limits	NA
dotse	Dot with line length of +/- standard error	Values for dots	Standard errors	NA	NA
scatdot	Scatter plot of dots	Values on horizontal (x) axis	Values on vertical (y) axis	NA	NA
ts	Time series (line) plot	NA	NA	NA	Name of array with dimensions c(51, t, 2) where t = # of time points, x values in [, , 1], y values in [, , 2]
tsconf	Time series (line) plot with confidence band	NA	NA	NA	Name of array with dimensions c(51, t, 4) as specified for ts with added lower limit in [, , 3], upper limit in [, , 4]

7.2 Time Series Data

The covid case rates used in Chapter 3 were for a single time point. We can plot the entire time series of 27 data points by specifying the `ts` glyph in the `type =` parameter within the `panelDesc` data frame. The only limitation to the number of time points that can be plotted is the detail that can be seen in a line graph as wide as the panel. We have plotted up to 200 points on a linked micromap plot with a single glyph panel.

As seen in the table, we first need to create an array of dimension `c(51, 27, 2)`, i.e., a matrix of 51 rows for states by 27 columns for the time points, with dates defined in `c(, , 1)` and the rates to be plotted in `c(, , 2)`. Below are the rate data for the first three states (Alaska (AK), Alabama (AL), Arizona (AZ)).

```
# set up input array with dates & rates
library(micromapST)
library(stringr)

st_covid_case_rates <- read.csv(
  "data/STCaseRatesPer1000.csv",
  row.names = 2
)
st_postal <- rownames(st_covid_case_rates)
temp_input <- st_covid_case_rates[, c(-1, -2, -3, -4)]
temp_rates <- data.frame(temp_input)
rownames(temp_rates) <- st_postal
head(temp_rates, n = 3L)

##   Feb_24_2020 Mar_25_2020 Apr_24_2020 May_24_2020 Jun_23_2020 Jul_23_2020
## AK        0       0.06      0.46     0.56     1.13     3.42
## AL        0       0.08      1.23     2.96     6.34    15.14
## AR        0       0.10      0.87     1.91     5.30    11.59
##   Aug_22_2020 Sep_21_2020 Oct_21_2020 Nov_20_2020 Dec_20_2020 Jan_19_2021
## AK       6.99     10.19     16.55     35.29     59.09    70.49
## AL      23.36     29.73     35.83     46.58     65.76    86.99
## AR      18.35     24.75     33.00     46.23     65.88    89.72
##   Feb_18_2021 Mar_20_2021 Apr_19_2021 May_19_2021 Jun_18_2021 Jul_18_2021
## AK      77.30     82.94     89.49     94.41     95.38    97.7
## AL      98.79    104.24    106.57    110.31    111.90   114.2
## AR     103.28    107.81    109.58    111.52    113.43   118.7
##   Aug_17_2021 Sep_16_2021 Oct_16_2021 Nov_15_2021 Dec_15_2021 Jan_14_2022
## AK     106.6     128.5     166.5     189.2     199.2    235.0
## AL     130.8     155.4     165.8     171.3     174.3    204.9
## AR     138.8     157.7     166.0     170.6     177.1    216.0
##   Feb_13_2022 Mar_15_2022 Apr_14_2022
## AK     300.7     313.2     320.2
## AL     256.9     263.0     264.6
## AR     258.2     265.3     267.5
```

First we need to convert the character-format dates to objects of class “Date”. The following

code defines the input column names as dates in the format “mmm_dd_yyyy”, where mmm, dd and yyyy represent the character month abbreviation, numeric day and 4-digit year, respectively, with underscores between them. These dates are represented internally as the number of days since January 1, 1970, although they are printed in the default date format.

```
# temp_dates is a properly formed date vector.
temp_dates <- as.Date(colnames(temp_rates),format="%b_%d_%Y")
temp_dates

## [1] "2020-02-24" "2020-03-25" "2020-04-24" "2020-05-24" "2020-06-23"
## [6] "2020-07-23" "2020-08-22" "2020-09-21" "2020-10-21" "2020-11-20"
## [11] "2020-12-20" "2021-01-19" "2021-02-18" "2021-03-20" "2021-04-19"
## [16] "2021-05-19" "2021-06-18" "2021-07-18" "2021-08-17" "2021-09-16"
## [21] "2021-10-16" "2021-11-15" "2021-12-15" "2022-01-14" "2022-02-13"
## [26] "2022-03-15" "2022-04-14"
```

Now we can build the necessary array. Following the code is a sample of data in the array: date and rate values for the first three states, for their first 8 time points.

```
# set up 3D array
temp_array2 <- array(
  dim = c(51, 27, 2),
  dimnames = list(st_postal)
)

# For each row (state), put dates in [, , 1] and numeric rates in [, , 2]
temp_array2[, , 1] <- rep(temp_dates, each = 51)
temp_array2[, , 2] <- sapply(temp_rates, as.numeric)

# setting the attribute of the dates defines how they will print on the axes
attr(temp_array2,"xIsDate") <- "%b-%y"      # <- TRUE gives the default yyyy-mm in numeric form
temp_array2[1:3, 1:8, ]
```

```
## , , 1
##
##    [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## AK 18316 18346 18376 18406 18436 18466 18496 18526
## AL 18316 18346 18376 18406 18436 18466 18496 18526
## AR 18316 18346 18376 18406 18436 18466 18496 18526
##
## , , 2
##
##    [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## AK     0 0.06 0.46 0.56 1.13 3.42 6.99 10.19
## AL     0 0.08 1.23 2.96 6.34 15.14 23.36 29.73
## AR     0 0.10 0.87 1.91 5.30 11.59 18.35 24.75
```

Note that the attribute function above (`attr`) defines how the dates will be printed on the time series axes. Here we request the format mmm-yy by `%b-%y`. See documentation for the `strptime` function for details on other possible formats.

Next we define the micromap plot (shown in Figure 7.1) using the postal abbreviations as the geographic identifier and sorting by the midpoint of the date range (March 2021) in descending order:

```
library(micromapST)

panel_desc_timeseries <- data.frame(
  type = c("map", "id", "ts"),
  lab1 = c("", "", "Cases/1000"),
  lab2 = c("", "", "Feb 2020–Apr 2022"),
  lab3 = c("", "", "Month-Year"),
  panelData = c(NA, NA, "temp_array2")
)

# sort by midpoint of dates (= March 20, 2021)
micromapST(
  statsDFrame = temp_rates,
  panelDesc = panel_desc_timeseries,
  rowNames = "ab",
  sortVar = "Mar_20_2021",
  ascend = FALSE,
  plotNames = "full",
  title = c("Cumulative covid rates by month, Feb 2020 – Apr 2022")
)
```

In Figure 3.2, we noted the high covid case rates in North and South Dakota in November, 2020. Now, the full time series lines show that these two states had case increases by September, consistent with the hypothesis of a superspread event there in August. We can also see the bump up in cases in early 2022 due to the rapid spread of a new covid variant.

This full time series line plot contains a great deal of information but might be too complex for the intended audience to understand. We can simplify the plot by replacing the lines by two arrows, one for the first half of the period and one for the second half.

Additional information may be gained by examining the distribution of case rates by county within each state. The boxplot gives a convenient summary of the county rates for this purpose. The first three records of the input to the boxplot, shown here, are for Alabama counties:

```
st_covid_case_rates <- read.csv(
  "data/STCaseRatesPer1000.csv",
  row.names = 2
)
cty_covid_case_rates <- read.csv(
  "data/cnty_cRates_04_14_2022per1000.csv",
  row.names = 1
)
head(cty_covid_case_rates, 3L)
```

##	County	st_postal	Days	Rates
1	Adair	IA	14	1000
2	Albion	IA	14	1000
3	Appanoose	IA	14	1000

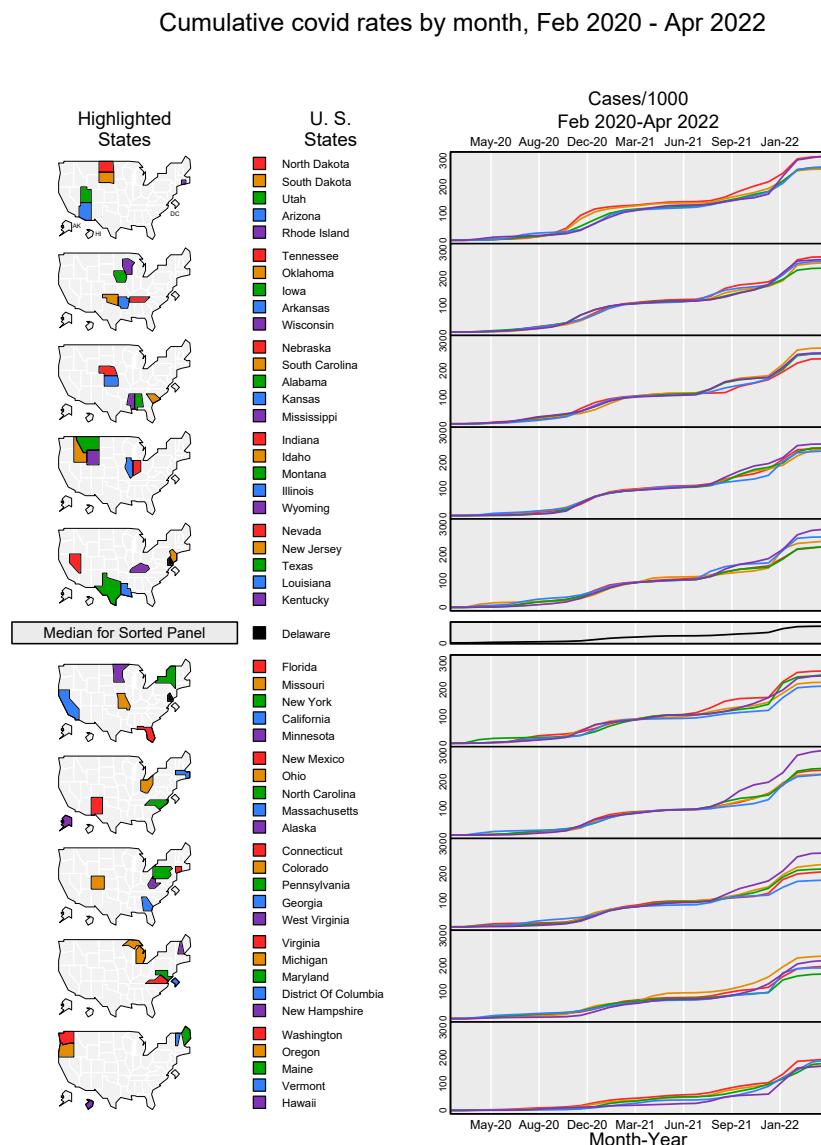


FIGURE 7.1 Time Series plots of cumulative covid case rates by month, Feb 2020 - Apr 2022, sorted by March 2021 (median date) rates.

```
## 1001 Autauga County      AL Apr_14_2022 281.9
## 1003 Baldwin County     AL Apr_14_2022 248.8
## 1005 Barbour County     AL Apr_14_2022 229.2
```

To create the boxplot information for the linked micromap plot, call the `boxplot` function but save its output instead of printing it (`plot = FALSE`). For our example, we categorize (`split`) the county rates by state, resulting in boxplot parameters being calculated for every state. Results for the first 3 states (Alaska, Alabama, Arkansas) are shown below; each column of `covid_box_list$stats` contains the 5-value summary for each state (minimum, quartiles, maximum). Outliers are saved in `covid_box_list$out` and are indexed by `covid_box_list$group`. For example, the first five outliers on the file are for groups 1, 2, 2, 3 and 3, corresponding to states AK, AL, AL, AR and AR, respectively.

```
covid_box_list <- boxplot(split
  cty_covid_case_rates$Rates,
  cty_covid_case_rates$st_postal
),
plot = FALSE
)
# print boxplot names, stats, outliers for 1st 3 states
covid_box_list$names[1:3]
```

```
## [1] "AK" "AL" "AR"
```

```
covid_box_list$stats[, 1:3]
```

```
##      [,1]  [,2]  [,3]
## [1,] 0.0 195.4 169.2
## [2,] 152.5 239.1 238.1
## [3,] 264.6 263.7 263.1
## [4,] 350.9 282.1 285.6
## [5,] 605.0 321.9 350.3
```

```
covid_box_list$group[1:5]
```

```
## [1] 1 2 2 3 3
```

```
covid_box_list$out[1:5]
```

```
## [1] 727.3 162.8 173.6 365.0 377.1
```

Now that the boxplot values have been computed outside the package, we just need to include the name of the matrix with those values in the `panelData` parameter, as shown in Table 7.1. To help interpret the boxplot results, `refvals` and `refTexts` specify the U.S. overall case rate and text label, respectively, for a vertical reference line . By default, this is a green dashed line.

```
library(micromapST)
```

```

panel_desc_arrow_boxplot <- data.frame(  

  type = c("map", "id", "arrow", "arrow", "boxplot"),  

  lab1 = c( "", "", "State rates", "State rates", "County rates Apr 22"),  

  lab2 = c( "", "", "Trend Feb 20-Mar 21", "Trend Mar 21-Apr 22", "(suppressed if 1-9 deaths)"),  

  lab3 = c( "", "", "Cases/1000", "Cases/1000", "Cases/1000"),  

  col1 = c(NA, NA, "Feb_24_2020", "Mar_20_2021", NA),  

  col2 = c(NA, NA, "Mar_20_2021", "Apr_14_2022", NA),  

  refVals = c(NA, NA, NA, NA, 238),  

  refTexts = c(NA, NA, NA, NA, "US rate"),  

  panelData = c( "", "", "", "", "covid_box_list")  

)  
  

micromapST(  

  statsDFrame = st_covid_case_rates,  

  panelDesc = panel_desc_arrow_boxplot,  

  sortVar = "Mar_20_2021",  

  ascend = FALSE,  

  title = c("Covid rates: State time trend arrows and county boxplots")  

)

```

rowNames value used is : ab

The set of arrows for the first half of the time period is consistent with the full time series plot, showing that North and South Dakota had the highest case rates from February 2020 to March 2021, although we lose the more precise estimate of when their rates started to increase. The second set of arrows shows the increase during the second half of the time period. Note that some states that had the greatest increase in this later period, as evidenced by their longer arrows, were different from the states with the greatest increase earlier. For example, see Rhode Island (RI), Kentucky (KY), Alaska (AK) and West Virginia (WV).

The box plot values are displayed in a typical format. The box encompasses the middle 50% of the data, with a short vertical bar at the median. Outliers are indicated by open circles in the state's color. The box plots in Figure 7.2 show that in general the county variation within state is small, but there are some county outliers, possibly due to small populations. For example, the most extreme outlier is Loving County TX, the least populous county in the U.S.

7.3 Scaled Response Data

Many surveys collect responses that have a rank order but are not strictly quantifiable. For example, the Likert scale (Likert, 1932) is a commonly-used set of symmetric responses for the respondent's agreement or disagreement with a series of questions; a five-category scale would typically be “Strongly disagree”, “Disagree”, “Neither agree nor disagree”, “Agree” and “Strongly agree”. The number of responses for each of these choices can be compared across categories of respondents, geographic areas, etc.

micromapST can display these sorts of data in several ways: segmenting a full bar into component parts, normalized to 100% total bar length or centered on a particular cutpoint.

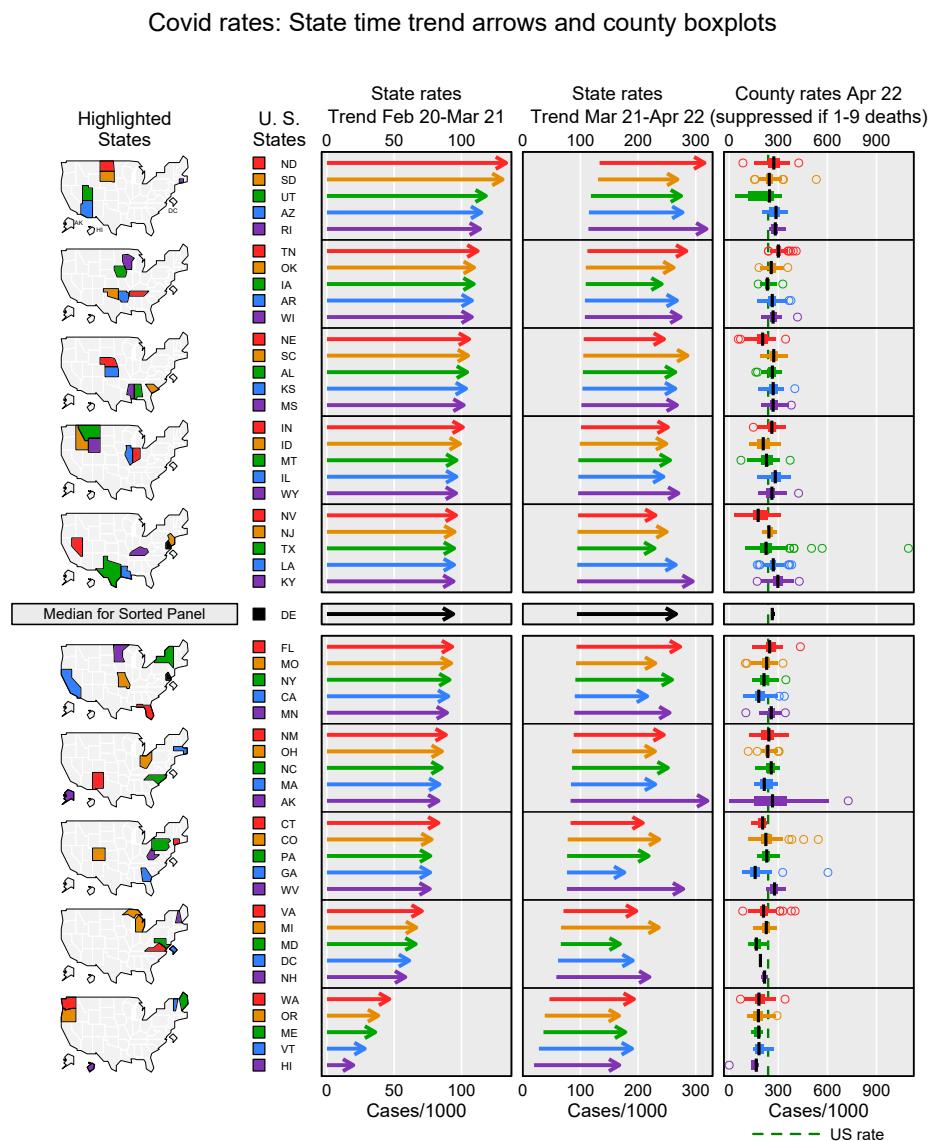


FIGURE 7.2 Time Series arrow plots of cumulative covid case rates, Feb 2020 - Mar 2021 and Mar 2021 - Apr 2022, and county rate boxplots.

Up to nine bar segments (response categories) can be displayed. The bar segments are colored in light to dark shades of the hue assigned to each geographic unit. To illustrate, we compare the math results for eighth-graders from the 2011 National Assessment of Educational Progress, standardized tests given to students in all states (National Center for Education Statistics, 2011).

In Figure 7.3, the first column plots the average math scores for students in each state. The second column displays individual student scores categorized into proficiency classes: “Less than basic”, “Basic”, “Proficient” and “Advanced”. Except for a few missing scores, each state’s total is nearly 100%, so the basic horizontal stacked bar design (`type = "segbar"`) is used. No scaling to 100% (`type = "normbar"`) is needed. The states are sorted in ascending order of the average math scores.

```
library(micromapST)
# Figure - dots and centered bar (source of code sample for section 4.7)
```

```
educ_8th_data <- read.csv(
  "data/MathProficiency8thGr2011.csv",
  row.names = 1
)
# columns = State abbrev, State name, Avg Score, 4 percents as above
head(educ_8th_data)
```

	State	avgscore	PctBelowBasic	PctAtBasic	PctProficient	PctAdvanced
## AK	Alaska	283	26	39	28	7
## AL	Alabama	269	40	40	17	3
## AR	Arkansas	279	30	41	24	5
## AZ	Arizona	279	32	37	24	7
## CA	California	273	39	36	19	6
## CO	Colorado	292	20	37	31	12

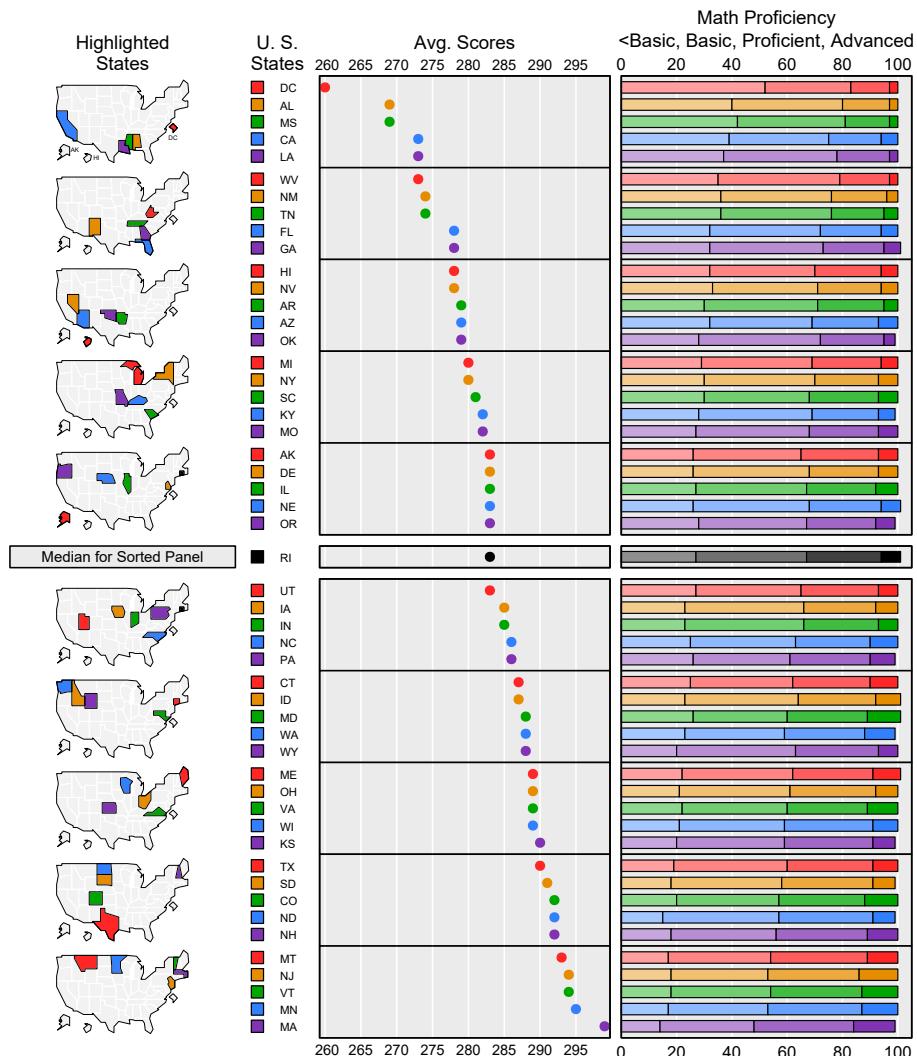
```
panel_desc_math_segbar <- data.frame(
  type = c("map", "id", "dot", "segbar"),
  lab1 = c("", "", "Avg. Scores", "Math Proficiency"),
  lab2 = c("", "", "<Basic, Basic, Proficient, Advanced"),
  col1 = c(NA, NA, "avgscore", "PctBelowBasic"),
  col2 = c(NA, NA, NA, "PctAdvanced")
)

micromapST(
  statsDFrame = educ_8th_data,
  panelDesc = panel_desc_math_segbar,
  sortVar = "avgscore",
  title = "Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade"
)
```

```
## rowNames value used is : ab
```

The order of states ranges from Washington DC as worst to Massachusetts as best on the math tests that year, as measured by both average score and the percent of students with less than basic proficiency. (Of course, DC is a totally urban city and is not fairly compared

Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade

**FIGURE 7.3** Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade, measured by average scores and achievement category.

to all other states which are a mix of urban, suburban and rural areas. However, this is how the data are reported.)

Except for the leftmost bar (“< Basic”) that has its leftmost edge at 0, it is difficult to compare the lengths of the other category bars because they are not aligned with each other in any way. Heiberger and Robbins (2014) suggested that segmented (stacked) horizontal bars be centered on the middle category so that the reader can judge the lengths of the segments to the right and to the left of center, a task that most people can easily do (Cleveland & McGill, 1984). The `type = "ctrbar"` option in **micromapST** will do this, centering at the midpoint of the middle bar if there is an odd number of bars and at the segment boundary dividing an even number of bars in half. Here we add a label at the bottom of the column to aid the reader’s interpretation of this double-ended bar (“% less than proficient | % at least proficient”). The scale is automatically relabeled to reflect the centering.

```
library(micromapST)
# Figure - dots and centered bar (source of code sample for section 4.7)

panel_desc_math_ctrbar <- data.frame(
  type = c("mapcum", "id", "dot", "ctrbar"),
  lab1 = c("", "", "Avg. Scores", "Math Proficiency"),
  lab2 = c("", "", "<Basic, Basic, Proficient, Advanced"),
  lab3 = c("", "", "% less than proficient | % at least proficient"),
  col1 = c(NA, NA, "avgscore", "PctBelowBasic"),
  col2 = c(NA, NA, NA, "PctAdvanced")
)

micromapST(
  statsDFrame = educ_8th_data,
  panelDesc = panel_desc_math_ctrbar,
  sortVar = "avgscore",
  title = "Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade"
)

```

rowNames value used is : ab

Centering between the second and third categories leads to the interpretation of the total length of the bars to the left and right of center as the percent of students who were less than proficient and at least proficient in eighth grade math, respectively. Centering in this way shows DC in a better light - although its average score is worst, its percent of students who were not proficient in math (represented by the length of the bar to the left of zero) was not much different from the next seven poorly-ranked states.

Adding cumulative shading to this plot highlights the generally lower performance of students in southern states compared to northern states. The top four panels, representing the 20 lowest ranking states, include 15 states in the southern half of the country. The original report (National Center for Education Statistics, 2011) presented these results in a segmented bar chart centered in the same way as in Figure 7.4, but sorted by state name. No geographic patterns could be discerned.

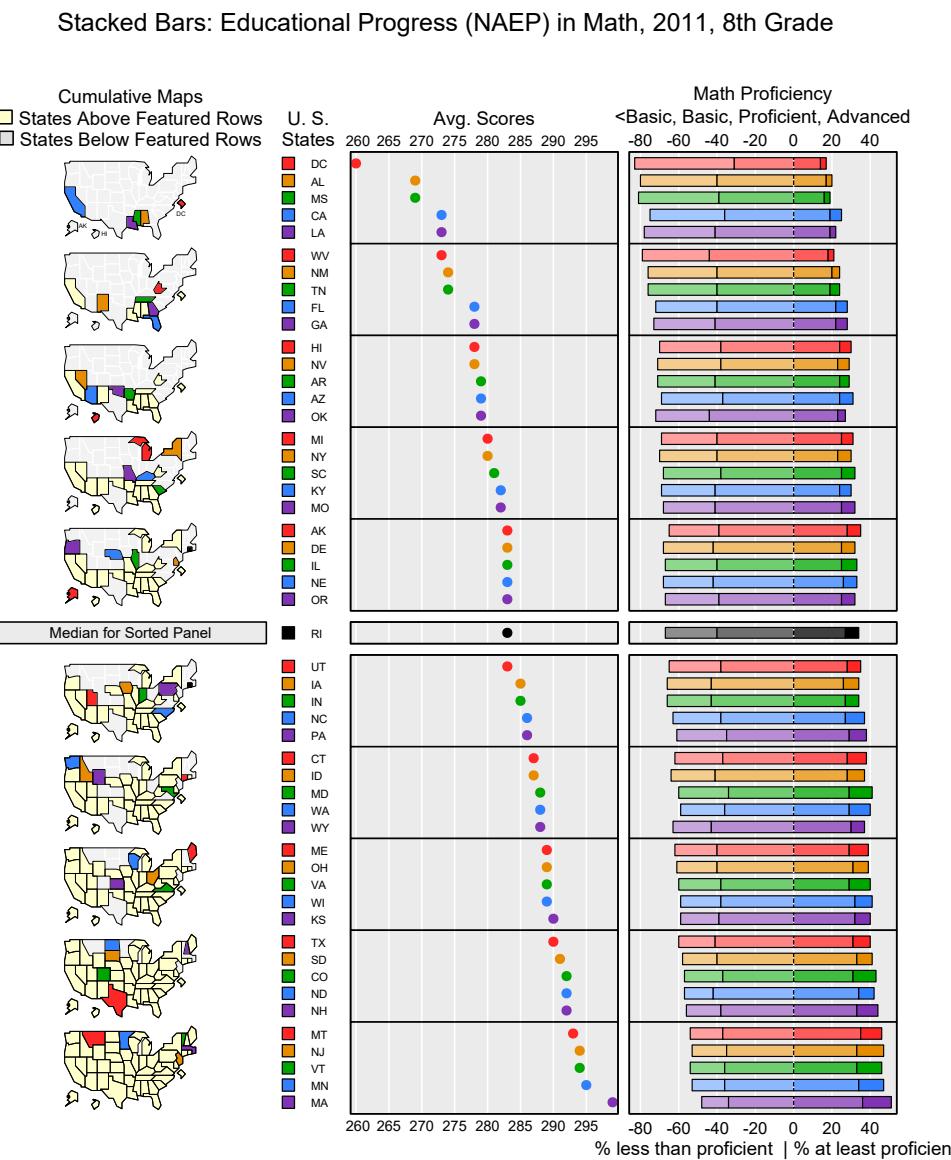


FIGURE 7.4 Stacked Bars: Educational Progress (NAEP) in Math, 2011, 8th Grade, centered at Not Proficient vs. At Least Proficient.

7.4 Scatter Plots

One of the simplest types of graphs is the two-dimensional scatter plot, which displays pairwise values, usually denoted (x, y). However, this graphic presents challenges in the linked micromap design - each state's micromap row is too small to display a two-dimensional plot and, even if we could scale such a plot to be visible, we would lose the information about where that state's value fits within the entire bivariate distribution of all states. The solution implemented in **micromapST** is to display a scatter plot for each panel, i.e., for five states in the default design. Each scatter plot contains dots for every state, with only those dots representing the states in that panel colored in. The colors for the state dots match the colors for other column glyphs, the state name key and on the map, just as in every other micromap design.

To simplify our example, we will plot the same data used in Section 3.4, Maryland counties' percent of residents living below 150% of the federal poverty level versus the percent of residents with at least four years of college education (The Nation's Report Card, 2022) . In Chapter 3, we compared two parallel columns of dots of these values, looking for an inverted "V" shape that would indicate negative correlation between them. Here in Figure 7.5, we can judge correlation more easily on the full scatter plot. Axis labels for the two dimensions of the scatter plot are specified in `lab3` and `lab4`. By default, a white line represents equal x and y values on the plot. However, we will overlay each plot with a locally-weighted scatterplot smoothing line (LOWESS; (Cleveland, 1981)), new in the 2025 package.

```
library(micromapST)

# Input data files.
# Data Set # 1
MD_pov_ed <- read.csv(
  "data/MDPovEducACSData20162020.csv",
  row.names = 3
)
head(MD_pov_ed, n = 3L)

##          CountyName StateName PopTotal PopMarginOfErr Pop25Up Pop25MarginErr
## 24001      Allegany Maryland    62610            538     44371            387
## 24003 Anne Arundel Maryland    558904           1489    388227            912
## 24510 Baltimore City Maryland    580311           1057    414275            790
##          NumLessHS NumHS NumSomeColl NumBachDegree NumLess125Pov NumLess150Pov
## 24001      3672 17411       13856        9432     13376       16509
## 24003      24922 86242       106991      170072     42915       54604
## 24510      59018 116928       100536      137793     144814      171279
##          PctLessHS PctHS PctSomeColl PctBachDegree PctLess125Pov PctLess150Pov
## 24001      0.083 0.392       0.312       0.213      0.214       0.264
## 24003      0.064 0.222       0.276       0.438      0.077       0.098
## 24510      0.142 0.282       0.243       0.333      0.250       0.295

panel_desc_MD_poved <- data.frame(
  type = c("mapcum", "id", "scatdot"),
```

```

lab1 = c("", "", "% College by % Pov"),
col1 = c(NA, NA, "PctBachDegree"),
col2 = c(NA, NA, "PctLess150Pov"),
lab3 = c(NA, NA, "% college"),
lab4 = c(NA, NA, "% poverty"),
parm = I(list(NA, NA, list(line = "LOWESS", line.lwd = 2)))
)

# sort by % college
micromapST(
  statsDFrame = MD_pov_ed,
  panelDesc = panel_desc_MD_poved,
  rowNames = "id",
  sortVar = "PctBachDegree",
  ascend = FALSE,
  plotNames = "full",
  bordGrp = "MarylandBG",
  title = c("Maryland counties: % Living < 150% of Poverty Level and % with 4+ Years of College")
)

```

This gives a clearer picture of the relationship between poverty and education in Maryland. Howard and Montgomery counties are outliers with high percents of college-educated residents. Excluding these outliers, the other counties do show a fairly strong negative correlation between poverty and college education, an association highlighted by the smoothing line.

7.5 Summary and Further Reading

In summary, **micromapST** can produce glyphs for most common types of data:

- dots and horizontal bars for single values,
- arrows and scatter plots for bivariate values,
- boxplot distributions of multiple values,
- segmented bars for ranked survey responses,
- time series line plots.

For each of these types of glyphs, there are options to control labeling and other features of the plot. Confidence intervals and/or error bars can be added to the dot and bar glyphs. Carr and Pickle's book (Carr & Pickle, 2010) provides extensive examples applying these glyphs to different data and the cognitive rationale of the design choices. **micromapST** can readily produce publication-quality linked micromaps, letting the user focus more on the content of the graph and less on its aesthetics.

Maryland counties: % Living < 150% of Poverty Level and % with 4+ Years of College

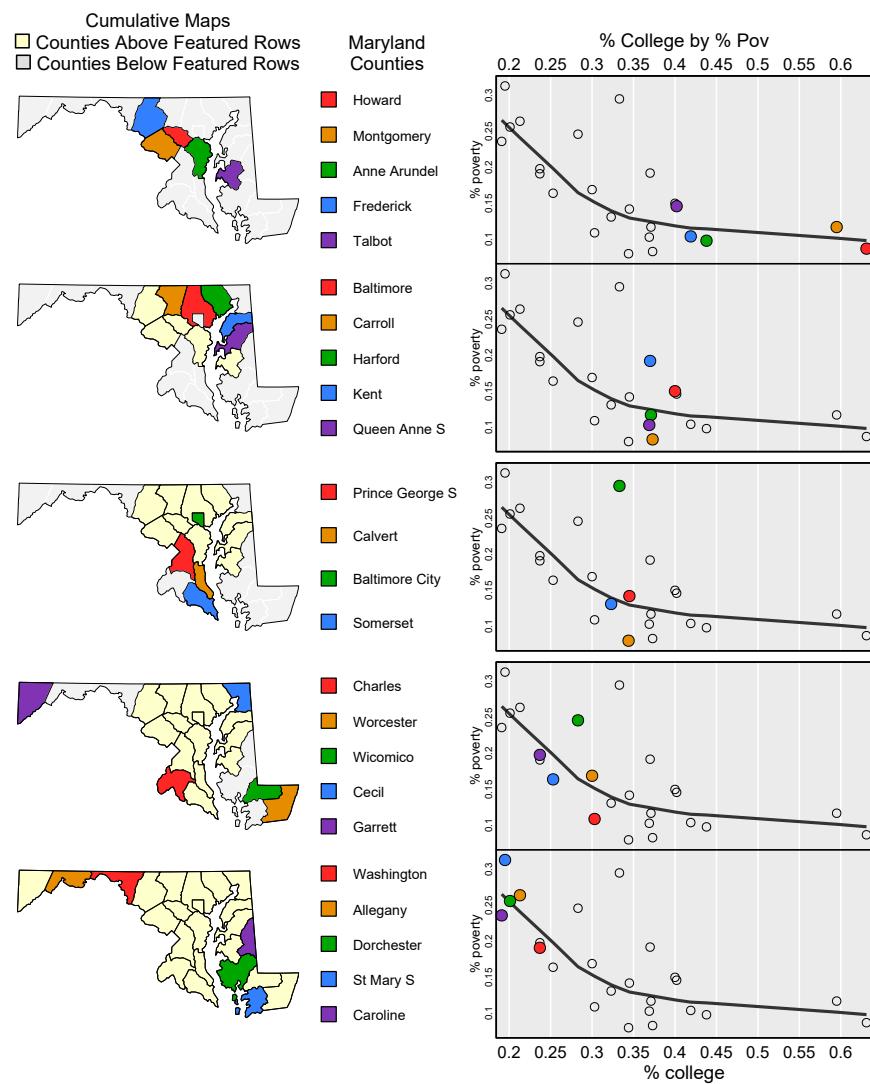


FIGURE 7.5 Scatter plot of percent of residents with income < 150% federal poverty level and percent with 4+ years of college, sorted by percent who attended college 4+ years.

References

- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Cleveland, W. S. (1981). LOWESS: A Program for Smoothing Scatterplots by Robust Locally Weighted Regression [<https://doi.org/10.2307/2683591>]. *The American Statistician*, 35(1), 54.
- Cleveland, W. S., & McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods [<https://doi.org/10.1080/01621459.1984.10478080>]. *Journal of the American Statistical Association*, 79(387), 531–554.
- Heiberger, R. M., & Robbins, N. R. (2014). Design of Diverging Stacked Bar Charts for Likert Scales and Other Applications [<https://doi.org/10.18637/jss.v057.i05>]. *Journal of Statistical Software*, 57(5), 1–32.
- Likert, R. (1932). A Technique for the Measurement of Attitudes [https://legacy.voteview.com/pdf/Likert_1932.pdf]. *Archives of Psychology*, 140, 5–55.
- National Center for Education Statistics. (2011). *The Nation's Report Card: Mathematics 2011 (NCES 2012-458)* [<https://nces.ed.gov/nationsreportcard/pdf/main2011/2012458.pdf>]. Institute of Education Sciences, U.S. Department of Education, Washington, D.C.
- The Nation's Report Card. (2022). Data Tools: NAEP Data Explorer [Web Page, <https://www.nationsreportcard.gov/ncores/xplore/NDE> (accessed May 4, 2022)].

8

Linked Micromap Plots for Point Locations

MARTIN HOLDREGE, JÜRGEN SYMANZIK

A few stand-alone attempts have been made in the past to create linked micromap plots for point locations such as climate stations, cities, or locations on a baseball field, rather than for areal locations (i.e., polygons). The creation of linked micromap plots for point locations requires that the point location is extended to a small (circular or quadratic) area that can be color-coded in a linked micromap plot. This chapter provides an overview of necessary steps to produce linked micromap plots for point locations to create areas of suitable sizes and to avoid overplotting of nearby point locations.

8.1 Introduction: The Challenge of Displaying Point Data Using Micromaps

One obstacle to creating micromaps for point locations is that the user must create polygons to represent the point locations. The more standard use of micromaps, displays information associated with geographic areas for which the associated spatial data likely already exists. In the case of point locations, points need to be displayed on the map as circles (or other shape). The coordinates of a given point location become the center of the circle. Once the polygons are created, maps and accompanying data can be visualized using the **micromap** (Payton & Olsen, 2015) package, which requires that each entity for which data is displayed is a polygon on the map.

When creating the polygons (circles) for display on a micromap, they need to be appropriately sized and should not overlap too much with each other. The locations of points that are very close to each other may need to be adjusted so they can be distinguished on the micromap. Lastly, to create more useful micromaps, the polygons representing the point locations also need to be integrated into a base map that shows relevant geographic boundaries.

As with more traditional micromaps, values from multiple variables can be shown for each location. Additionally, as with other micromaps, there is a space constraint, and generally, it is challenging to effectively display more than roughly 50 point locations in one micromap..

It is also important to first consider whether a micromap of point locations is the appropriate visualization for the data that you're trying to display. You should consider whether the points represent a broader geographic area, which could be represented by larger polygons, as is done in more traditional micromaps presented in the previous chapters. For example, in Section 8.4 we provide an example where we plot data associated with County Seats in the state of Utah. In that case there is one point location (city) per county, so it would also

be possible to make an effective micromap where instead of showing the point locations, the corresponding counties are shown, which would potentially require fewer steps to create.

8.2 Outline of Steps to Create Linked Micromap Plots for Point Locations

Broadly, the following steps need to be taken to display point locations on micromaps.

1. Obtain coordinates of the point locations of interest along with the corresponding data to display.
2. Obtain geographic data for the underlying base map. Strictly speaking, the micromaps of point locations could be shown without a base map but in most cases we think that would make a less useful visualization because there would be no geographic context.
3. Create a simple map-based plot of the point locations. The locations of points that are close to each other may need to be adjusted. If points are very close together, they may overlap on the final map, making them hard or impossible to distinguish from each other. The trade off here is that adjusting point locations does lead to some geographic distortion. We provide examples of both manual and semi-automated approaches to adjusting point locations.
4. Create polygons around the adjusted point locations. This step is necessary because the **micromap** package was designed to show polygons, not points, on maps. Thus, each point becomes a small polygon that is then visible in the resulting micromaps. Multiple R packages offer ways to create buffers (polygons) around points, and the `points2circles()` function (described below) provides a convenient wrapper to do this.
5. Combine the base map and the polygons representing point locations into a single spatial object. This step may require re-projecting one or both spatial data sets into a common projection. The resulting object contains polygons that will have data associated with them (i.e., the polygons showing the point locations), as well as polygon(s) that form the background of the map.
6. As described in Section 2.3.2, the `micromap::mmpplot()` function, which creates the linked micromap plot, requires a specially formatted data frame as input. Therefore, convert the object created in the previous step (which may be, for example, an object from the **sf** package (Pebesma, 2018)) into such a data frame and plot it.
7. Create an initial draft linked micromap plot as outlined in Section 2.3.3. To make troubleshooting easier, we recommend that you initially do not add all visualization elements (e.g., error bars, labels, multiple variables).
8. Iterate until the size of the circles representing point locations and their level of overlap is acceptable (see Steps 3 to 7).
9. As necessary, add additional visualization elements including labels and titles, and adjust dimensions of the linked micromap plot until you are satisfied with the final product.

We walk through these steps, and provide additional details using three separate examples. In Section 8.3, we create a micromap showing the populations of large cities across Asia. Section 8.4 provides a more complex example where point locations are manually adjusted to limit overlap, and Section 8.5 provides an example where point locations are repelled from each other using the `point_repel()` function.

8.3 Plotting Large Cities in Asia

In this example, we create a relatively simple linked micromap plot that shows the locations of the 10 largest cities in Asia, and their associated populations.

To begin, the necessary R packages need to be loaded.

```
library(micromap)
library(sf)
library(dplyr)
source("R/Ch6_functions.R") # provides custom functions
```

8.3.1 Loading Point Location Data

Next we load the point location data, which is in the form of a data frame, that contains four columns that we will use: `city`, `population`, `lat`, and `long`.

```
cities <- readRDS("data/Ch6-data/asian_cities.RDS")

head(cities)

## # A tibble: 6 x 5
##   city      country population    lat   long
##   <chr>     <chr>        <dbl> <dbl> <dbl>
## 1 Shanghai  China       24256800  31.2 121.
## 2 Beijing   China       21516000  39.9 116.
## 3 Mumbai    India       20667656  19.1  72.9
## 4 Delhi     India       16787941  28.7  77.2
## 5 Karachi   Pakistan    14910352  24.9  67.0
## 6 Istanbul  Turkey     14657000  41.1   29
```

We convert the population column to be in units of millions of people, so that the numbers are more readable in the final micromap.

```
cities$population <- cities$population / 10^6
```

Before moving on to create the micromap it can be helpful to create a preliminary figure of the point locations. In examining the locations of the cities in this data set (Figure 8.1), we can see that none of the cities are very close together. Therefore, when we draw circles around the point locations, we won't need to be worried about handling overlap in the final figure.

```
plot(cities$long, cities$lat, xlab = "Longitude", ylab = "Latitude")
```

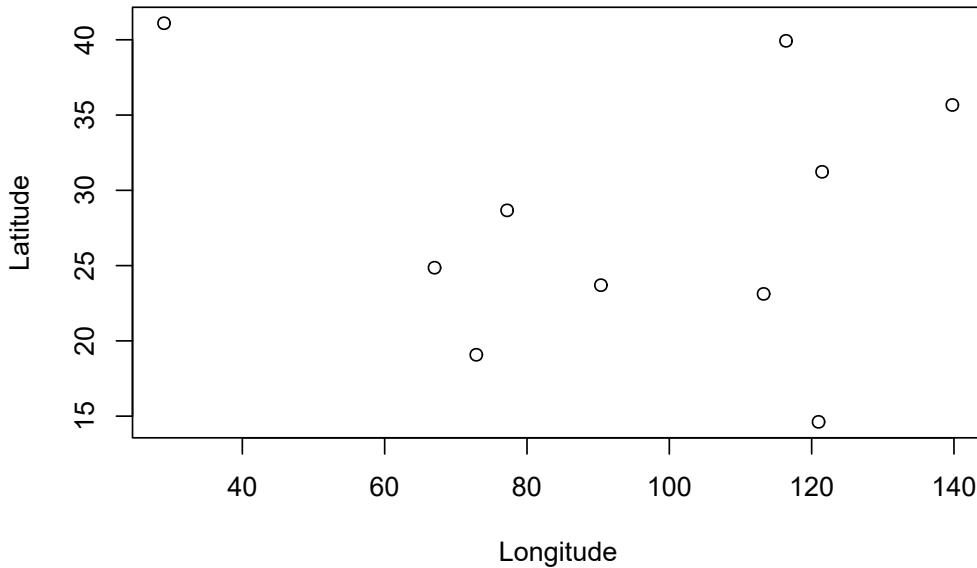


FIGURE 8.1 Latitude and longitude of 10 large cities in Asia. Note, in this example the point locations are far apart, and therefore the final map will not have overplotting issues.

8.3.2 Generating Polygons around Points

Our next task is to draw circles around the points. However, first the point location data needs to be properly formatted. We first convert the `cities` data frame to a simple features (`sf`) object from the `sf` package (Pebesma, 2018). Usually metadata or other descriptions of the data will provide the coordinate reference systems (“crs”). In this example we happen to know that the crs is EPSG:4326 and we can specify that using the `crs` argument. Correctly specifying the crs is important for Section 8.3.4 where we combine this object with a base map.

In this step we also use the `sf::st_transform()` function to transform the data to a projection that is more suitable for mapping locations in Asia. This transformation has the added benefit of making `sf::st_crop()` work smoothly in a later cropping step (the data are now projected onto a plane unlike the original un-projected latitude and longitude coordinates).

```
cities_sf <- st_as_sf(cities, coords = c("long", "lat"), crs = 4326) %>%
  st_transform(crs = "EPSG:8859")
names(cities_sf)

## [1] "city"      "country"    "population" "geometry"
```

The `cities_sf` object contains the same columns as the original data set, except that the `long` and `lat` columns have been replaced by a `geometry` column. The `geometry` column contains spatial data that defines the point locations of each city (row) in the data set. `sf` objects are special kinds of data frames, that can represent a wide variety of spatial data (in the `geometry` column) and also contain additional spatial metadata (see <https://r.geocompx.org/> for more details).

Next we create circles around the point locations, which we do with the `points2circles()` function. This function can be supplied a data frame with columns that give the `x` and `y` coordinates, or an `sf` object.

By default the radius of the circles is 5% of the range of the `x` variable (i.e., 5% of the width of the map). A different value can be passed to the `radius` argument to adjust the size of the circles. The resulting output of `points2circles()` is an object of class `sf`.

```
circles_sf <- points2circles(cities_sf,
  radius = 5
)
names(circles_sf)

## [1] "city"      "country"    "population" "geometry"
```

```
plot(circles_sf[1])
```

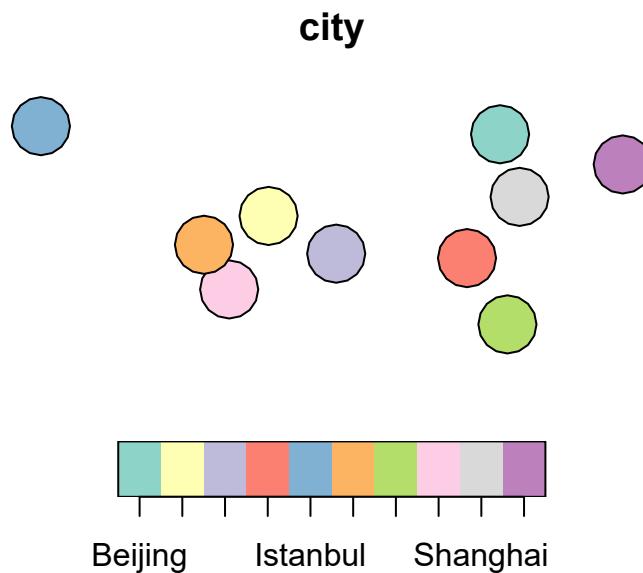


FIGURE 8.2 Latitude and longitude of 10 large cities in Asia, now plotted with circles around them.

The default method of the `micromap::mmplot()` function requires a data frame as input, which can be created using `micromap::create_map_table()` (see Section 2.2). These `sf` objects first need to be converted to a `Spatial` object, before it can be passed to `create_map_table()` (this is accomplished with `sf::as_Spatial()`).

```
circles_df <- create_map_table(
  tmp.map = as_Spatial(circles_sf),
  IDcolumn = "city"
)
```

8.3.3 Creating a Draft Linked Micromap Plot

Figure 8.3 shows a rudimentary micromap with the locations of the cities shown as circles.

```
mmpplot(
  stat.data = cities,
  map.data = circles_df,
  panel.types = c("labels", "dot", "map"),
  panel.data = list("city", "population", NA),
  map.link = c("city", "ID"),
  ord.by = "population",
  grouping = 5
)
```

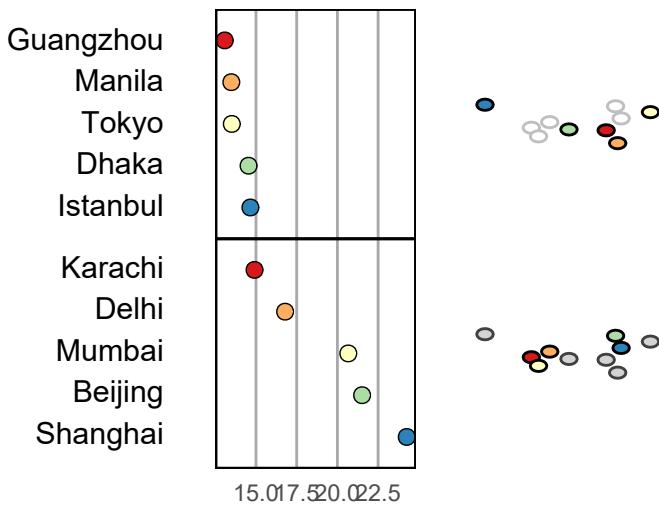


FIGURE 8.3 Map showing the 10 largest cities in Asia. Note that the locations of the cities are shown, but there are no country polygons to indicate the location of the cities.

8.3.4 Combining Point Locations with a Background Map

To make a better micromap, the circles showing the city locations need to be combined with a base map. Some data wrangling may be needed to acquire the necessary base map (see Section 4 for more details on preparing shapefiles for use in micromaps).

For this example we're using a data set of country polygons for the whole world. This data set can be downloaded using the **rnaturrearth** package (Massicotte & South, 2023).

```
wrld <- rnaturalearth::ne_download(
  scale = "small",
  type = "countries",
  returnclass = "sf",
  category = "cultural",
  load = TRUE
)
```

For convenience we provide a copy of this data set (containing a subset of the columns).

```
wrld <- readRDS("data/Ch6-data/world_countries.RDS")
```

We first transform the `wrld` object so that it has the same coordinate reference system as the `circles_sf` object.

```
wrld <- st_transform(wrld, crs = st_crs(circles_sf))
```

Next we use the `sf::st_crop()` function so that only the region we're interested in will be shown. We then crop the map based on the extent of the `circles_sf` object we previously created. Note that we are using the `sf::st_bbox()` function to get the bounding box of the `circles_sf` object, and then cropping to the extent of that bounding box.

```
asia <- st_crop(wrld, st_bbox(circles_sf))
```

Prior to combining the `asia` and `circles_sf` objects, a new column needs to be added to the `asia` object. When creating micromaps, the input includes an ‘ID’ column that contains names that uniquely identify the polygons for which data will be shown (see Section @ref(#Ch2-Example1)). Polygons that are shown on the map but for which no data is shown (i.e., which form the map in the background) have `NA` values in the ‘ID’ column. The ‘ID’ column in the `circles_sf` object is `city`, and so here we add an empty column of that name to `asia`, before binding the data sets together.

```
asia$city <- NA
comb_sf <- rbind(circles_sf["city"], asia["city"])
names(comb_sf)

## [1] "city"      "geometry"

plot(comb_sf[1])
```

The resulting `comb_sf` object contains two columns. Note that when selecting a specific column of an `sf` object using `object['columnName']`, an `sf` object is returned with the desired column as well as the `geometry` column. In other words, the `geometry` column is ‘sticky’ and comes along when selecting columns in this way, so that the associated spatial information is not lost.

Next we need convert the object into a regular data frame using `micromap::create_map_table()` so that it can be used in the `micromap::mmplot()` function.

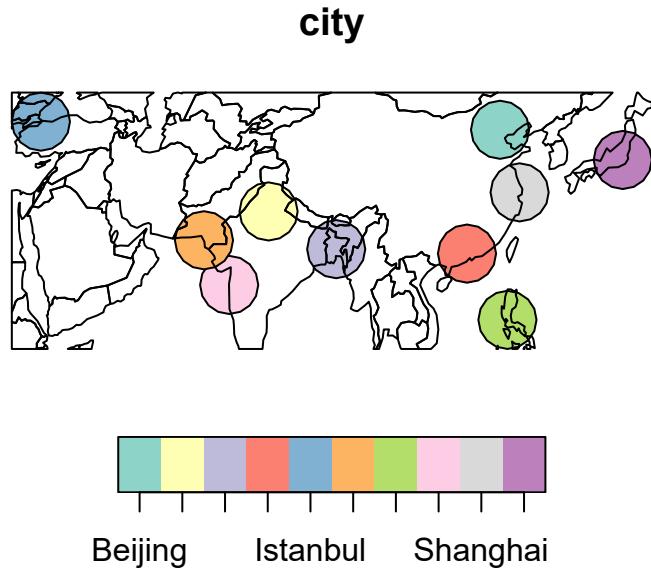


FIGURE 8.4 Map showing the 10 largest cities in Asia. Note that the locations of the cities are shown. In addition, country polygons are shown as well to indicate the location of the cities in Asia.

```
comb_df <- create_map_table(
  tmp.map = as_Spatial(comb_sf),
  IDcolumn = "city"
)
head(comb_df, 3)

##           ID region poly   coordsx coordsy hole plotorder plug
## 1 Shanghai     1      1 -2077374 3910574     0        1    0
## 2 Shanghai     1      1 -2112843 3732262     0        1    0
## 3 Shanghai     1      1 -2213848 3581096     0        1    0
```

Note that in the `comb_df` data frame the `ID` column is `NA` for all polygons associated with the base map, and is filled with the city name for all polygons (circles) that represent cities. Another draft linked micromap plot is shown in Figure 8.5.

```
mmpplot(
  stat.data = cities,
  map.data = comb_df,
  panel.types = c("labels", "dot", "map"),
  panel.data = list("city", "population", NA),
  map.link = c("city", "ID"),
  ord.by = "population",
  grouping = 5,
  panel.att = list(
    list(3,
      map.all = TRUE,
```

```

    nodata.border.color = "black"
)
)
)
)
```

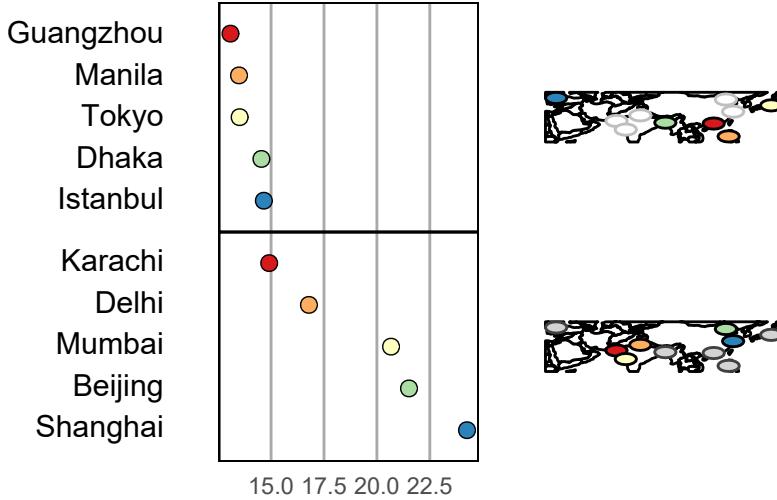


FIGURE 8.5 Draft linked micromap plot showing the 10 largest cities in Asia and their population. Note that the locations of the cities are shown. In addition, country polygons are shown as well to indicate the location of the cities in Asia.

8.3.5 Refined Linked Micromap Plot Showing the Base Map

Now we can create the micromap with the base map displayed. To properly display the base map the `map.all` argument for the map panel needs to be set to `TRUE` (in this case the maps are shown in the third panel). That way, polygons that have no data (i.e., polygons where the `ID` column is `NA`) also appear. Additionally, the `nodata.border.color` argument needs to be passed a color (here we use “black”), to define the color of the base map lines. The width of those lines is set with the `nodata.border.size` argument. The resulting linked micromap plot is shown in Figure 8.6.

The `micromap::mmpplot` documentation (`?mmpplot`), provides information on additional arguments.

```

library(labeling)

mmpplot(
  stat.data = cities,
  map.data = comb_df,
  panel.types = c("labels", "dot", "map"),
  panel.data = list("city", "population", NA),
  map.link = c("city", "ID"),
```

```
ord.by = "population",
rev.order = TRUE,
grouping = 5,
colors = RColorBrewer::brewer.pal(n = 5, name = "Reds")[5:1],
panel.att = list(
  list(
    1,
    header = "Cities",
    align = "left",
    panel.width = 0.5
  ),
  list(
    2,
    header = "Population",
    graph.bgcolor = "lightgray",
    panel.width = 0.75,
    point.size = 1.5,
    xaxis.ticks = as.list(labeling::extended(
      dmin = min(cities$population),
      dmax = max(cities$population),
      m = 6
    )),
    xaxis.labels = as.list(labeling::extended(
      dmin = min(cities$population),
      dmax = max(cities$population),
      m = 6
    )),
    xaxis.title = "[in Million]",
    xaxis.title.size = 1.25
  ),
  list(
    3,
    header = "Light Gray Means\nHighlighted Above",
    map.all = TRUE,
    active.border.color = "black",
    active.border.size = 0.3,
    inactive.border.color = "gray",
    inactive.border.size = 0.3,
    outer.hull = FALSE,
    nodata.border.color = "black",
    nodata.border.size = 0.5,
    panel.width = 1
  )
)
)
```

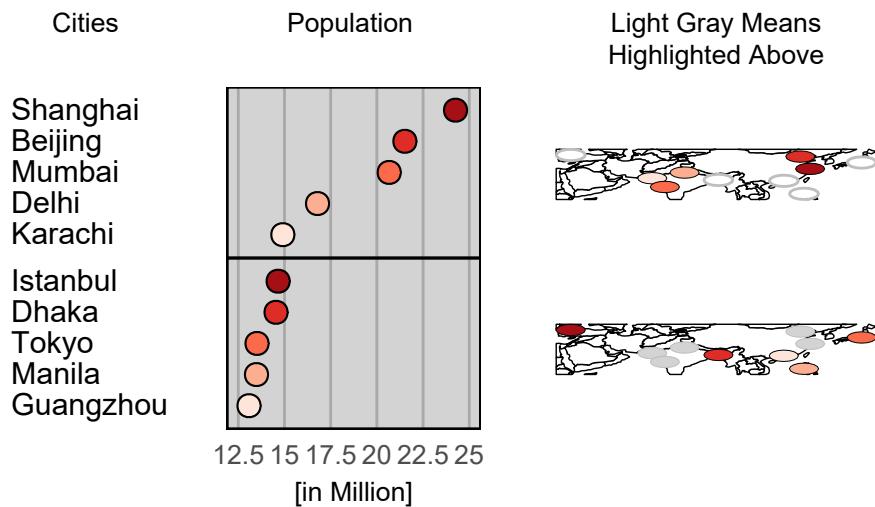


FIGURE 8.6 Refined linked micromap plot showing the 10 largest cities in Asia and their population.

8.4 County Seats in Utah

Here, we provide an example of a micromap that shows data for the 29 towns or cities in Utah, USA, that are the County seats of the 29 counties in the State of Utah. In this example, we also adjust the locations of some of the points to reduce overplotting in the final figure.

8.4.1 Loading Point Locations and Base Map Data

First, we load the point locations. This is an `sf` object, which provides the name of the county seat, as well as its elevation, population, spatial coordinates (in the `geometry` column), and the name of the county.

```
seats1 <- readRDS("data/Ch6-data/utah_counties.RDS")
class(seats1)

## [1] "sf"           "data.frame"

names(seats1)

## [1] "elevation"    "county_seat"   "county"        "pop_last_cen" "geometry"
```

Next, we want to load a spatial data set from the `tigris` package (Walker, 2024) that contains polygons of counties in Utah, which we can use as a base map.

```
county_polygon <- tigris::counties("Utah",
  cb = TRUE,
  year = 2020
)
```

Before moving on, we need to check whether the base map and point locations have the same coordinate reference system.

```
st_crs(county_polygon) == st_crs(seats1)
```

```
## [1] FALSE
```

Since they are different, one of the objects needs to be re-projected. Here, we are re-projecting the `seats1` object, so that it has the same coordinate reference systems as the `county_polygon` object.

```
seats2 <- st_transform(seats1, crs = st_crs(county_polygon))
```

8.4.2 Adjusting Point Locations

Figure 8.7 shows that some county seats are quite close to each other, especially in the north-central portion of the state. In many cases (such as the one shown in Section 8.3), no adjustment of point locations is needed because the points are naturally well spaced. However, in this case we may want adjust the locations of some of these points so that they are more easily visible in the final micromap. The user needs to consider how they want to balance being able to see all points in the micromap (i.e., so there is limited overlap), while not moving points so far as to create a misleading visualization. We think it is also important that figure captions disclose when point locations have been adjusted.

```
plot(county_polygon$geometry)
plot(seats2$geometry,
  add = TRUE,
  pch = 19,
  col = "blue"
)
```

There are multiple ways that point locations could be adjusted. A tedious, but effective, method could be to manually choose new coordinates. In this case, another viable option is to replace the coordinates of some county seats with the centroid of their respective county. That way, we can avoid excessive overplotting of the points on the micromap while still having the points appear in the correct county.

Here, there are several county seats have locations we may want to adjust. To do this, we first create a vector of names of counties who's county seats we want to move. Then we calculate the centroids of those counties using the `sf::st_centroid()` function.

```
county_names <- c(
  "Weber", "Morgan", "Davis", "Salt Lake",
```

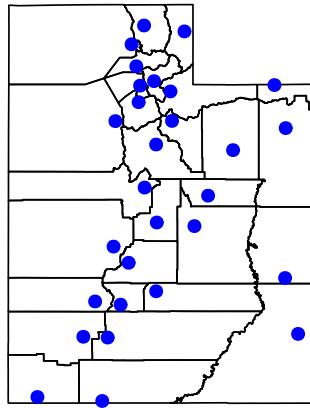


FIGURE 8.7 Locations of counties and county seats (points), in Utah.

```
"Summit", "Wasatch", "Box Elder"
)

centroids <- county_polygon %>%
  filter(NAME %in% county_names) %>%
  st_centroid()
```

Next we replace the coordinates of the county seats with the centroids of the county they are located in. Note here we are first arranging the columns that contain county names into alphabetical order in both the `seats2` and `centroids` objects so that the row ordering matches and correct coordinates are replaced.

```
seats_adjust1 <- arrange(seats2, county)

centroids <- arrange(centroids, NAME)

seats_adjust1$geometry[seats_adjust1$county %in% county_names] <-
  centroids$geometry
```

Figure 8.8 shows the effect of adjusting these points.

```

plot(county_polygon$geometry)
plot(seats2$geometry[seats2$county %in% county_names],
  add = TRUE,
  pch = 19,
  col = "blue"
)
plot(seats_adjust1$geometry[seats_adjust1$county %in% county_names],
  add = TRUE,
  pch = 19,
  col = "red"
)

```

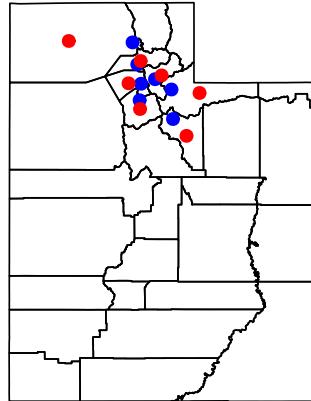


FIGURE 8.8 Original locations of county seats in blue, and adjusted locations (county centroids) in red.

We leave it as an exercise to the reader to adjust the locations of other county seats to further improve the final map.

8.4.3 Preparing Points and Base Map

Now that we have adjusted the point locations, we need to create circles around the points, that will be displayed on the micromap.

```

seats_circles <- points2circles(seats_adjust1, radius = 6)
plot(seats_circles$geometry)

```

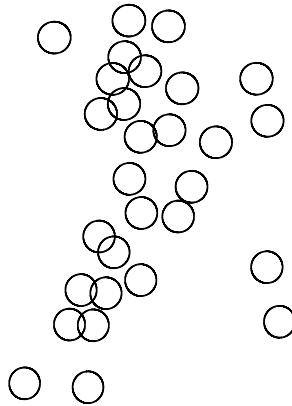


FIGURE 8.9 Circles drawn around the locations of county seats. Note that overlap between individual circles is fairly limited, because some locations have been adjusted.

Next, the `seats_circles` needs to be combined with `county_polygons`.

To be able to bind the two objects, the `county_polygon` object needs to have an empty `county_seats` column (as described in Section @ref(#Ch6-Example1-Combining))

```
county_polygon$county_seat <- NA
seat_map <- rbind(
  county_polygon[["county_seat"]],
  seats_circles[["county_seat"]]
)
```

The `seat_map` object is a `sf` object that needs to be converted to a `Spatial` object before it can be flattened into a regular data frame using the `micromap::create_map_table` function.

```
seat_map_df <- create_map_table(
  tmp.map = as(seat_map, "Spatial"),
  IDcolumn = "county_seat"
)
```

8.4.4 Creating a Linked Micromap Plot

We now have the necessary data to create the micromap using the `mmplot` function.

Note that the `stat.data` argument in the `micromap::mmpplot` function must be passed a data frame, and the class of `seats1` is `sf`. Therefore, we use the `sf::st_drop_geometry` function to drop the geometry column (i.e., which contains the spatial data) and convert the object to a regular data frame.

```
seats_df <- st_drop_geometry(seats1)
```

Note that, as described in Section @ref(#Ch6-Example1Micromap2), the `map.all`, `no-data.border.color`, and `nodata.border.size` arguments are specified for the third panel so that the polygons that make up the base map are plotted.

```
mmpplot(
  stat.data = seats_df,
  map.data = seat_map_df,
  panel.types = c("map", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, NA, "county_seat", "elevation", "pop_last_cen"),
  map.link = c("county_seat", "ID"),
  ord.by = "elevation",
  grouping = c(5, 5, 4, 4, 5, 5),
  median.row = TRUE,
  colors = RColorBrewer::brewer.pal(n = 5, name = "YlGnBu")[5:1],
  vertical.align = "center",
  panel.att = list(
    list(
      1,
      header = "Light Gray Means\nHighlighted Above",
      map.all = TRUE,
      active.border.color = "black",
      active.border.size = 0.3,
      inactive.border.color = "gray",
      inactive.border.size = 0.3,
      panel.width = 1,
      outer.hull = FALSE,
      nodata.border.color = "black",
      nodata.border.size = 0.5,
      outer.hull.size = 0.5
    ),
    list(
      2,
      point.type = 20,
      point.border = TRUE,
      point.size = 2,
      panel.width = 1.1
    ),
    list(
      3,
      header = "County Seat",
      align = "left",
      panel.width = 0.8
    )
  ),
  no-data.border.color = "black",
  nodata.border.size = 0.5
)
```

```
list(
  4,
  header = "Elevation",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  xaxis.ticks = list(3000, 5000, 7000),
  xaxis.labels = list(3000, 5000, 7000),
  xaxis.title = "feet",
  panel.width = 1.2
),
list(
  5,
  header = "Population",
  graph.bgcolor = "lightgray",
  point.size = 1.5,
  right.margin = 0.25,
  xaxis.ticks = list(0, 100000, 200000),
  xaxis.labels = list(0, 100000, 200000),
  xaxis.title = "Count (last Census)",
  panel.width = 1.2
)
)
```

8.5 Premier League Football Stadiums in England and Wales

For this example, we will be using locations of premier league football stadiums in England and Wales. A challenge with this data set is being able to properly visualize multiple locations that are very close together, such as having several stadiums located in London, Liverpool, and Manchester. This is likely to be a common issue that a user will run into, where locations are clustered in, for example, urban areas.

Here, we show a micromap with the actual stadium locations plots, and then create a second version of the micromap where we use a more automated approach to repel locations of stadiums that are too close together.

8.5.1 Obtaining Point Locations

The data set contains the name of the stadium, the team that plays there, the capacity of the stadium, number of total spectators over the 2018 season, the average number of spectators at a match, and the coordinates (**geometry** column).

```
stadiums <- readRDS("data/Ch6-data/stadiums.RDS")
names(stadiums)

## [1] "stadium"      "team"        "capacity"     "spectators"   "average"
```

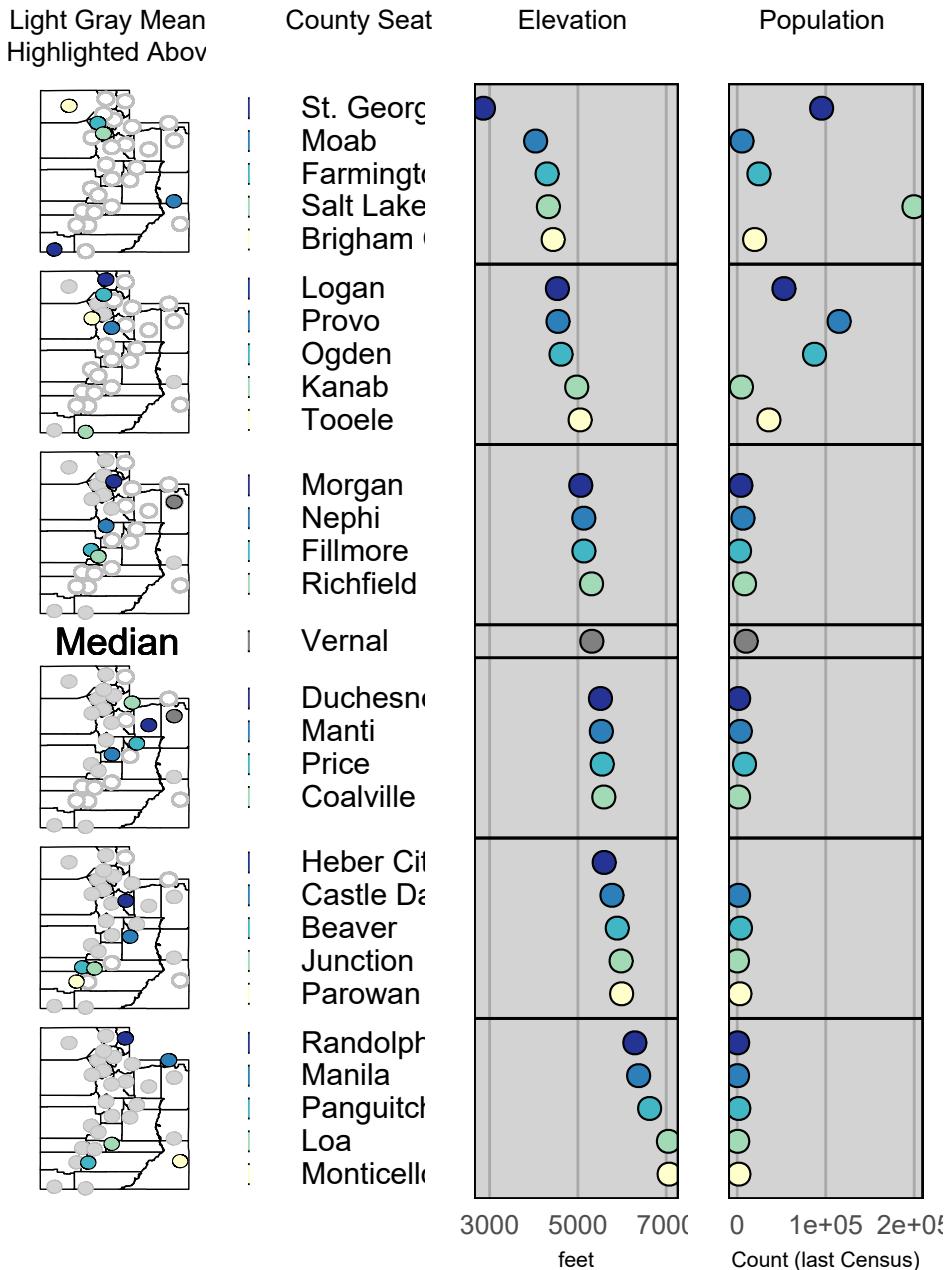


FIGURE 8.10 Micromap showing the elevations and population at the last Census of county seats in Utah. Some locations have been adjusted to decrease overplotting.

```
## [6] "geometry"
```

Next we “draw” circles around the points.

```
stadium_circles <- points2circles(stadiums, radius = 8)
```

8.5.2 Acquiring the Underlying Map

We can extract a base map of the UK, and then crop it to the extent we are interested (here we are only keeping the `geometry` information because we’re not interested in any of the associated country information).

```
uk_map <- wrld[wrld$NAME == "United Kingdom", ]
uk_map <- st_transform(uk_map, st_crs(stadium_circles))

uk_map <- st_crop(
  uk_map["geometry"],
  c(xmin = -5, ymin = 50, xmax = 2, ymax = 56)
)
```

Now add the empty “ID” column.

```
uk_map$stadium <- NA
```

8.5.3 Combining with Points

As before, we need to check whether the base map for the UK and point locations for the stadiums have the same coordinate reference system. Here, they are the same as we previously matched the coordinate reference systems to be able to better crop the UK to a meaningful area that omits most of Scotland and Northern Ireland and only keeps England and Wales.

```
st_crs(uk_map) == st_crs(stadium_circles)

## [1] TRUE

# uk_map <- st_transform(uk_map, st_crs(stadium_circles))
```

Next, we bind the base map and the point locations together.

```
stadiums_map1 <- rbind(uk_map["stadium"], stadium_circles["stadium"])

stadiums_map_df <- create_map_table(
  tmp.map = as(stadiums_map1, "Spatial"),
  IDcolumn = "stadium"
)
```

8.5.4 Creating a Draft Linked Micromap Plot

Here, we create a draft linked micromap map showing stadium locations that are not adjusted (i.e., they are still overlapping).

```
mmpplot(
  stat.data = st_drop_geometry(stadiums),
  map.data = stadiums_map_df,
  panel.types = c("labels", "dot", "map"),
  panel.data = list("stadium", "capacity", NA),
  map.link = c("stadium", "ID"),
  ord.by = "capacity",
  grouping = 5,
  panel.att = list(
    list(
      1,
      panel.width = 1.8
    ),
    list(
      3,
      map.all = TRUE,
      active.border.color = "black",
      active.border.size = 0.3,
      inactive.border.color = "gray",
      inactive.border.size = 0.3,
      panel.width = 1,
      outer.hull = FALSE, # shows outside of map
      # this is required to see all map borders:
      nodata.border.color = "black",
      nodata.border.size = 0.5,
      outer.hull.size = 0.5
    )
  )
)
```

8.5.5 Refined Linked Micromap Plot with Adjusted Stadium Locations

The `point_repel()` function can be used to repel points that are too close together, so that in this case, the clusters of stadiums in London, Liverpool, and Manchester no longer will overplot each other. Here, some of the default arguments in `point_repel()` were changed (`rep.fact` and `adj.max`) to increase the distance between points. The default arguments may need to be adjustment to obtain satisfactory repulsions between points (see the function documentation for details). Note that points that are already far apart are not moved.

```
stadiums_repel <- point_repel(stadiums, rep.fact = 50, adj.max = 5, rep.dist.lmt = 5, attr.fact = 0.01)
par(mfrow = c(1, 2))
plot(stadiums$geometry,
  main = "Original Point Locations"
)
plot(stadiums_repel$geometry,
```

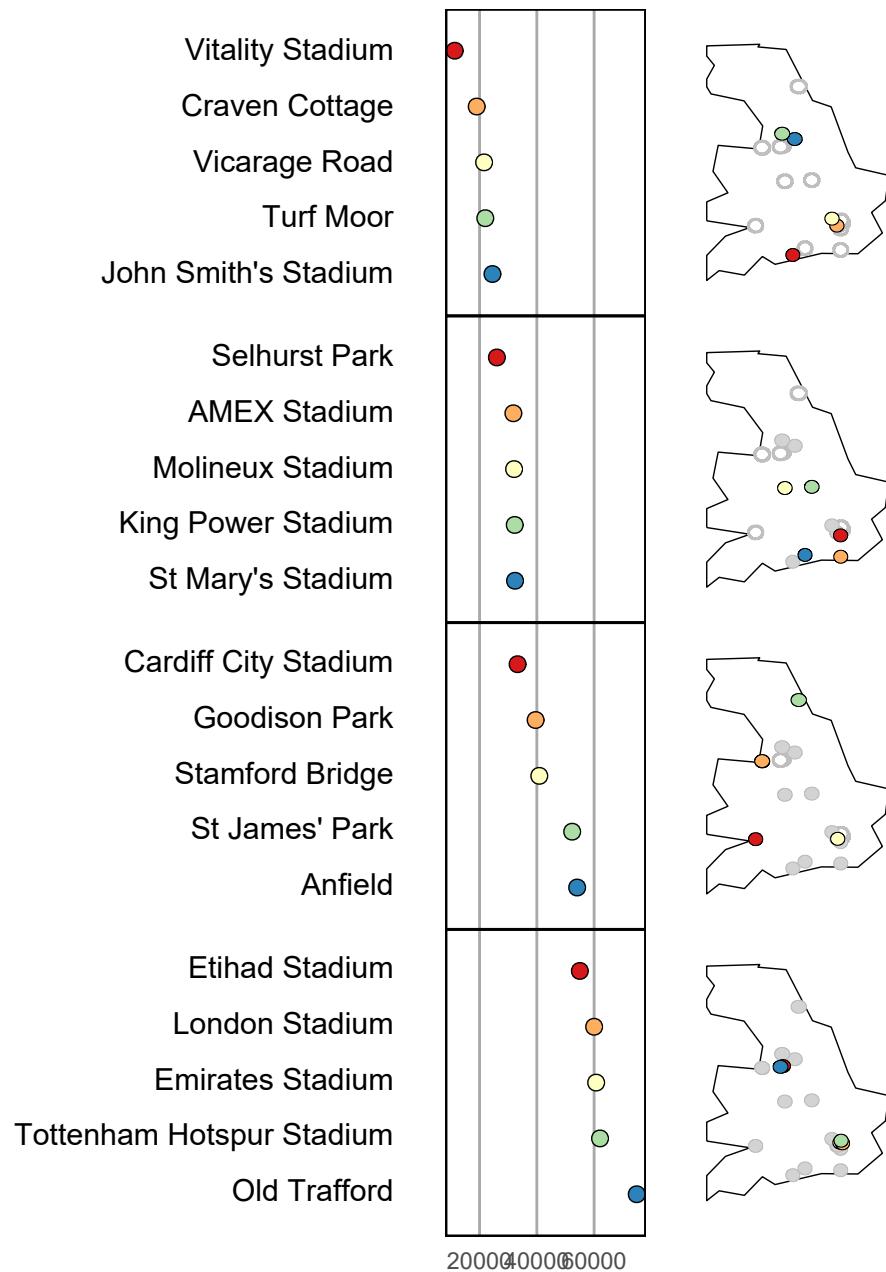
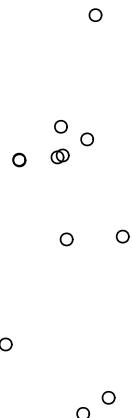
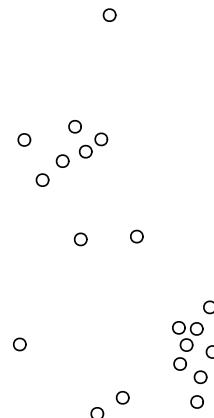


FIGURE 8.11 Micromap showing stadium locations. Note that these maps are not very readable due to the overlap in locations

```
    main = "Repelled Point Locations"
)
```

Original Point Locations**Repelled Point Locations**

Now circles need to be drawn around the points.

```
stadium_circles_repel <- points2circles(stadiums_repel, radius = 8)

stadiums_map_repel <- rbind(
  uk_map["stadium"],
  stadium_circles_repel["stadium"]
)

stadiums_map_repel_df <- create_map_table(
  tmp.map = as(stadiums_map_repel, "Spatial"),
  IDcolumn = "stadium"
)
```

```
mmpplot(
  stat.data = st_drop_geometry(stadiums),
  map.data = stadiums_map_repel_df,
  panel.types = c("map", "labels", "dot_legend", "labels", "dot", "dot"),
  panel.data = list(NA, "stadium", NA, "team", "capacity", "average"),
  map.link = c("stadium", "ID"),
  ord.by = "capacity",
  rev.order = TRUE,
```

```
grouping = 5,
colors = RColorBrewer::brewer.pal(n = 5, name = "RdYlBu")[5:1],
panel.att = list(
  list(
    1,
    header = "Light Gray Means\nHighlighted Above",
    map.all = TRUE,
    active.border.color = "black",
    active.border.size = 0.3,
    inactive.border.color = "gray",
    inactive.border.size = 0.3,
    panel.width = 1.7,
    outer.hull = FALSE,
    nodata.border.color = "black",
    nodata.border.size = 0.5,
    outer.hull.size = 0.5
  ),
  list(
    2,
    header = "Stadium",
    align = "right",
    text.size = 0.7,
    left.margin = -0.7,
    right.margin = -0.2,
    panel.width = 1.65
  ),
  list(
    3,
    point.type = 20,
    point.border = TRUE,
    point.size = 1.8,
    panel.width = 1.8
  ),
  list(
    4,
    header = "Team",
    align = "left",
    text.size = 0.7,
    panel.width = 1.65
  ),
  list(
    5,
    header = "Stadium\nCapacity",
    graph.bgcolor = "lightgray",
    point.size = 1.3,
    xaxis.ticks = list(0, 40000, 80000),
    xaxis.labels = list(0, 40000, 80000),
    xaxis.labels.size = 0.5,
    left.margin = -0.5,
```

```

    right.margin = 0.25,
    panel.width = 1.4
),
list(
  6,
  header = "Average\nAttendance",
  graph.bgcolor = "lightgray",
  point.size = 1.3,
  xaxis.ticks = list(0, 40000, 80000),
  xaxis.labels = list(0, 40000, 80000),
  xaxis.labels.size = 0.5,
  left.margin = -0.5,
  right.margin = 0.5,
  panel.width = 1.4
)
)
)

```

8.6 Summary and Further Reading

In this chapter, we have outlined the essential steps to create linked micromap plots for point locations. The point location data requires some pre-processing before the **micromap** R package can be used to create the figure.

The process begins with obtaining coordinates and data of interest for the point locations. Next, geographic data for the underlying base map is obtained and prepared. A preliminary plot of the point locations is created, and if necessary adjustments are made to the point locations to limit overplotting. Polygons (circles or other shape) are then generated to represent the point locations. These polygons are integrated with the base map. The combined object is then converted into a data frame suitable for use with the `micromap::mmpplot()` function. A draft simple linked micromap plot is created, and the visualization is refined iteratively, adjusting the size and overlap of polygons, adding labels, and enhancing other visualization elements.

To become proficient in creating linked micromaps for point locations we recommend you also read Chapter 1 for a better foundational understanding of the rationale behind micromaps. Additionally, see Chapter 2 for a more thorough coverage of the **micromap** R package and its functionality. Note also that there are many potential applications for micromaps of point locations that we did not explore here, such as environmental and medical data collected at fixed sites (see Chapter 13 for more details on environmental data and Chapter 14 for more details on medical data).

Previously, linked micromap plots for point locations have been used for time series plots of climate data from weather station locations, accessible through the Utah Climate Center web page at Utah State University (Thapliyal, 2009; Yarra, 2010). The use of linked micromap plots for point locations also has been demonstrated for positions on a baseball field by Carr and Pickle (2010), Figure 1.5. Lastly, the types of micromaps we presented in this

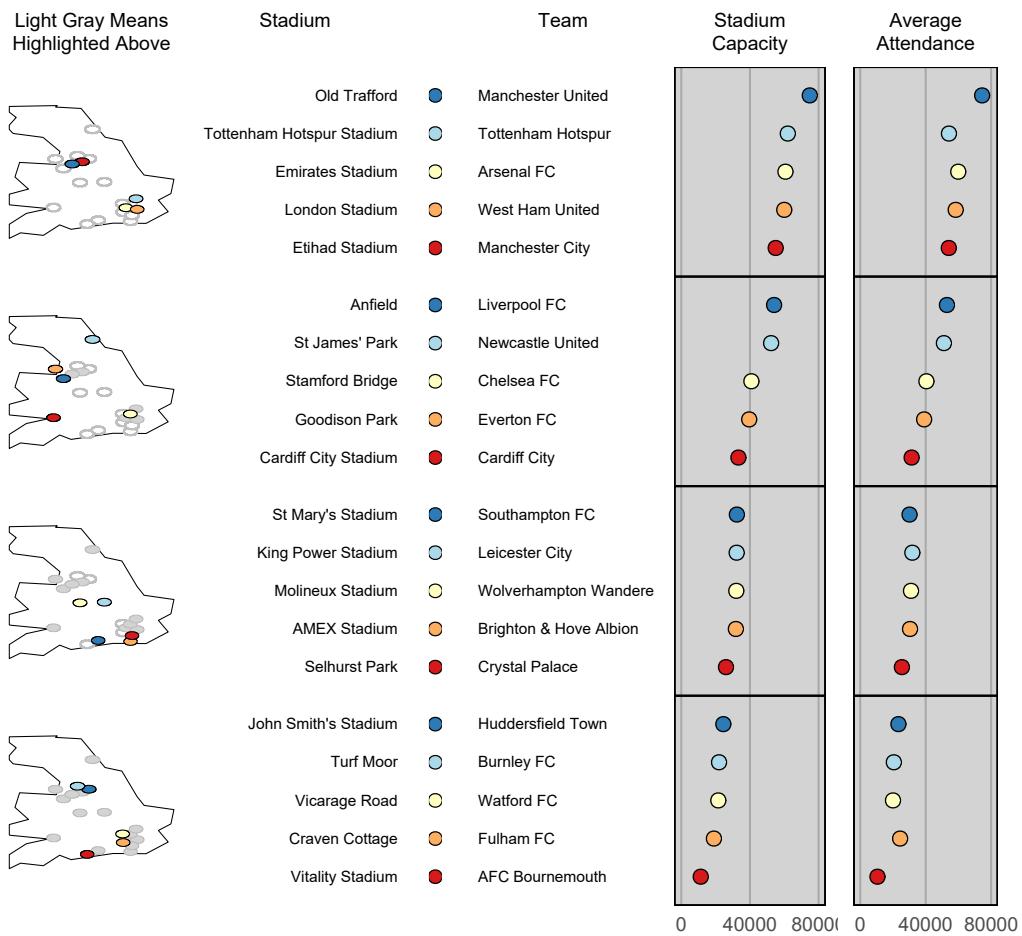


FIGURE 8.12 Micromap showing stadium locations. Here the stadium locations have been moved slightly so that complete overlap of points does not occur. Micromaps can contain two label columns, as seen here where both the stadium name and team name are provided.

chapter could also be presented in different and creative ways such as via web applications (see Chapter 9).

References

- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Massicotte, P., & South, A. (2023). *rnaturrearth: World Map Data from Natural Earth* [R package version 1.0.1 (<https://CRAN.R-project.org/package=rnaturrearth>)].

- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data [<https://doi.org/10.32614/RJ-2018-009>]. *The R Journal*, 10(1), 439–446.
- Thapliyal, A. (2009). Enhancement of Web-based Interactive Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Walker, K. (2024). *tigris: Load Census TIGER/Line Shapefiles* [R package version 2.1, (<https://CRAN.R-project.org/package=tigris>)].
- Yarra, P. K. (2010). Refactoring Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].

9

Web-based Linked Micromap Plots Using Shiny

MARCUS W. BECK

Web-based interactive linked micromaps were first developed in the late 1990s. Early examples included visualizing hazardous air pollutant data using the Graphics Production Library (Symanzik, Axelrad, et al., 1999) and use of micromaps by the National Cancer Institute to visualize long-term cancer data (Bell et al., 2006). These early examples primarily used Java-based web technologies to combine micromap visualization tools with online functionality. Since then, the development of R packages to create web-based applications has allowed new opportunities to visualize and share micromaps online. This chapter will introduce the reader to the creation of linked micromap plots via the R **shiny** package (Chang et al., 2021). Core concepts of creating shiny applications will be discussed, including an example that demonstrates a minimal working shiny application with the **micromap** package (Beck, 2022; Payton & Olsen, 2015).

9.1 The Need for Web Applications

The ability to develop and host content online has obvious benefits for research and data-intensive applications. Reaching new audiences and creating a more immersive experience with interactive functionality can improve understanding beyond text descriptions in technical reports or research articles. This is especially important when research products have intended applications for non-technical audiences or to simply facilitate sharing of ideas among colleagues, such as for applied environmental (e.g., Chapter 13) or medical (e.g., Chapter 14) fields. Web-based applications can greatly improve technology transfer from the research community to practitioners, collaborators, or other stakeholders that may not need to fully understand the technical details of an underlying application, but require robust tools built on quality science to make informed decisions.

In this context, a web-based application (web app or dashboard, hereafter) can have two fundamental roles, where the primary goal of each is to increase understanding of concepts or results from research products or data. The first and most common use is creating a web app to support an existing or otherwise finalized research product. Simple examples include developing an application to share results as supplemental information from a research article, creating a web page to synthesize and distill large, publicly available datasets, or using a web application for prioritizing how decisions are made with limited resources. In the latter case, these may also be generically called decision support tools. The second, more under-utilized role of web apps is as intermediate tools that can enhance the research or exploratory analysis process. For many of the reasons that web apps can improve delivery of science to decision-makers or to inform the public, web apps can also benefit the research

community by improving collaboration among colleagues. We encourage readers to appreciate and understand how web apps can have benefits at different stages of analysis, not just as a final product.

The concepts and tools presented in this chapter can apply equally well to the two use cases above. In addition to understanding the theory of building web apps with micromaps, additional considerations include how the application is hosted online for access by others and what steps are made to ensure the application is maintained in the future. A detailed discussion of these concepts is beyond the scope of this chapter, but examples and resources are provided to allow the reader to understand their importance and how to further build on the concepts herein. We primarily cover the basics of developing a shiny application, but refer the reader specifically to Wickham (2021) and Sievert (2020) for a more comprehensive introduction to using shiny.

9.2 Existing Examples of Micromap Web Applications

The first web apps that used micromaps to explore spatial patterns in data were created in the late 1990s. Although R has been in use since this time, the ability to create web apps with shiny was not available until 2012 (the first version on CRAN was uploaded in December 2012). These early applications were similar to shiny in that they utilized functional programming and text markup languages to allow exploration of data within a web browser. Shiny uses similar tools within an R framework, i.e., it uses functions to create HTML and Java-based graphics for the developer, whereas early micromap apps were coded directly with these core tools. This required detailed knowledge of fundamental concepts of web programming that most R users currently do not have. Thus, there are few early examples of micromap web apps.

The first documented web app is described by Symanzik, Axelrad, et al. (1999) that used the Graphics Production Library (GPL, Carr et al., 1996) to visualize hazardous air pollutant data with micromaps in a web browser. The GPL was developed by the Bureau of Labor Statistics as a standalone software package based on Java to provide metadata access to underlying datasets through clickable metadata icons. Functionally, the GPL is similar to shiny by providing interactive components that allowed a user to access information of interest. Carr et al. (1996) state that the GPL addressed a critical need of creating quality graphics in a web-based format, where the latter was especially challenging at a time when access to robust software for creating online content was limited as the internet was gaining popularity as a tool to deliver data resources. The use of GPL with linked micromaps by Symanzik, Axelrad, et al. (1999) served a critical need by allowing a user to quickly filter a national-level dataset to regions of interest. Users were able to select a state, then advance to the underlying micromap display and tabular results that summarized air pollutant data by census tracts at the county level. As such, the application was inherently user-driven where the functionality was available to prioritize areas of concern based on location. More contemporary web apps built in shiny often serve similar purposes by allowing a user to determine which subsets of data to view from a larger database.

An additional example is provided by the National Cancer Institute that displayed national-scale state cancer profiles with linked micromaps (Bell et al., 2006; Pickle et al., 2015). Similar to Symanzik, Axelrad, et al. (1999), micromaps were displayed with an interactive interface developed using Java to allow a user to select regions and appropriate summary

statistics. However, the cancer profiles were the first example of using the **micromapST** R package in a web format, which predated the development of shiny by about a decade, demonstrating an early example of the utility of R as a platform to facilitate the creation of web content. These early applications also focused on the importance of communicating public health data in an accurate, clear, and concise manner for consumption by an increasingly interested user base (i.e., policymakers and the public) that was becoming more comfortable using the internet as a valuable tool to support decision-making. Emphasis was placed on tailoring graphics towards the needs of the audience, as well as ensuring confidentiality and displaying uncertainty in results that was true to the underlying data. This use case was especially relevant for linked micromaps that provided a useful format to address these needs. Notable limitations of the web apps were stated as difficulties in graphics display due to sizing issues and an inability to access or modify the underlying data (Pickle et al., 2015). Many of these issues have been addressed in the implementation of shiny and more recent supporting R packages, although micromaps will fundamentally be limited by the spatial units and their groupings chosen by the user.

A common theme of these early web-based tools was the need to deliver relevant summaries of large datasets in a simple and easily navigable format. The use of micromaps provided useful summary statistics in a spatial context, whereas the web-based platforms further advanced the utility of these data by providing user options to select locations or results of interest. However, there are not many examples of these early tools given the required expertise to create them with web programming platforms, such as Java. Subject-matter experts in the fields of public health or the environmental sciences are not conventionally trained to use these tools and the advent of modern tools like shiny has provided additional opportunities to improve how end users engage with data that can support critical decision-making. Moreover, these early tools are not accessible online in their original format, suggesting a need for developers to consider longevity, app maintenance, and permanence as a fundamental objective of app development. The remainder of this chapter describes these modern tools and additional considerations for hosting them online to ensure they are maintained in the future.

9.3 Shiny as an Open Source Tool for Web Applications

9.3.1 What is Shiny?

The R **shiny** package (Chang et al., 2021; Wickham, 2021) allows a developer to create web apps entirely in R. Shiny can expose existing R scripts to a web browser to allow anybody to access underlying features outside of R and the local environment of a personal computer. Shiny is commonly used to 1) communicate complex workflows to a non-technical audience with informative visualizations and interactive components, 2) share analysis output easily with colleagues without having to walk them through details of a script, and 3) help inform understanding of an analysis by creating a user interface to quickly evaluate data. Because linked micromaps can be easily created in R using the **micromap** or **micromapST** packages, creating web apps with shiny is a simple extension that can greatly improve understanding of data.

There are many advantages to using shiny over other platforms for creating web apps. The primary advantage is the ability to create rich web content entirely using R. There is no need to have a detailed understanding of web programming, such as HTML, CSS,

or JavaScript. However, shiny leverages this broader suite of web programming tools so that they are available for use should a developer have the need to expand an application’s utilities beyond the core features within shiny. More simply, shiny can be used as a web interface for any R workflow. This means that any custom analysis or graphic created by an R user can be fully integrated into a web app, unlike other platforms that may have rigid templates where functionality is sacrificed for ease of use.

Understanding shiny can be challenging at first because it introduces a new way of thinking about code. “Simple” R scripts are run linearly, being read from top to bottom in a conventional analysis workflow. The script is written, the code is sourced to the R console, and results or objects are saved in the environment of the current R session after running the script. A shiny app runs from an R script, but instead of executing code linearly, it uses **reactive** programming that detects when an input is changed on the application, runs the minimal amount of code that uses that input, then updates the output as needed. So, rather than running linearly, the script has interconnected components that share pieces of information that are executed on the fly to produce the results.

Reactivity can be daunting at first because it requires the developer to think about which pieces of code require inputs from other pieces and how that information is used to create output. This is not fundamentally different from writing functions in R and creating a shiny application should be straightforward if a developer is comfortable with functional programming (Wickham, 2019). Reactivity can be conceptualized by the building blocks of a shiny app. Every shiny app has the following:

- **User interface (UI):** Includes all inputs and outputs, as well as the appearance of the dashboard. In this context, “output” means the final product (e.g., plots, tables, etc.) that are placed in the user interface, but created by processing inputs sent to the server. In web-speak, this is the front end.
- **Server:** The guts or engine of how the inputs are used to create the outputs. This is where the working parts of the analysis are contained. The server can be as simple or as complicated as needed for an application. In web-speak, this is the back end.

The `ui` and `server` components can be contained in an R script, as follows:

```
library(shiny)

ui <- fluidPage()
server <- function(input, output) {}
shinyApp(ui = ui, server = server)
```

In RStudio, the application can be run clicking the “Run App” button at the top right of the source window or sending the code to the R console (Figure 9.1).

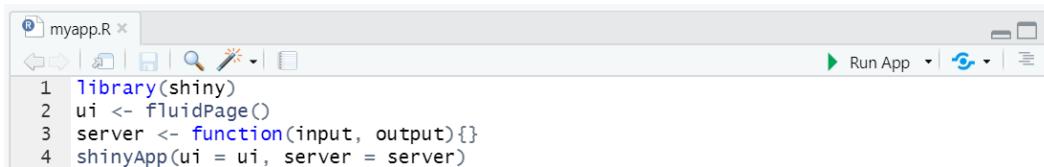


FIGURE 9.1 A minimal working shiny application in RStudio showing code for the user interface and server. The application can be run in RStudio by selecting “Run App”.

The above application will open locally, either through a local port opened by RStudio or

in a web browser depending on the user options in `shinyApp()`. In this example, an empty screen appears because the application currently does nothing and it is only accessible to the user in the current R session. Building a more functional shiny application only requires adding code to the server and user interface. Making the application accessible to others is described in section 9.3.3.

The example above can be expanded to demonstrate how code within the server and user interface might look in practice. The application below takes random samples from a normal distribution and creates a histogram showing the density using base R (Figure 9.2). The user interface determines how many random samples are used to create the plot. To make a shiny app, the inputs and outputs must be identified by the developer to determine where they are placed in the code. The input is what a user can modify (the sample size) and the output is the plot. Inputs and outputs go in the `ui` object. The `server` takes the inputs, produces the content for the output, and then sends the results back to the `ui`. Placing these components into the template is as follows:

```
library(shiny)

ui <- fluidPage(
  numericInput(inputId = "n", label = "Sample size", value = 50),
  plotOutput(outputId = "myplot")
)

server <- function(input, output) {
  output$myplot <- renderPlot({
    dat <- rnorm(n = input$n)
    hist(x = dat)
  })
}

shinyApp(ui = ui, server = server)
```

Evaluating what happens in the application each time a different sample size is selected by a user and how the application functions with the user interface and server components of the code is critical to understanding reactivity. The running application follows these steps (Figure 9.3) when the web app is initialized in the browser and when a user selects a new sample size:

1. The `input` value `n` selected by the user from the `ui` is sent to the `server`, seen as `input$n`.
2. The `dat` object is created as a random sample with size `n` and then a histogram is created as reactive output with `renderPlot`.
3. The plot output named `myplot` (chosen by the developer) is appended to the `output` list of objects in the `server`.
4. The plot is then rendered on the `ui` using `plotOutput` by referencing the `myplot` name from the `output` object.

From this simple workflow, some generalized rules and concepts about shiny reactivity can be described that apply to most shiny applications.

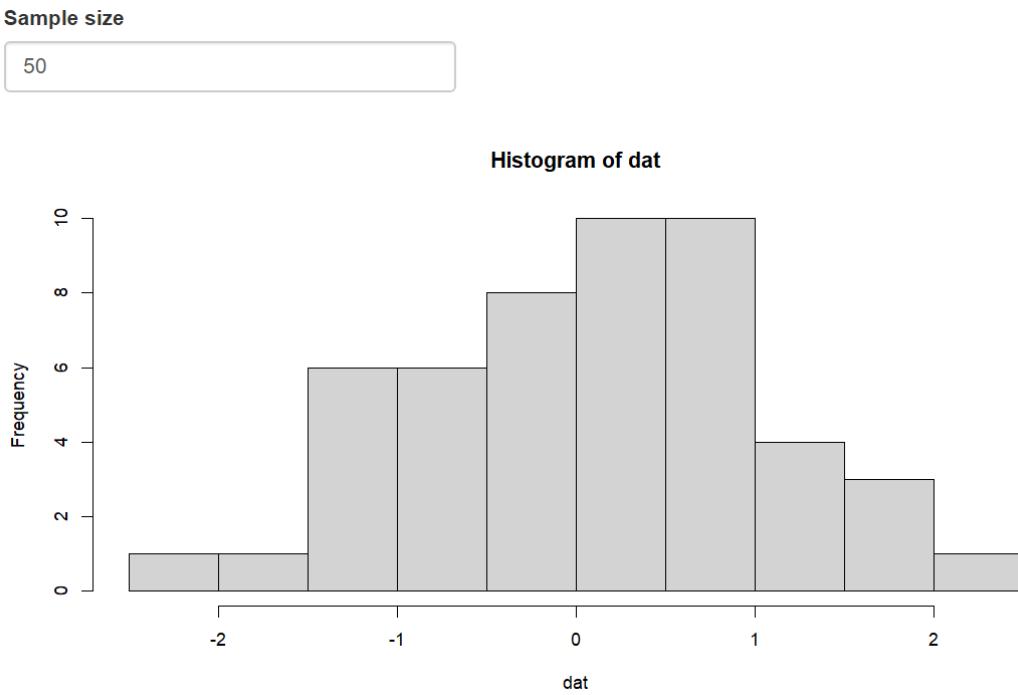


FIGURE 9.2 A minimal working shiny application with additional components in the server and user interface that produces a histogram with options to select the sample size.

```
library(shiny)

ui <- fluidPage(
  numericInput(inputId = "n", label = "Sample size", value = 50),
  plotOutput(outputId = "myplot")
)

server <- function(input, output){
  output$myplot <- renderPlot({
    dat<-3norm(n = input$n)
    hist(x = dat)
  })
}

shinyApp(ui = ui, server = server)
```

FIGURE 9.3 A representation of reactivity each time the user input is changed to select a different sample size for the histogram.

- All input objects are defined in the `ui` object, given a name inside the `input` function and then referenced in the `server` file by `input$name` (`input$n` in the example).

```
numericInput(inputId = "n", label = "Sample size", value = 50)
```

- All output objects to use in the `ui` object are created in the `server` object by assigning a “rendered” object to the `output` object by `output$name` (`output$myplot` in this case).

```
output$myplot <- renderPlot({
  dat <- rnorm(n = input$n)
  hist(x = dat)
})
```

- The `ui` object controls where and when the output is rendered, typically using a function named `fooOutput()` (`foo` meaning generic, e.g., `plot`, `table`, etc.) that has a complementary reactive function named `renderFoo()` in the `server` file.

```
plotOutput(outputId = "myplot")
```

- The `ui` object can be created with a function (`fluidPage()` here as one type of layout) with at least two inputs (one input, one output) separated by commas.
- The `server` object can be created with the `server()` function, where the input from the `ui` and output to send back to the `ui` is evaluated as an expression inside the curly braces `{}`.

All shiny applications use these concepts to create rich, interactive content that is accessible in a web browser. For example, the user interface can include different types of input selections, or “widgets”, depending on how a developer intends a user to change inputs or interact with the data. These widgets can be as simple as entering numeric input, i.e., `numericInput()` as from above, or can be more involved to select items from a predetermined list using a dropdown menu or radio buttons, using a slider to select values, choosing date ranges, uploading files, or even entering custom text. A full description of the available input options included with the shiny package can be found at <https://shiny.rstudio.com/gallery/widget-gallery.html>. The `shinyWidgets` package (Perrier et al., 2022) also includes additional widgets beyond those provided with shiny.

For shiny output, plots and tables can easily be created with `renderPlot()` and `renderTable()` in the server and their corresponding output functions, `plotOutput()` and `tableOutput()` in the user interface, respectively. Additional output can include images (`renderImage()` in the server, `imageOutput()` in the `ui`), text (`renderText()` in the server, `textOutput()` in the `ui`), and even dynamic user interface options (`renderUI()` in the server, `uiOutput()` in the `ui`), where the user selection depends on input from another component of the web app. The output options in a shiny app can also be greatly expanded using the `htmlwidgets` package (in this context, “widget” does not refer to a `ui` selection). The `htmlwidgets` package (Vaidyanathan et al., 2021) creates R bindings for a shiny application to use existing JavaScript libraries, allowing the creation of additional interactive visualizations (e.g., `plotly` and `leaflet`). As before, a shiny developer only needs to understand R to use these tools.

The layout of the dashboard as seen from the user’s perspective can also be modified by the developer. The template from above uses the standard `fluidPage()` design that organizes the

user interface into a simple row/column format. Although the simple histogram template does not have any visual cues as to its organization on the web page, the `fluidPage()` layout has placed the numeric input selection and plot on separate rows defined by standard HTML tags created by shiny. The content is scaled automatically to fill the dimensions of the browser or mobile device of the user. The **flexdashboard** (Iannone et al., 2020) and **shinydashboard** (Chang & Borges Ribeiro, 2021) packages can be used to create more flexible dashboard layouts with content placed in specific CSS elements, such as boxes or tabs that are organized logically to improve how a user engages with the web app. The `flexdashboard` package uses `rmarkdown` (Xie et al., 2021), as compared to an R script used by most shiny apps. In fact, any `rmarkdown` file can be rendered using shiny with `runtime: shiny` in the YAML code at the top, allowing those already familiar with `rmarkdown` to easily embed shiny components in a rendered HTML document.

Using the above concepts and packages allows a developer to create more involved and useful applications, such as creating custom micromap plots and options to allow a user to explore patterns more easily than using a conventional R script. In the following section, a simple shiny application is described to demonstrate how it can be used with the **micromap** package.

9.3.2 Minimal Micromap Example

A simple example of a shiny application using the **micromap** package (Payton & Olsen, 2015) is described in this section. The application is accessible at https://beckmw.shinyapps.io/micromap_app/ using the RStudio shinyapps.io (<https://www.shinyapps.io/>) platform. The source code for the application is available on GitHub at https://github.com/fawda123/micromap_app and archived through Zenodo at <https://doi.org/10.5281/zenodo.6532271> (Beck, 2022). The services used to host and archive this application and reasons for their use are described in the next section.

The `micromap` web app is a simple extension of the `server` and `ui` format described in the previous section. The structure is similar with a notable difference that the server and user interface components are in separate .R files, although they can also be placed in a single file as above. The separation into two scripts is a decision made by the developer based on personal preference, rather than a functional difference. For example, a developer may prefer to use two files to simplify working with the code for each component. The application uses the `USstates` dataset as the spatial polygon boundaries and the `state.x77` dataset from the **datasets** package that is included with the base R installation. The app plots the chosen state data from `state.x77` (e.g., population, income, etc.) as two dot plots, with the state labels on the left and maps on the right using the `mmpplot()` function (see Chapter 2). The application allows a user to select which state data to show in the dot plots, the number of perceptual groups, whether a median value is shown, and which dot plot determines the ordering of the rows in the micromap (Figure 9.4).

The ui.R file is as follows:

```
library(shiny)
library(shinythemes)
library(datasets)

# variables
vrs <- colnames(x = state.x77)
```

Micromap Shiny Application

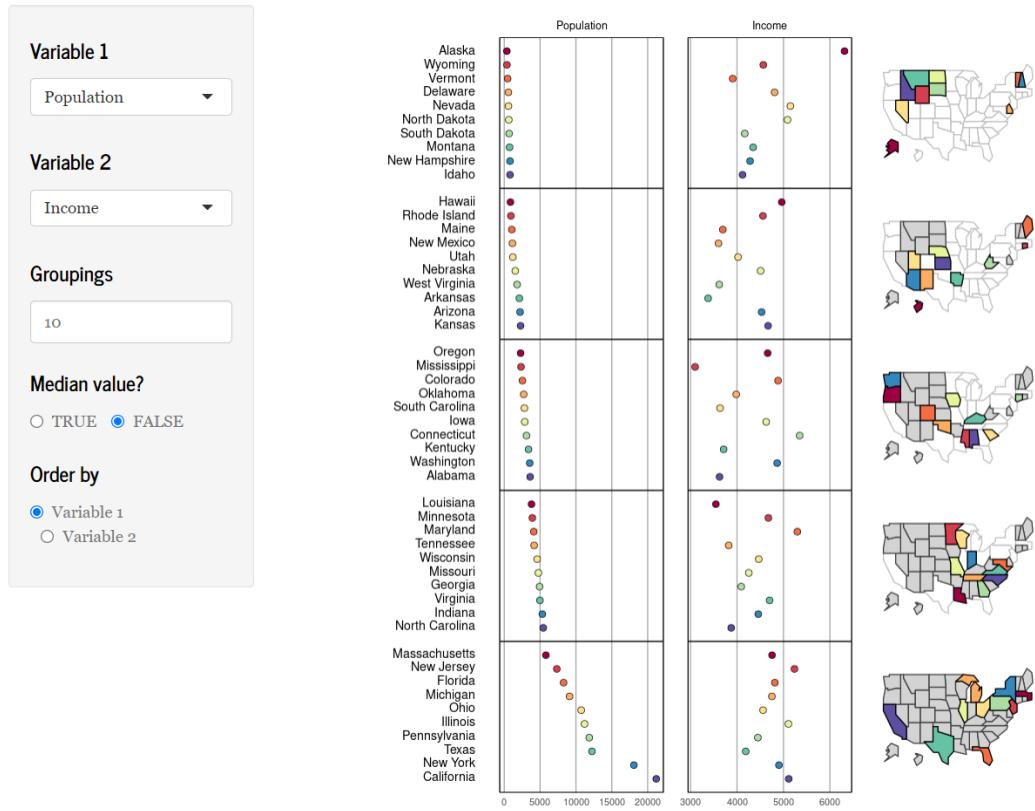


FIGURE 9.4 A minimal working shiny application using the micromap package.

```
# define ui for application
ui <- fluidPage(
  theme = shinytheme(theme = "journal"),
  titlePanel(title = "Micromap Shiny Application"),
  sidebarLayout(
    # side bar
    sidebarPanel(
      width = 2,
      selectInput(
        inputId = "vr1",
        label = h4("Variable 1"),
        choices = vrs,
        selected = vrs[1]
      ),
      selectInput(
        inputId = "vr2",
        label = h4("Variable 2"),
        choices = vrs,
        selected = vrs[2]
      )
    ),
    mainPanel(
      # main panel
      # ... (omitted for brevity)
    )
  )
)
```

```

choices = vrs,
selected = vrs[2]
),
numericInput(
  inputId = "grps",
  label = h4("Groupings"),
  value = 10,
  min = 1,
  max = 50
),
radioButtons(
  inputId = "medval",
  label = h4("Median value?"),
  choices = c(TRUE, FALSE),
  selected = c(FALSE),
  inline = TRUE
),
radioButtons(
  inputId = "ords",
  label = h4("Order by"),
  choices = c("Variable 1", "Variable 2"),
  selected = c("Variable 1"),
  inline = TRUE
)
),

# main panel
mainPanel(
  plotOutput(outputId = "p1", width = "700px", height = "750px")
)
)
)
)

```

The ui.R file begins by importing the relevant package dependencies, specifically the **shiny**, **shinythemes**, and **datasets** packages. The **vrs** object is created as options to use for selecting variables from the **state.x77** data for the micromap dot plots with the input widgets. Various components of the web app are created within the **ui** object, specifically the **fluidPage()** layout, a custom CSS theme from the **shinythemes** package (Chang, 2021), several user inputs that are sent to the server described below, and finally the micromap plot created in the server and defined as output in the user interface by **plotOutput()**. The user inputs include options to select which variables to plot from a menu with **selectInput()**, numeric options for the number of perceptual groups with **numericInput()**, and radio buttons with **radioButtons()** for selecting whether the median is shown on the micromap and which variable in the dot plots is used for the ordering. Other functions define the visual layout, notably the **sidebarLayout()** that places the inputs in a **sidebarPanel()** on the left and the main plot in a **mainPanel()** on the right (Figure 9.4).

The server.R file is as follows:

```
library(shiny)
library(micromap)
library(datasets)

# data
statdat <- state.x77

# spatial data
data(USstates)

# define server for the application
server <- function(input, output) {
  output$p1 <- renderPlot({

    # include median value, groupings
    medval <- as.logical(x = input$medval)
    grps <- input$grps

    # input variables, dates, and order
    vr1 <- input$vr1
    vr2 <- input$vr2

    # name the variable for ordering
    ords <- input$ords
    ords <- gsub(pattern = "Variable\\s", replacement = "vr", x = ords)

    # subset variables and dates, combine
    topo <- statdat[, c(vr1, vr2)]
    topo <- data.frame(topo)
    names(topo) <- c("vr1", "vr2")
    topo$ST <- row.names(x = topo)

    # create map table for polygons
    polys <- create_map_table(tmp.map = USstates, IDcolumn = "ST_NAME")

    # plot
    mmpplot(
      stat.data = topo,
      map.data = polys,
      panel.types = c("labels", "dot", "dot", "map"),
      panel.data = list("ST", "vr1", "vr2", NA),
      ord.by = ords,
      median.row = medval,
      grouping = grps,
      map.link = c("ST", "ID"),
      panel.att = list(
        list(2, header = vr1),
        list(3, header = vr2)
      )
    )
  })
}
```

```

    )
})
}

```

The server components begin by importing the dependencies for the file, specifically the **shiny**, **micromap**, and **datasets** packages. The **state.x77** data is then assigned to the **statdat** object and the *USstates* spatial polygons data frame are also imported from the **micromap** package. Both of these data objects are used within the server code, which is the remainder of the script.

The code within the **server** object creates a micromap plot object for output (**output\$p1**) to the user interface using the **renderPlot()** function from **shiny**. Within **renderPlot()**, the inputs from the user interface are used to define what data are plotted. Specifically, **input\$medval** is a logical argument that determines if the median value is shown in the micromap, **input\$grps** is a numeric value for the number of perceptual groups, **input\$ords** is used to create a character string that defines the ordering of the dot plot based on the variable name, and **input\$vr1** and **input\$vr2** are the user selections for which variables in **statdat** are used for the dot plots. The **statdat** object is then subset and renamed based on the variable selections. The rest of the code is the conventional setup for a linked micromap with **mmpplot**, i.e., **create_map_table()** is used to create the input for the **map.data** argument in **mmpplot** and then **mmpplot** is used with the data subset to create the final plot.

The web app is a proof of concept for how a micromap can be used in a shiny framework. It is simple by design to expand on the concepts described earlier in this chapter. In practice, **shiny** can be used to create more complex and rich web apps for micromaps to improve how users engage with spatial data in different contexts. One contemporary example is provided by the Virginia Department of Environmental Quality (DEQ) and is available at <https://evjones.shinyapps.io/FreshwaterProbMonEDA/> (Figure 9.5). The application is used by Virginia DEQ to support an annual report for their freshwater probabilistic monitoring program (additional examples from Virginia DEQ are in Chapter 13). Readers are invited to explore this application to better understand the full potential of **shiny** as a platform for facilitating the delivery of scientific products.

9.3.3 Getting Shiny Online

Using **shiny** to create web apps includes additional considerations that are equally, if not more, important than the development process. First, **shiny** allows a developer to create an application that is accessible through a web browser. Exposing the application outside of the local environment (i.e., the developer's personal computer) requires additional steps to make the the app accessible by others through a URL web address. There are several options for hosting a **shiny** application online and each varies in the ease of use, cost, and options desired by the developer. Some of these options are:

1. Deploy to [shinyapps.io](https://www.shinyapps.io/) (<https://www.shinyapps.io/>) directly from the RStudio console using the **rsconnect** R package (Atkins et al., 2021). The server is maintained by RStudio and the **rsconnect** package identifies all dependencies required for an application on deployment. This option may be preferred for novice **shiny** developers because knowledge of server management is not required. However, the service uses a tiered pricing plan. The free tier is good for very simple apps, but a larger plan will be needed if a developer expects others to have any meaningful engagement with an app.

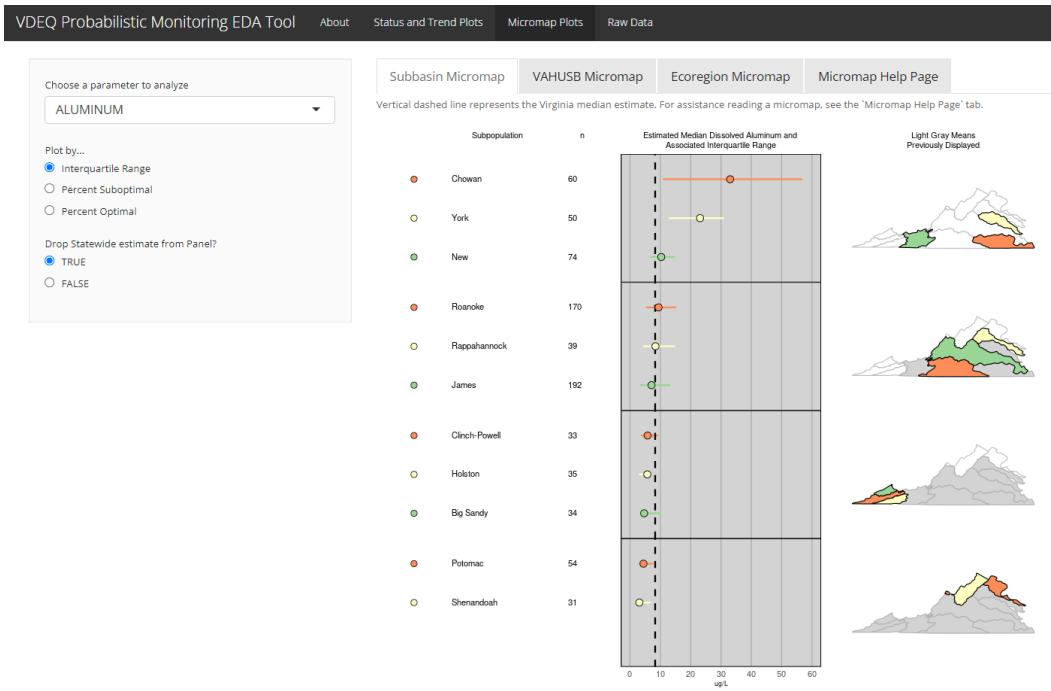


FIGURE 9.5 A screenshot of a shiny app with linked micromaps from the Virginia Department of Environmental Quality, available at <https://evjones.shinyapps.io/FreshwaterProbMonEDA/>.

2. The open-source Shiny Server (<https://rstudio.com/products/shiny/shiny-server/>) from RStudio can be used for full ownership of an application. This requires more extensive knowledge of server management, specifically on a Unix operating system. A developer will be responsible for maintaining the Shiny Server, RStudio Server, and any of the application dependencies. However, this may be a more practical option if resources are limited and an organization desires full control of web resources.
3. RStudio Connect (<https://rstudio.com/products/connect/>) can be used as a commercial solution to hosting shiny applications. RStudio Connect is a business model version of multiple resources provided by RStudio in an online format. The services are marketed for larger organizations with greater enterprise needs, as opposed to an individual developer.
4. Docker (<https://www.docker.com/>) shiny apps can also be created as an alternative solution that leverages both the open source versions of Shiny Server and RStudio Server. Docker allows a developer to solve portability and versioning problems with software management that is independent of the operating system. Docker takes a “snapshot” of a local software environment (i.e., R version, package dependencies, etc.) and creates a standalone application (or image) that can run on any server that uses Docker. Server maintenance is also greatly simplified because only Docker is required to run a shiny app. However, additional expertise on using Docker is required.

In addition to hosting, a shiny developer must consider how the source code is managed,

both during development and for long-term maintenance. Use of version control software, such as Git (<https://git-scm.com/>), and a web-hosting service for sharing version control projects, such as GitHub (<https://github.com/>), should be used to create an accessible, long-term record of the development history of an app. This creates a record of changes that a developer can retrieve at any time and allows others to view and reuse code from an existing app. In the latter case, licensing options are available on GitHub so that a developer can define how code can be reused from an existing repository. GitHub is also linked to Zenodo (<https://zenodo.org/>) to archive versions of a repository as a stand-alone set of files and assign a unique identifier that can be cited and tracked. This service extends the utility of GitHub by establishing a stronger precedent for “FAIR” data practices as a tenet of broader open science principles (Mons, 2018; M. D. Wilkinson et al., 2016).

9.4 Summary and Further Reading

This chapter provided a general overview of how **shiny** can be used to extend the utility of linked micromaps in a web-based framework. Section 9.1 described the benefits and goals of developing a web app, section 9.2 described some early examples of micromap web apps, and section 9.3 covered the very basics of shiny, a minimal example, and some guidance for hosting an app online. The reader is encouraged to view other chapters in this text for additional details on building micromaps with **micromap** (Chapter 2) or **micromapST** (Chapter 3) as a precursor to using these packages with shiny. Further, Chapters 13 and 14 provide details on how micromaps have been applied in the environmental and medical fields, respectively, and can serve as additional motivation for using shiny in different disciplines.

The overview of shiny and its reactive principles in section 9.3 provided only the basics of shiny’s functionality and additional resources should be consulted to develop a greater understanding of shiny in practice. Wickham (2021) and Sievert (2020), in particular, cover these principles in greater detail. Modularity is an additional concept that can improve how an app functions, specifically in production (Fay et al., 2022). Modules address complexity in an app by creating standalone pieces of code for creating input or output, much like functions in conventional R context, while also addressing labeling conflicts that are common in complex web apps. Saia et al. (2022) also provide an excellent overview of best practices for app development in a “ten simple rules” format that caters to the research community that typically has no formal training in these tools. Other important considerations for app development include understanding fundamentals of user interface design (i.e, the **ui** component of an app), which is critical for guiding and improving how a user engages with an app (see Chapter 6 in Fay et al., 2022). The best developed code for an app will have minimal value if a user is unable to navigate the front end.

Shiny is also not the the only tool available for developing a web app. As mentioned above, shiny is useful if a developer is already familiar with R and comfortable creating an app with code. Those with web experience may prefer using the core tools, such as Java, HTML, and CSS, although this may create additional challenges when workflows are not entirely within the R environment. Conversely, other platforms provide a graphical user interface (GUI), e.g., ArcGIS StoryMaps (<https://storymaps.arcgis.com/>), Tableau (<https://www.tableau.com/>), or Power BI <https://powerbi.microsoft.com/>), that can be used for app development, eliminating the need for direct coding. For GUI-driven development, functionality may be sacrificed for ease of use because a developer may not have complete control over the server and user interface components. In all cases, shiny is a compromise between directly using web

programming languages and more simple development software. Shiny offers a platform that is currently relevant for R users, but it is inevitable that this work space will change as new tools address the evolving needs of the research community and those that need scientific products to make informed decisions. The role of R and shiny in this future environment will necessarily change and paying attention to these dynamics is critical in understanding how web apps can help communicate scientific information.

References

- Atkins, A., McPherson, J., & Allaire, J. J. (2021). *rsconnect: Deployment Interface for R Markdown Documents and Shiny Applications* [R package version 0.8.25 (<https://CRAN.R-project.org/package=rsconnect>)].
- Beck, M. W. (2022). fawda123/micromap_app: V2.0.0 (Micromap Shiny Application Using State Data) [Zenodo Web Page, <https://doi.org/10.5281/zenodo.6532271>].
- Bell, S. B., Hoskins, R. E., Pickle, L. W., & Wartenberg, D. (2006). Current Practices in Spatial Analysis of Cancer Data: Mapping Health Statistics to Inform Policymakers and the Public [<https://doi.org/10.1186/1476-072X-5-49>]. *International Journal of Health Geographics*, 5, 49.
- Carr, D. B., Valliant, R., & Rope, D. J. (1996). Plot Interpretation and Information Webs: A Time-Series Example from the Bureau of Labor Statistics. *Statistical Computing and Statistical Graphics Newsletter*, 7(2), 19–26.
- Chang, W. (2021). *shinythemes: Themes for Shiny* [R package version 1.2.0 (<https://CRAN.R-project.org/package=shinythemes>)].
- Chang, W., & Borges Ribeiro, B. (2021). *shinydashboard: Create Dashboards with 'Shiny'* [R package version 0.7.2 (<https://CRAN.R-project.org/package=shinydashboard>)].
- Chang, W., Cheng, J., Allaire, J. J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., & Borges, B. (2021). *shiny: Web Application Framework for R* [R package version 1.7.1 (<https://CRAN.R-project.org/package=shiny>)].
- Fay, C., Rochette, S., Guyader, V., & Girard, C. (2022). *Engineering Production-Grade Shiny Apps*. CRC Press, Boca Raton, FL.
- Iannone, R., Allaire, J. J., & Borges, B. (2020). *flexdashboard: R Markdown Format for Flexible Dashboards* [R package version 0.5.2 (<https://CRAN.R-project.org/package=flexdashboard>)].
- Mons, B. (2018). *Data Stewardship for Open Science: Implementing FAIR Principles*. CRC Press, Boca Raton, FL.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Perrier, V., Meyer, F., & Granjon, D. (2022). *shinyWidgets: Custom Inputs Widgets for Shiny* [R package version 0.6.4 (<https://CRAN.R-project.org/package=shinyWidgets>)].
- Pickle, L. W., Pearson Jr., J. B., & Carr, D. B. (2015). micromapST: Exploring and Communicating Geospatial Patterns in US State Data [<https://doi.org/10.18637/jss.v063.i03>]. *Journal of Statistical Software*, 63(3), 1–25.
- Saia, S. M., Nelson, N. G., Young, S. N., Parham, S., & Vandegrift, M. (2022). Ten Simple Rules for Researchers Who Want to Develop Web Apps [<https://doi.org/10.1371/journal.pcbi.1009663>]. *PLoS Computational Biology*, 18(1), e1009663.
- Sievert, C. (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. CRC Press, Boca Raton, FL.

- Symanzik, J., Axelrad, D. A., Carr, D. B., Wang, J., Wong, D., & Woodruff, T. J. (1999). HAPs, Micromaps and GPL — Visualization of Geographically Referenced Statistical Summaries on the World Wide Web. *Annual Proceedings (ACSM-WFPS-PLSO-LSAW 1999 Conference CD)*. American Congress on Surveying & Mapping.
- Vaidyanathan, R., Xie, Y., Allaire, J. J., Cheng, J., Sievert, C., & Russell, K. (2021). *htmlwidgets: HTML Widgets for R* [R package version 1.5.4 (<https://CRAN.R-project.org/package=htmlwidgets>)].
- Wickham, H. (2019). *Advanced R (Second Edition)* [<https://adv-r.hadley.nz/>]. CRC Press, Boca Raton, FL.
- Wickham, H. (2021). *Mastering Shiny: Build Interactive Apps, Reports, and Dashboards Powered by R* [<https://mastering-shiny.org/>]. O'Reilly Media, Inc., Sebastopol, CA.
- Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016). The FAIR Guiding Principles for Scientific Data Management and Stewardship [<https://doi.org/10.1038/sdata.2016.18>]. *Scientific Data*, 3, 160018.
- Xie, Y., Dervieux, C., & Riederer, E. (2021). *R Markdown Cookbook* [<https://bookdown.org/yihui/rmarkdown-cookbook>]. CRC Press, Boca Raton, FL.

10

Conditioned Micromaps

BRENT D. MAST

Conditioned micromaps, sometimes also called conditioned choropleth maps (CC maps) are the second of the three main variations of micromap visualizations. They allow an exploratory investigation of two explanatory variables and one response variable, usually in a geographic setting. The reader will learn how to create such conditioned micromaps and how to interpret them.

10.1 Introduction

As a reminder, see Chapter 1 for general style requirements for our *Micromap Plots in R* book. In particular, please do the following:

- Introduce meaningful labels for the sections, figures, and tables in your chapter.
 - Create index entries for all R packages (such as the **micromap** R package) and for all datasets (such as the *USstates* and *edPov* datasets) that are used in your chapter.
 - Include references for R packages and publications related to your chapter, such as for the **micromap** (Payton & Olsen, 2015) and **micromapST** (Carr & Pearson Jr., 2015) R packages and some micromap articles, book chapters, and books (Carr, 2001; Carr & Pickle, 2010; Symanzik & Carr, 2008).
 - Also create index entries for main topics such as linked micromap plots, conditioned choropleth maps, perceptual group, color blindness,, and quantile-quantile plot.
-

10.2 Main

Here goes the main content of your chapter. Introduce additional sections as needed.

For convenience, Figures 10.1 and 10.4 show basic conditioned choropleth maps (CCmaps)plot

```
# Original CCmaps function taken from the maptools R package (now retired and archived on CRAN)

library(lattice)
```

```

# trellis.par.set(sp.theme()) # sets bpy.colors() ramp
# demo(meuse, ask = FALSE, echo = FALSE)
# l2 = list("SpatialPolygonsRescale", layout.north.arrow(), offset = c(181300,329800),
#   scale = 400)
# l3 = list("SpatialPolygonsRescale", layout.scale.bar(), offset = c(180500,329800),
#   scale = 500, fill=c("transparent","black"))
# l4 = list("sp.text", c(180500,329900), "0")
# l5 = list("sp.text", c(181000,329900), "500 m")
#
# meuseplot <- spplot(meuse, c("ffreq"), sp.layout=list(l2,l3,l4,l5), col.regions= "black",
#   pch=c(1,2,3), key.space=list(x=0.1,y=.95,corner=c(0,1)))
#
# class(meuseplot)
# lattice:::print.trellis(x = meuseplot)

# library(maptools)
library(sp)
# library(foreign)
# library(grid)
# library(lattice)
# library(stats)
# library(utils)
# library(graphics)
# library(grDevices)
# library(methods)
# library(raster)
library(sf)

myCCmaps <- function(
  obj, zcol = NULL, cvar = NULL, cvar.names = NULL, ...,
  names.attr, scales = list(draw = FALSE),
  xlab = NULL, ylab = NULL,
  aspect = mapasp(obj, xlim, ylim), sp.layout = NULL,
  xlim = sp::bbox(obj)[1, ], ylim = sp::bbox(obj)[2, ])
{
  stopifnot(is(obj, "SpatialPolygonsDataFrame"))
  stopifnot(!is.null(zcol), !is.null(cvar))
  n <- length(slot(obj, "polygons"))
  stopifnot(length(zcol) == 1L)
  ncc <- length(cvar)
  stopifnot(ncc <= 2, ncc > 0)
  if (is.null(cvar.names)) {
    cvar.names <- names(cvar)
  }
  nlcc <- integer(ncc)
  lcc <- vector(mode = "list", length = ncc)
  fcc <- logical(ncc)
  for (i in 1:ncc) {
    ccc <- class(cvar[[i]])
    stopifnot(ccc %in% c("factor", "shingle"))
  }
}

```

```

fcc[i] <- ccc == "factor"
stopifnot(length(cvar[[i]]) == n)
nlcc[i] <- nlevels(cvar[[i]])
lcc[[i]] <- levels(cvar[[i]])
}
obj <- obj[zcol]
zcol <- names(obj)
Outside <- function(x, y, z) (x < y | x > z)
if (ncc == 1) {
  if (fcc[1]) {
    for (j in 1:nlcc[1]) {
      vn <- paste(cvar.names[1], lcc[[1]][j], sep = "_")
      io <- as.character(cvar[[1]]) != lcc[[1]][j]
      obj[[vn]] <- obj[[zcol]]
      is.na(obj[[vn]]) <- io
    }
  } else {
    ilcc <- do.call("rbind", lcc[[1]])
    for (j in 1:nlcc[1]) {
      vn <- paste(cvar.names[1], j, sep = "_")
      io <- Outside(cvar[[1]], ilcc[j, 1], ilcc[j, 2])
      obj[[vn]] <- obj[[zcol]]
      is.na(obj[[vn]]) <- io
    }
  }
  nms <- names(obj)
  nms <- nms[-(match(zcol, nms))]
  if (fcc[1]) {
    lattice:::print.trellis(
      x = splot(obj,
                 zcol = nms, ..., scales = scales,
                 xlab = xlab, ylab = ylab,
                 aspect = aspect, sp.layout = sp.layout,
                 xlim = xlim, ylim = ylim,
                 strip = strip.custom(
                   which.given = 1,
                   factor.levels = lcc[[1]],
                   par.strip.text = list(cex = 0.8),
                   bg = "grey95"
                 )
               )
  }
} else {
  lattice:::print.trellis(
    x = splot(obj,
               zcol = nms, ..., scales = scales,
               xlab = xlab, ylab = ylab,
               aspect = aspect, sp.layout = sp.layout,
               xlim = xlim, ylim = ylim,

```

```

strip = strip.custom(
  which.given = 1,
  shingle.intervals = as.matrix(lcc[[1]]),
  var.name = cvar.names[1],
  par.strip.text = list(cex = 0.8),
  bg = "grey95",
  fg = "grey75"
)
)
)
)
}
} else {
  if (all(fcc)) {
    for (i in 1:nlcc[1]) {
      for (j in 1:nlcc[2]) {
        vn <- paste(cvar.names[1], lcc[[1]][i], cvar.names[2],
                    lcc[[2]][j],
                    sep = "_")
        )
        obj[[vn]] <- obj[[zcol]]
        ioi <- as.character(cvar[[1]]) != lcc[[1]][i]
        ioj <- as.character(cvar[[2]]) != lcc[[2]][j]
        io <- ioi | ioj
        is.na(obj[[vn]]) <- io
      }
    }
    nms <- names(obj)
    nms <- nms[-(match(zcol, nms))]
    lcc1 <- lcc[[1]]
    xlcc <- NULL
    for (i in 1:nlcc[1]) {
      xlcc <- c(xlcc, rep(lcc1[i], nlcc[2]))
    }
    lcc2 <- lcc[[2]]
    xlcc2 <- rep(lcc2, nlcc[1])
    lattice:::print.trellis(
      x = splot(obj,
                zcol = nms, ..., scales = scales,
                xlab = xlab, ylab = ylab,
                aspect = aspect, sp.layout = sp.layout,
                xlim = xlim, ylim = ylim,
                strip = strip.custom(
                  which.given = 1,
                  factor.levels = xlcc,
                  par.strip.text = list(cex = 0.8),
                  bg = "grey95"
                ),
                strip.left = strip.custom(
                  which.given = 1,

```

```

factor.levels = xlcc2,
par.strip.text = list(cex = 0.8),
bg = "grey95"
)
)
)
}
} else if (any(fcc)) {
if (fcc[1]) {
jlcc <- do.call("rbind", lcc[[2]])
for (i in 1:nlcc[1]) {
for (j in 1:nlcc[2]) {
vn <- paste(cvar.names[1], lcc[[1]][i], cvar.names[2],
j,
sep = "_"
)
obj[[vn]] <- obj[[zcol]]
ioi <- as.character(cvar[[1]]) != lcc[[1]][i]
ioj <- Outside(cvar[[2]], jlcc[j, 1], jlcc[j, 2])
io <- ioi | ioj
is.na(obj[[vn]]) <- io
}
}
nms <- names(obj)
nms <- nms[-(match(zcol, nms))]
lcc1 <- lcc[[1]]
xlcc <- NULL
for (i in 1:nlcc[1]) {
xlcc <- c(xlcc, rep(lcc1[i], nlcc[2]))
}
lcc2 <- matrix(unlist(lcc[[2]]), ncol = 2, byrow = TRUE)
xlcc2 <- matrix(rep(t(lcc2), nlcc[1]),
byrow = TRUE,
ncol = 2
)
lattice:::print.trellis(
x = splot(obj,
zcol = nms, ..., scales = scales,
xlab = xlab, ylab = ylab,
aspect = aspect, sp.layout = sp.layout,
xlim = xlim, ylim = ylim,
strip = strip.custom(
which.given = 1,
factor.levels = xlcc,
par.strip.text = list(cex = 0.8),
bg = "grey95"
),
strip.left = strip.custom(
which.given = 1,
shingle.intervals = xlcc2,

```

```

    var.name = cvar.names[2],
    par.strip.text = list(cex = 0.8),
    bg = "grey95",
    fg = "grey75"
  )
)
)
)
}
} else {
  ilcc <- do.call("rbind", lcc[[1]])
  for (i in 1:nlcc[1]) {
    for (j in 1:nlcc[2]) {
      vn <- paste(cvar.names[1], i, cvar.names[2],
        lcc[[2]][j],
        sep = "_"
      )
      obj[[vn]] <- obj[[zcol]]
      ioi <- Outside(cvar[[1]], ilcc[i, 1], ilcc[i, 2])
      ioj <- as.character(cvar[[2]]) != lcc[[2]][j]
      io <- ioi | ioj
      is.na(obj[[vn]]) <- io
    }
  }
  nms <- names(obj)
  nms <- nms[-(match(zcol, nms))]
  lcc1 <- matrix(unlist(lcc[[1]]), ncol = 2, byrow = TRUE)
  xlcc <- matrix(ncol = 2)
  for (i in 1:nlcc[1]) {
    xlcc <- rbind(xlcc, matrix(rep(lcc1[i, ], nlcc[2]),
      ncol = 2, byrow = TRUE
    ))
  }
  xlcc <- xlcc[-1, ]
  lcc2 <- lcc[[2]]
  xlcc2 <- rep(lcc2, nlcc[1])
  lattice:::print.trellis(
    x = spplot(obj,
      zcol = nms, ..., scales = scales,
      xlab = xlab, ylab = ylab,
      aspect = aspect, sp.layout = sp.layout,
      xlim = xlim, ylim = ylim,
      strip = strip.custom(
        which.given = 1,
        shingle.intervals = xlcc,
        var.name = cvar.names[1],
        par.strip.text = list(cex = 0.8),
        bg = "grey95",
        fg = "grey75"
      ),
      strip.left = strip.custom(

```

```

        which.given = 1,
        factor.levels = xlcc2,
        par.strip.text = list(cex = 0.8),
        bg = "grey95"
    )
)
)
)
}
} else {
  ilcc <- do.call("rbind", lcc[[1]])
  jlcc <- do.call("rbind", lcc[[2]])
  for (i in 1:nlcc[1]) {
    for (j in 1:nlcc[2]) {
      vn <- paste(cvar.names[1], i, cvar.names[2], j, sep = "_")
      obj[[vn]] <- obj[[zcol]]
      ioi <- Outside(cvar[[1]], ilcc[i, 1], ilcc[i, 2])
      ioj <- Outside(cvar[[2]], jlcc[j, 1], jlcc[j, 2])
      io <- ioi | ioj
      is.na(obj[[vn]]) <- io
    }
  }
  nms <- names(obj)
  nms <- nms[-match(zcol, nms)]
  lcc1 <- matrix(unlist(lcc[[1]]), ncol = 2, byrow = TRUE)
  xlcc <- matrix(ncol = 2)
  for (i in 1:nlcc[1]) {
    xlcc <- rbind(xlcc, matrix(rep(lcc1[i, ], nlcc[2]),
      ncol = 2, byrow = TRUE
    ))
  }
  xlcc <- xlcc[-1, ]
  lcc2 <- matrix(unlist(lcc[[2]]), ncol = 2, byrow = TRUE)
  xlcc2 <- matrix(rep(t(lcc2), nlcc[1]),
    byrow = TRUE,
    ncol = 2
  )
  lattice:::print.trellis(
    x = spplot(obj,
      zcol = nms, ..., scales = scales,
      xlab = xlab, ylab = ylab,
      aspect = aspect, sp.layout = sp.layout,
      xlim = xlim, ylim = ylim,
      strip = strip.custom(
        which.given = 1,
        shingle.intervals = xlcc,
        var.name = cvar.names[1],
        par.strip.text = list(cex = 0.8),
        bg = "grey95",
        fg = "grey75"
      )
    )
  )
}

```

```

),
strip.left = strip.custom(
  which.given = 1,
  shingle.intervals = xlcc2,
  var.name = cvar.names[2],
  par.strip.text = list(cex = 0.8),
  bg = "grey95",
  fg = "grey75"
)
)
)
}
}
invisible(obj)
}

```

```

# Original CCmaps function taken from the maptools R package (now retired and archived on CRAN)

library(lattice)
library(sp)
library(sf)

myCCmaps <- function(
  obj,
  zcol = NULL,
  cvar = NULL,
  cvar.names = NULL,
  ...,
  names.attr,
  scales = list(draw = FALSE),
  xlab = NULL,
  ylab = NULL,
  aspect = mapasp(obj, xlim, ylim),
  sp.layout = NULL,
  xlim = sp::bbox(obj)[1, ],
  ylim = sp::bbox(obj)[2, ],
  pstat.function = "none",
  pstat.loc = "LL",
  pstat.digits = 0,
  pstat.cex = 1.0) {
  stopifnot(is(obj, "SpatialPolygonsDataFrame"))
  stopifnot(!is.null(zcol), !is.null(cvar))
  n <- length(slot(obj, "polygons"))
  stopifnot(length(zcol) == 1L)
  ncc <- length(cvar)
  stopifnot(ncc <= 2, ncc > 0)
  if (is.null(cvar.names)) {
    cvar.names <- names(cvar)
  }

```

```

nlcc <- integer(ncc)
lcc <- vector(mode = "list", length = ncc)
fcc <- logical(ncc)

stopifnot(pstat.function %in% c("none", "mean", "median"))
stopifnot(pstat.loc %in% c("LL", "UL", "LR", "UR"))
stopifnot(is.numeric(pstat.digits))
stopifnot(pstat.digits >= 0)
stopifnot(is.numeric(pstat.cex))
stopifnot(pstat.cex > 0)

for (i in 1:ncc) {
  ccc <- class(cvar[[i]])
  stopifnot(ccc %in% c("factor", "shingle"))
  fcc[i] <- ccc == "factor"
  stopifnot(length(cvar[[i]]) == n)
  nlcc[i] <- nlevels(cvar[[i]])
  lcc[[i]] <- levels(cvar[[i]])
}
obj <- obj[zcol]
zcol <- names(obj)

Outside <- function(x, y, z) {
  (x < y | x > z)
}

CreatePStat <- function(obj, nms, j, xlim, ylim,
                       pstat.function, pstat.loc, pstat.digits, pstat.cex) {
  if (pstat.loc == "LL") {
    textloc <- c(xlim[1] + 0.2 * (xlim[2] - xlim[1]), ylim[1] + 0.1 * (ylim[2] - ylim[1]))
  } else if (pstat.loc == "UL") {
    textloc <- c(xlim[1] + 0.2 * (xlim[2] - xlim[1]), ylim[2] - 0.1 * (ylim[2] - ylim[1]))
  } else if (pstat.loc == "LR") {
    textloc <- c(xlim[2] - 0.2 * (xlim[2] - xlim[1]), ylim[1] + 0.1 * (ylim[2] - ylim[1]))
  } else if (pstat.loc == "UR") {
    textloc <- c(xlim[2] - 0.2 * (xlim[2] - xlim[1]), ylim[2] - 0.1 * (ylim[2] - ylim[1]))
  }

  if (pstat.function == "mean") {
    pmean <- round(mean(obj[[nms[j]]]), na.rm = TRUE), pstat.digits)
    list("sp.text",
         textloc,
         # Based on
         # https://stackoverflow.com/questions/16369872/how-to-create-lattice-plot-title-with-new-line-itali
         as.expression(bquote(bar(x) == .(pmean))),
         cex = pstat.cex,
         which = j
    )
  } else if (pstat.function == "median") {

```

```

pmedian <- round(median(obj[[nms[j]]], na.rm = TRUE), pstat.digits)
list("sp.text",
  textloc,
  paste("m = ", pmmedian),
  cex = pstat.cex,
  which = j
)
}
}

if (ncc == 1) {
  if (fcc[1]) {
    for (j in 1:nlcc[1]) {
      vn <- paste(cvar.names[1], lcc[[1]][j], sep = "_")
      io <- as.character(cvar[[1]]) != lcc[[1]][j]
      obj[[vn]] <- obj[[zcol]]
      is.na(obj[[vn]]) <- io
    }
  } else {
    ilcc <- do.call("rbind", lcc[[1]])
    for (j in 1:nlcc[1]) {
      vn <- paste(cvar.names[1], j, sep = "_")
      io <- Outside(cvar[[1]], ilcc[j, 1], ilcc[j, 2])
      obj[[vn]] <- obj[[zcol]]
      is.na(obj[[vn]]) <- io
    }
  }
  nms <- names(obj)
  nms <- nms[-(match(zcol, nms))]
}

if (pstat.function != "none") {
  sp.layout.add <- vector(mode = "list", length = nlcc[1])
  for (j in 1:nlcc[1]) {
    sp.layout.add[[j]] <- CreatePStat(
      obj, nms, j, xlim, ylim,
      pstat.function, pstat.loc, pstat.digits, pstat.cex
    )
  }
  sp.layout <- c(sp.layout, sp.layout.add)
}

if (fcc[1]) {
  lattice:::print.trellis(
    x = spplot(obj,
      zcol = nms, ..., scales = scales,
      xlab = xlab, ylab = ylab,
      aspect = aspect,
      sp.layout = sp.layout,
      xlim = xlim, ylim = ylim,

```



```

    }
    sp.layout <- c(sp.layout, sp.layout.add)
}

lcc1 <- lcc[[1]]
xlcc <- NULL
for (i in 1:nlcc[1]) {
  xlcc <- c(xlcc, rep(lcc1[i], nlcc[2]))
}
lcc2 <- lcc[[2]]
xlcc2 <- rep(lcc2, nlcc[1])
lattice:::print.trellis(
  x = splot(obj,
    zcol = nms, ..., scales = scales,
    xlab = xlab, ylab = ylab,
    aspect = aspect,
    sp.layout = sp.layout,
    xlim = xlim, ylim = ylim,
    strip = strip.custom(
      which.given = 1,
      factor.levels = xlcc,
      par.strip.text = list(cex = 0.8),
      bg = "grey95"
    ),
    strip.left = strip.custom(
      which.given = 1,
      factor.levels = xlcc2,
      par.strip.text = list(cex = 0.8),
      bg = "grey95"
    )
  )
)
}
} else if (any(fcc)) {
  if (fcc[1]) {
    jlcc <- do.call("rbind", lcc[[2]])
    for (i in 1:nlcc[1]) {
      for (j in 1:nlcc[2]) {
        vn <- paste(cvar.names[1], lcc[[1]][i], cvar.names[2], j, sep = "_")
        obj[[vn]] <- obj[[zcol]]
        ioi <- as.character(cvar[[1]]) != lcc[[1]][i]
        ioj <- Outside(cvar[[2]], jlcc[j, 1], jlcc[j, 2])
        io <- ioi | ioj
        is.na(obj[[vn]]) <- io
      }
    }
    nms <- names(obj)
    nms <- nms[-(match(zcol, nms))]
  }
  if (pstat.function != "none") {

```

```

sp.layout.add <- vector(mode = "list", length = nlcc[1] * nlcc[2])
for (j in 1:(nlcc[1] * nlcc[2])) {
  sp.layout.add[[j]] <- CreatePStat(
    obj, nms, j, xlim, ylim,
    pstat.function, pstat.loc, pstat.digits, pstat.cex
  )
}
sp.layout <- c(sp.layout, sp.layout.add)
}

lcc1 <- lcc[[1]]
xlcc <- NULL
for (i in 1:nlcc[1]) {
  xlcc <- c(xlcc, rep(lcc1[i], nlcc[2]))
}
lcc2 <- matrix(unlist(lcc[[2]]), ncol = 2, byrow = TRUE)
xlcc2 <- matrix(rep(t(lcc2), nlcc[1]),
  ncol = 2, byrow = TRUE
)
lattice:::print.trellis(
  x = spplot(obj,
    zcol = nms, ..., scales = scales,
    xlab = xlab, ylab = ylab,
    aspect = aspect,
    sp.layout = sp.layout,
    xlim = xlim, ylim = ylim,
    strip = strip.custom(
      which.given = 1,
      factor.levels = xlcc,
      par.strip.text = list(cex = 0.8),
      bg = "grey95"
    ),
    strip.left = strip.custom(
      which.given = 1,
      shingle.intervals = xlcc2,
      var.name = cvar.names[2],
      par.strip.text = list(cex = 0.8),
      bg = "grey95",
      fg = "grey75"
    )
  )
)
}
} else {
  ilcc <- do.call("rbind", lcc[[1]])
  for (i in 1:nlcc[1]) {
    for (j in 1:nlcc[2]) {
      vn <- paste(cvar.names[1], i, cvar.names[2], lcc[[2]][j], sep = "_")
      obj[[vn]] <- obj[[zcol]]
      ioi <- Outside(cvar[[1]], ilcc[i, 1], ilcc[i, 2])
    }
  }
}

```

```

ioj <- as.character(cvar[[2]]) != lcc[[2]][j]
io <- ioi | ioj
is.na(obj[[vn]]) <- io
}
}
nms <- names(obj)
nms <- nms[-(match(zcol, nms))]

if (pstat.function != "none") {
  sp.layout.add <- vector(mode = "list", length = nlcc[1] * nlcc[2])
  for (j in 1:(nlcc[1] * nlcc[2])) {
    sp.layout.add[[j]] <- CreatePStat(
      obj, nms, j, xlim, ylim,
      pstat.function, pstat.loc, pstat.digits, pstat.cex
    )
  }
  sp.layout <- c(sp.layout, sp.layout.add)
}

lcc1 <- matrix(unlist(lcc[[1]]), ncol = 2, byrow = TRUE)
xlcc <- matrix(ncol = 2)
for (i in 1:nlcc[1]) {
  xlcc <- rbind(
    xlcc,
    matrix(rep(lcc1[i, ], nlcc[2]),
      ncol = 2, byrow = TRUE
    )
  )
}
xlcc <- xlcc[-1, ]
lcc2 <- lcc[[2]]
xlcc2 <- rep(lcc2, nlcc[1])
lattice:::print.trellis(
  x = spplot(obj,
    zcol = nms, ..., scales = scales,
    xlab = xlab, ylab = ylab,
    aspect = aspect,
    sp.layout = sp.layout,
    xlim = xlim, ylim = ylim,
    strip = strip.custom(
      which.given = 1,
      shingle.intervals = xlcc,
      var.name = cvar.names[1],
      par.strip.text = list(cex = 0.8),
      bg = "grey95",
      fg = "grey75"
    ),
    strip.left = strip.custom(
      which.given = 1,

```

```

        factor.levels = xlcc2,
        par.strip.text = list(cex = 0.8),
        bg = "grey95"
    )
)
)
)
}
} else {
    ilcc <- do.call("rbind", lcc[[1]])
    jlcc <- do.call("rbind", lcc[[2]])
    for (i in 1:nlcc[1]) {
        for (j in 1:nlcc[2]) {
            vn <- paste(cvar.names[1], i, cvar.names[2], j, sep = "_")
            obj[[vn]] <- obj[[zcol]]
            ioi <- Outside(cvar[[1]], ilcc[i, 1], ilcc[i, 2])
            ioj <- Outside(cvar[[2]], jlcc[j, 1], jlcc[j, 2])
            io <- ioi | ioj
            is.na(obj[[vn]]) <- io
        }
    }
    nms <- names(obj)
    nms <- nms[-(match(zcol, nms))]
}

if (pstat.function != "none") {
    sp.layout.add <- vector(mode = "list", length = nlcc[1] * nlcc[2])
    for (j in 1:(nlcc[1] * nlcc[2])) {
        sp.layout.add[[j]] <- CreatePStat(
            obj, nms, j, xlim, ylim,
            pstat.function, pstat.loc, pstat.digits, pstat.cex
        )
    }
    sp.layout <- c(sp.layout, sp.layout.add)
}

lcc1 <- matrix(unlist(lcc[[1]]), ncol = 2, byrow = TRUE)
xlcc <- matrix(ncol = 2)
for (i in 1:nlcc[1]) {
    xlcc <- rbind(
        xlcc,
        matrix(rep(lcc1[i, ], nlcc[2]),
               ncol = 2, byrow = TRUE
        )
    )
}
xlcc <- xlcc[-1, ]
lcc2 <- matrix(unlist(lcc[[2]]), ncol = 2, byrow = TRUE)
xlcc2 <- matrix(rep(t(lcc2), nlcc[1]),
                 ncol = 2, byrow = TRUE
)
}

```

```

lattice:::print.trellis(
  x = spplot(obj,
    zcol = nms, ..., scales = scales,
    xlab = xlab, ylab = ylab,
    aspect = aspect,
    sp.layout = sp.layout,
    xlim = xlim, ylim = ylim,
    strip = strip.custom(
      which.given = 1,
      shingle.intervals = xlcc,
      var.name = cvar.names[1],
      par.strip.text = list(cex = 0.8),
      bg = "grey95",
      fg = "grey75"
    ),
    strip.left = strip.custom(
      which.given = 1,
      shingle.intervals = xlcc2,
      var.name = cvar.names[2],
      par.strip.text = list(cex = 0.8),
      bg = "grey95",
      fg = "grey75"
    )
  )
)
}
invisible(obj)
}

```

```

# Based on R code from the help page at
# https://maptools.r-forge.r-project.org/reference/CCmaps.html

# nc.sids_orig <- readShapeSpatial(
#   system.file("shapes/sids.shp",
#     package = "maptools"
#   )[1],
#   IDvar = "FIPSNO",
#   proj4string = CRS("+proj=longlat +ellps=clrk66")
# )

mync.sids <- st_read("data/Ch08/sids.shp",
  crs = "+proj=longlat +ellps=clrk66"
)

## Reading layer `sids' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\sids.shp'
## using driver `ESRI Shapefile'
## Simple feature collection with 100 features and 14 fields

```

```

## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: -84.32 ymin: 33.88 xmax: -75.46 ymax: 36.59
## Geodetic CRS: +proj=longlat +ellps=clrk66

nc.sids <- as_Spatial(mync.sids)
rownames(slot(nc.sids, "data")) <- as.character(nc.sids$FIPSNO)

nc.sids$ft.SID74 <- sqrt(1000) * (sqrt(nc.sids$SID74 / nc.sids$BIR74) +
  sqrt((nc.sids$SID74 + 1) / nc.sids$BIR74))
nc.sids$ft.NWBIR74 <- sqrt(1000) * (sqrt(nc.sids$NWBIR74 / nc.sids$BIR74) +
  sqrt((nc.sids$NWBIR74 + 1) / nc.sids$BIR74))
sh_nw4 <- equal.count(nc.sids$ft.NWBIR74, number = 4, overlap = 1 / 5)

myCCmaps(
  obj = nc.sids,
  zcol = "ft.SID74",
  cvar = list("Nonwhite_births" = sh_nw4),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(20),
  main = "Transformed SIDS rates 1974-8"
)

```

Transformed SIDS rates 1974-8

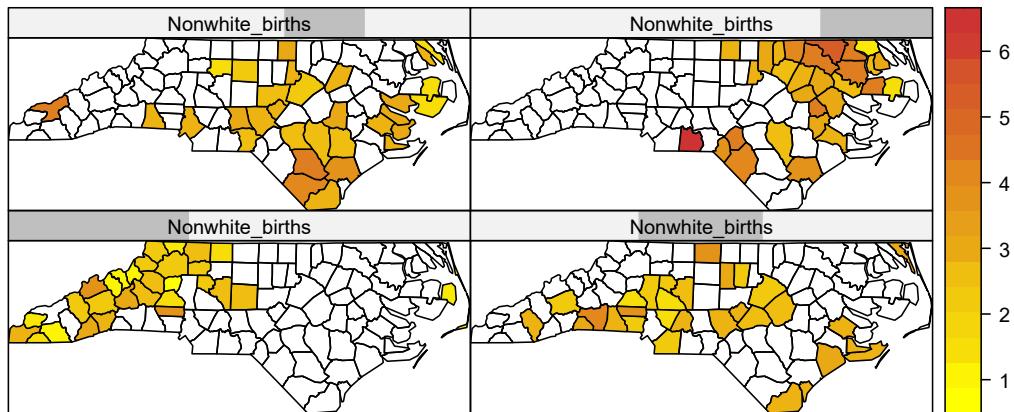


FIGURE 10.1 Trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.

```

sh_nw4nooverlap <- equal.count(nc.sids$ft.NWBIR74, number = 4, overlap = 0)

myCCmaps(
  obj = nc.sids,

```

```

zcol = "ft.SID74",
cvar = list("Nonwhite_births" = sh_nw4nooverlap),
at = seq(0, 7, by = 0.5),
col.regions = colorRampPalette(c("yellow1", "brown3"))(14),
main = "Transformed SIDS rates 1974-8",
scales = list(draw = TRUE)
)

```

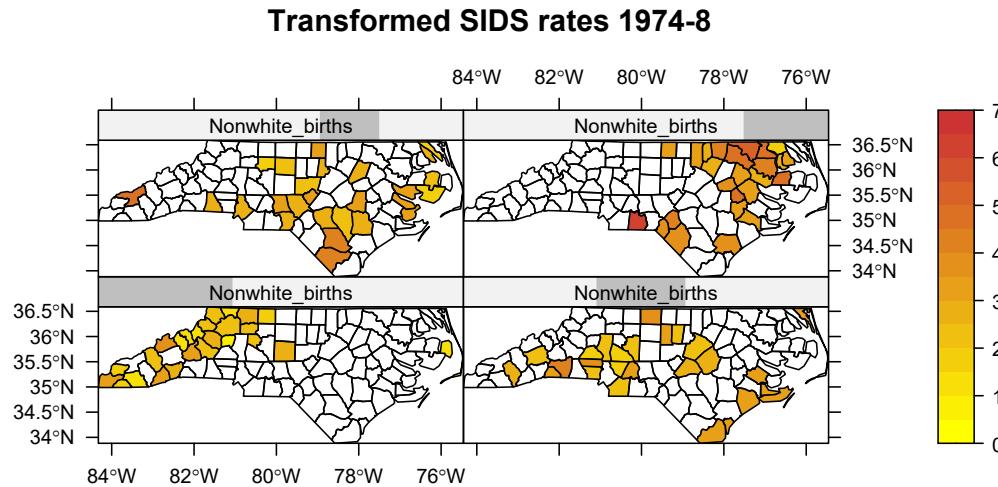


FIGURE 10.2 First modified trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.

```

# modified 2

myCCmaps(
  obj = nc.sids,
  zcol = "ft.SID74",
  cvar = list("Nonwhite_births" = sh_nw4nooverlap),
  pretty = TRUE,
  col.regions = colorRampPalette(c("yellow1", "brown3"))(14),
  main = "Transformed SIDS rates 1974-8",
  scales = list(draw = TRUE),
  pstat.function = "mean",
  pstat.digits = 2
)

```

```

# Extended R code for 2 dimensions

sh_nw3 <- equal.count(nc.sids$ft.NWBIR74, number = 3, overlap = 0)

```

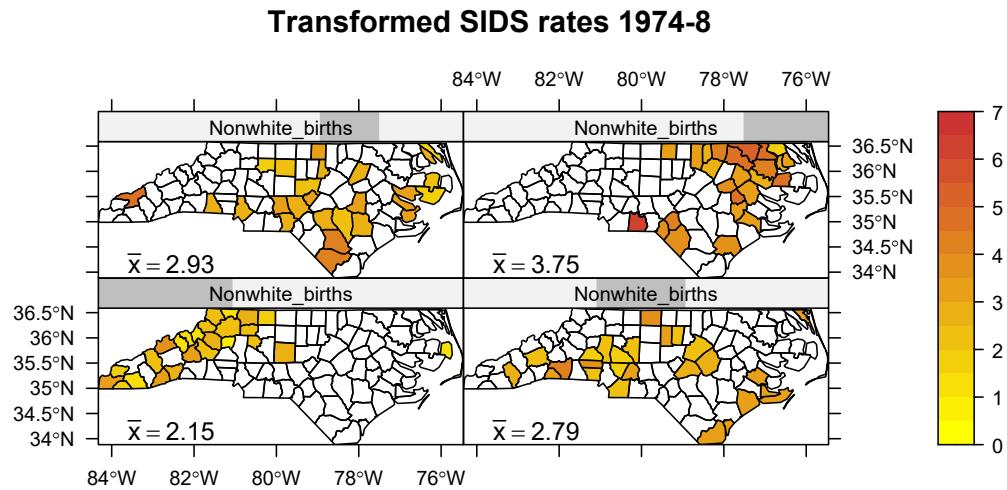


FIGURE 10.3 Second modified trivial 1-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.

```
sh_bir3 <- equal.count(nc.sids$BIR79, number = 3, overlap = 0)

myCCmaps(
  obj = nc.sids,
  zcol = "ft.SID74",
  cvar = list(
    "Nonwhite_births" = sh_nw3,
    "Births79" = sh_bir3
  ),
  at = seq(0, 7, by = 0.5),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(14),
  main = "Transformed SIDS rates 1974-78",
  layout = c(3, 3),
  pstat.function = "median",
  pstat.digits = 1,
  pstat.cex = 0.7
)
```

Use R code from the end of Section 3.3.5 from <http://www.geo.hunter.cuny.edu/~ssun/R-Spatial/mapping.html> as the basis to experiment with CCmaps in ggplot / ggmap.

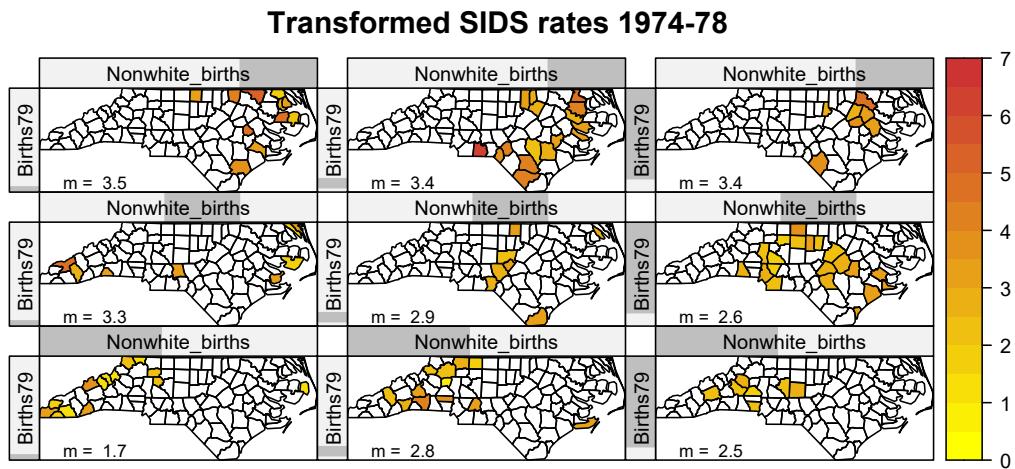


FIGURE 10.4 Trivial 2-dimensional conditioned choropleth maps example based on the CCmaps() function from the maptools R package.

10.3 WV Examples from Brent Mast

```

# Original file: WV_counties2.R
# Original output: WV_2way_CCmap.jpeg
# Original data: wv_merged.rdata

# setwd("C:\\Users\\bmast\\GD\\Chapters\\WV")
# install.packages("maptools")
# #library(maptools)
library(lattice)
library(sf)
library(dplyr)
library(sp)

poly <- st_read("data/Ch08/WV_Counties.shp",
  crs = "+proj=longlat +ellps=clrk66"
)

## Reading layer 'WV_Counties' from data source
##  `D:\\Dropbox\\JUE\\Papers\\2022_MicromapBook\\MicromapRBookSource\\data\\Ch08\\WV_Counties.shp'
##  using driver 'ESRI Shapefile'

## Simple feature collection with 55 features and 34 fields
## Geometry type: POLYGON
## Dimension:      XY

```

```
## Bounding box: xmin: -82.64 ymin: 37.2 xmax: -77.72 ymax: 40.64
## Geodetic CRS: +proj=longlat +ellps=clrk66
```

```
# summary(poly)
poly$fips <- as.character(poly$FIPS)
head(poly$fips)
```

```
## [1] "54069" "54051" "54077" "54065" "54061" "54103"
```

```
load("data/Ch08/wv_merged.Rdata")
# summary(wv_merged)
head(wv_merged$fips)
```

```
## [1] "54001" "54003" "54005" "54007" "54009" "54011"
```

```
poly2 <- inner_join(poly, wv_merged, join_by(fips))
# summary(poly2)
```

```
wv <- as_Spatial(poly2)
# summary(wv)
rownames(slot(wv, "data")) <- wv$fips
```

```
mi_ec <- equal.count(wv$MI, 3, overlap = 0)
coi_ec <- equal.count(wv$z_COI_nat, 3, overlap = 0)
```

```
myCCmaps(
  obj = wv,
  zcol = "viol_rate",
  cvar = list(
    "COI" = coi_ec,
    "Median_Income" = mi_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(20),
  main = "Violent Crimes per 10,000 Population",
  layout = c(3, 3)
)
```

```
# modified
```

```
myCCmaps(
  obj = wv,
  zcol = "viol_rate",
  cvar = list(
    "COI" = coi_ec,
    "Median_Income" = mi_ec
  ),
  at = seq(0, 400, by = 50),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(8),
```

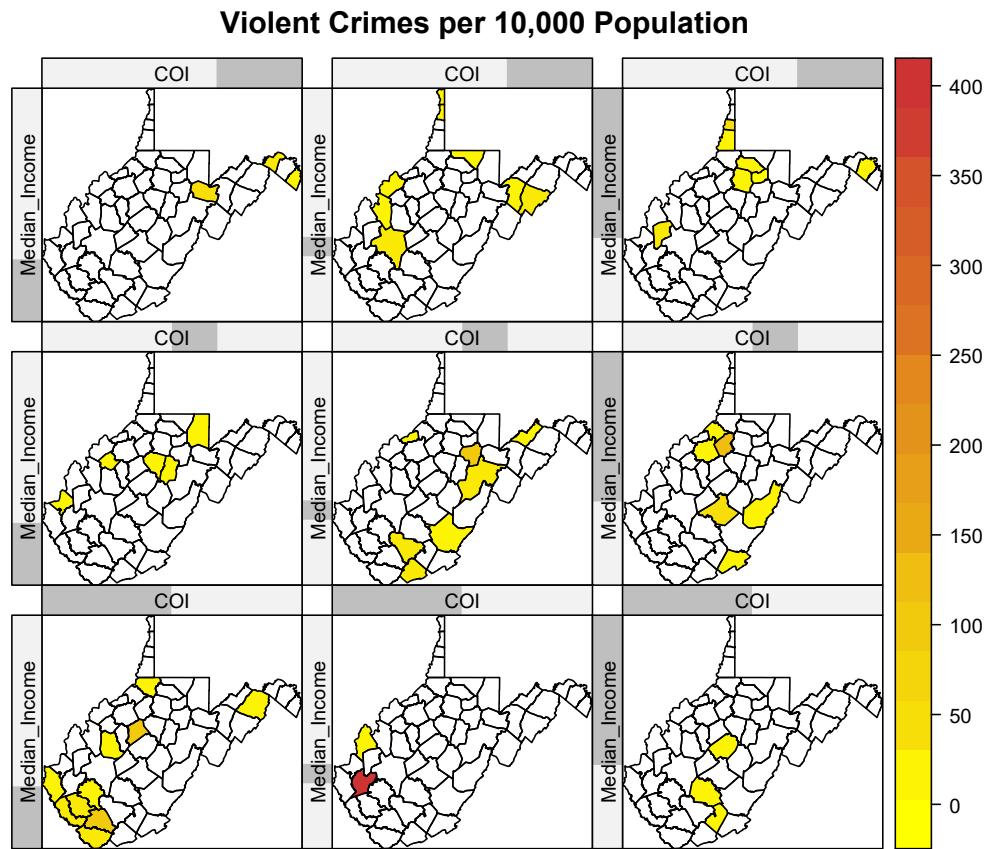


FIGURE 10.5 Conditioned choropleth map 1.

```
main = "Violent Crimes per 10,000 Population",
layout = c(3, 3),
pstat.function = "mean",
pstat.loc = "UL"
)
```

```
# Original file: WV_counties3.R
# Original output: WV_2way_CCmap2.jpeg
# Original data: No new data file

# setwd("C:\\\\Users\\\\bmast\\\\GD\\\\Chapters\\\\WV")
# install.packages("maptools")
# library(maptools)
library(lattice)
library(sf)
library(dplyr)
library(sp)
```

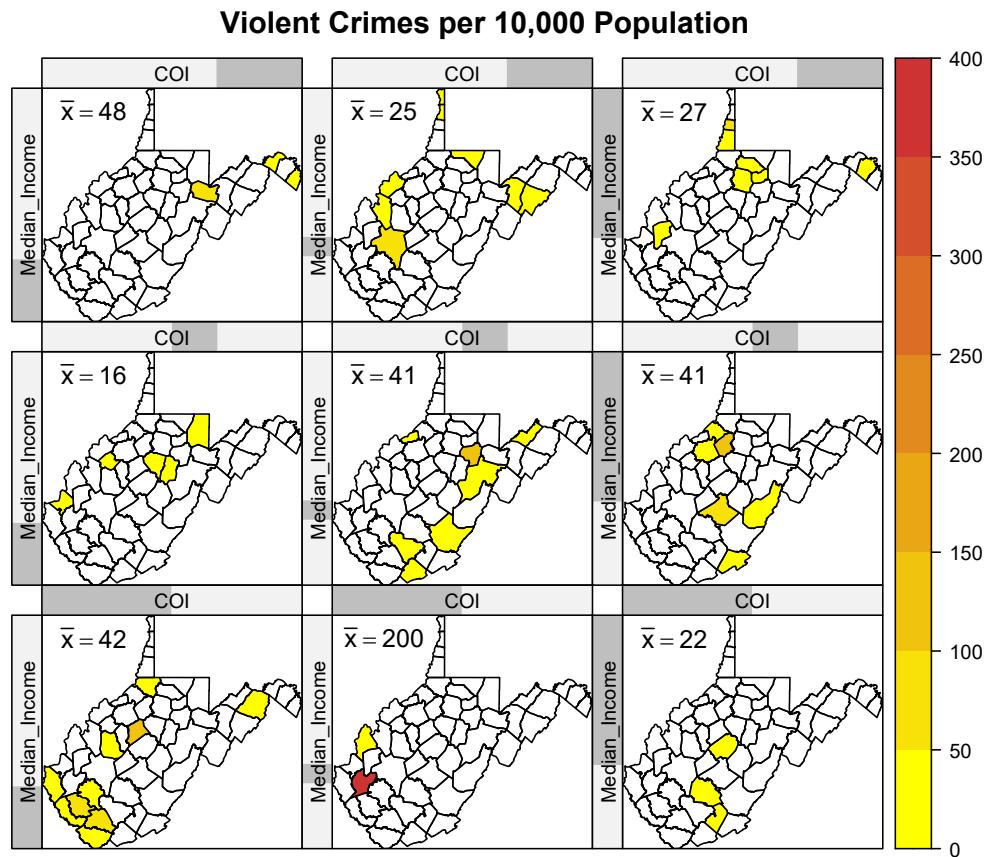


FIGURE 10.6 Modified conditioned choropleth map 1.

```

library(RColorBrewer)

poly <- st_read("data/Ch08/WV_Counties.shp",
  crs = "+proj=longlat +ellps=clrk66"
)

## Reading layer `WV_Counties' from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\WV_Counties.shp'
##   using driver `ESRI Shapefile'

## Simple feature collection with 55 features and 34 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -82.64 ymin: 37.2 xmax: -77.72 ymax: 40.64
## Geodetic CRS:  +proj=longlat +ellps=clrk66

```

```

poly$fips <- as.character(poly$FIPS)
head(poly$fips)

## [1] "54069" "54051" "54077" "54065" "54061" "54103"

load("data/Ch08/wv_merged.Rdata")
# summary(wv_merged)
head(wv_merged$fips)

## [1] "54001" "54003" "54005" "54007" "54009" "54011"

poly2 <- inner_join(poly, wv_merged, join_by(fips))
# summary(poly2)

wv <- as_Spatial(poly2)
# summary(wv)
rownames(slot(wv, "data")) <- wv$fips

prop_ec <- equal.count(wv$prop_rate, 3, overlap = 0)
viol_ec <- equal.count(wv$viol_rate, 3, overlap = 0)

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Property_Crime_Rate" = prop_ec,
    "Violent_Crime_Rate" = viol_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(20),
  main = "Child Opportunity Index",
  layout = c(3, 3)
)

# modified

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Property_Crime_Rate" = prop_ec,
    "Violent_Crime_Rate" = viol_ec
  ),
  at = seq(-0.06, 0.02, by = 0.01),
  col.regions = brewer.pal(11, "RdYlBu")[1:8],
)

```

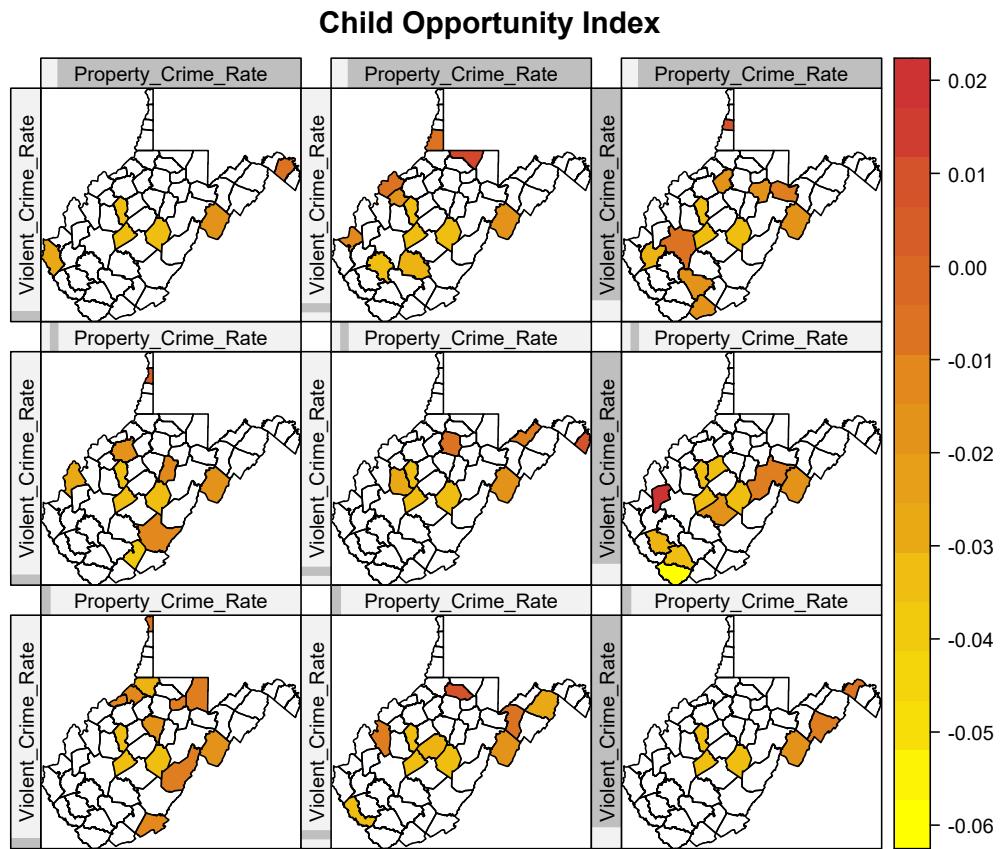


FIGURE 10.7 Conditioned choropleth map 2.

```
main = "Child Opportunity Index",
layout = c(3, 3),
pstat.function = "mean",
pstat.loc = "UR",
pstat.digits = 3,
pstat.cex = 0.8
)
```

10.4 WV Conditioned Choropleth Maps with Factors

```
female_majority <- as.factor(wv$FEMALES > wv$MALES)
levels(female_majority)
```

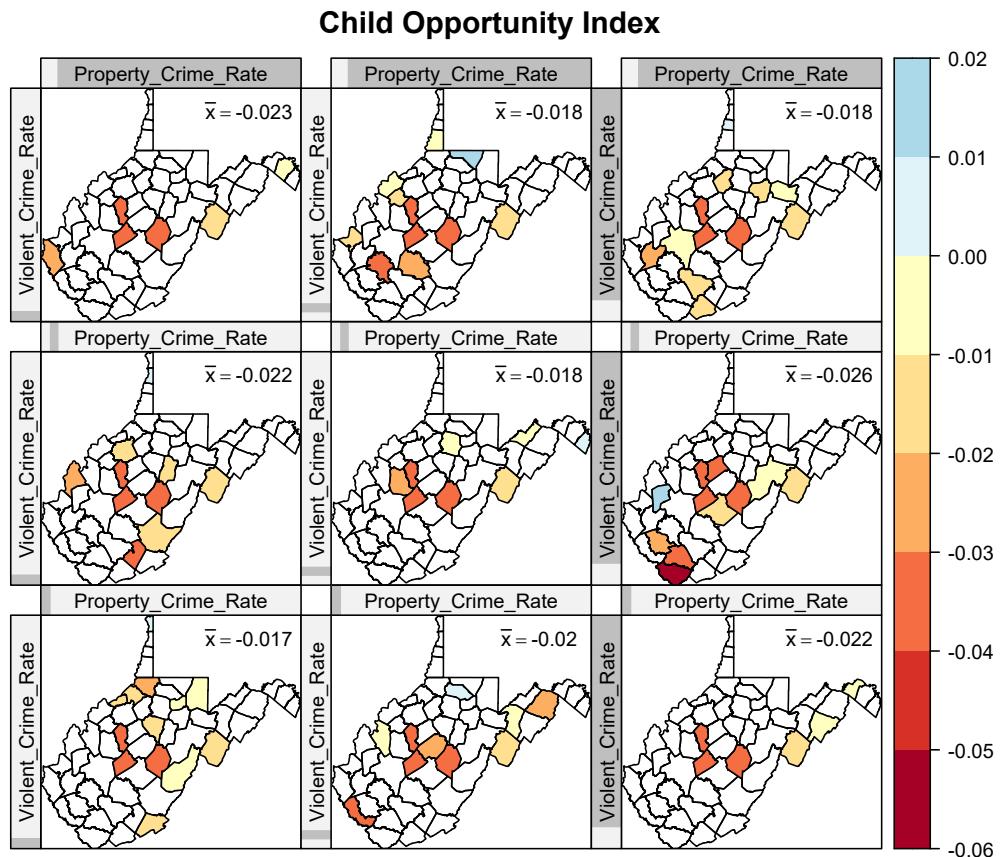


FIGURE 10.8 Modified conditioned choropleth map 2.

```

## [1] "FALSE" "TRUE"

levels(female_majority) <- c("Female_Majority_FALSE", "Female_Majority_TRUE")
levels(female_majority)

## [1] "Female_Majority_FALSE" "Female_Majority_TRUE"

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Female_Majority" = female_majority
  ),
  at = seq(-0.06, 0.02, by = 0.01),
  col.regions = brewer.pal(11, "RdYlBu")[1:8],
  main = "Child Opportunity Index",

```

```

  layout = c(2, 1),
  pstat.function = "mean",
  pstat.loc = "LR",
  pstat.digits = 4
)

```

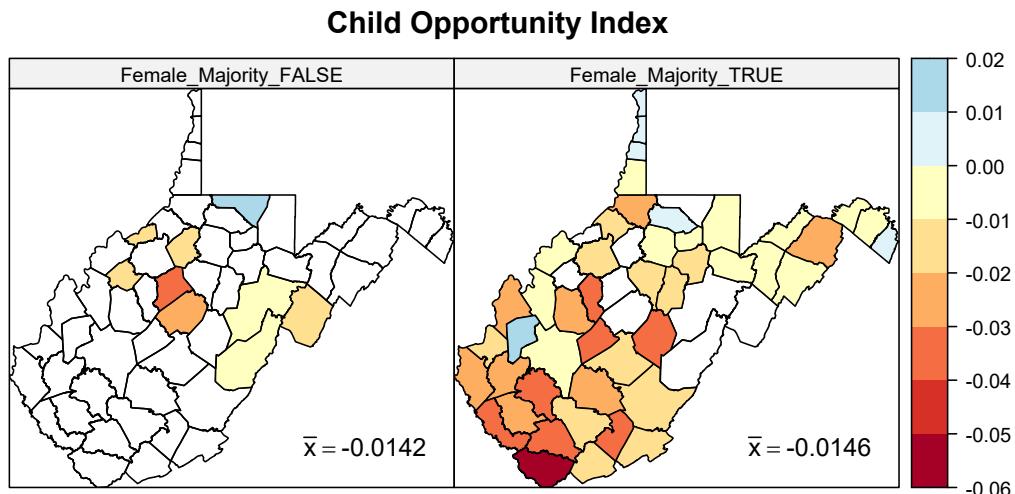


FIGURE 10.9 Conditioned choropleth map 1 for 1 categorical variable.

```

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Property_Crime_Rate" = prop_ec,
    "Female_Majority" = female_majority
  ),
  at = seq(-0.06, 0.02, by = 0.01),
  col.regions = brewer.pal(11, "RdYlBu")[1:8],
  main = "Child Opportunity Index",
  layout = c(2, 3),
  pstat.function = "mean",
  pstat.loc = "LR",
  pstat.digits = 4,
  pstat.cex = 0.8
)

```

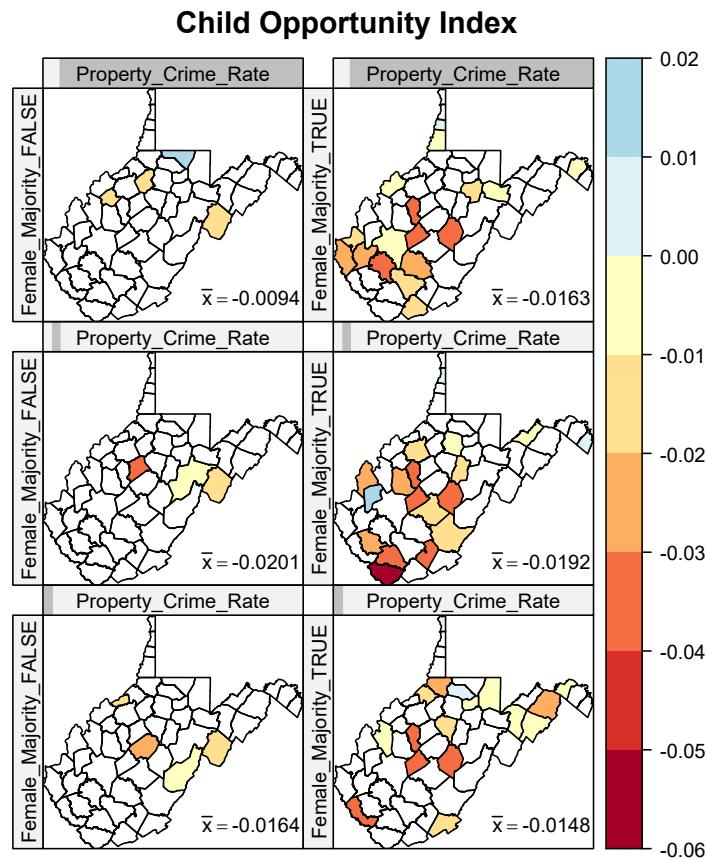


FIGURE 10.10 Conditioned choropleth map 2 for 1 categorical and 1 quantitative variable.

```
myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Female_Majority" = female_majority,
    "Property_Crime_Rate" = prop_ec
  ),
  at = seq(-0.06, 0.02, by = 0.01),
  col.regions = brewer.pal(11, "RdYlBu")[1:8],
  main = "Child Opportunity Index",
  layout = c(3, 2),
  pstat.function = "median",
  pstat.loc = "UR",
  pstat.digits = 3,
```

```
pstat.cex = 0.7
)
```

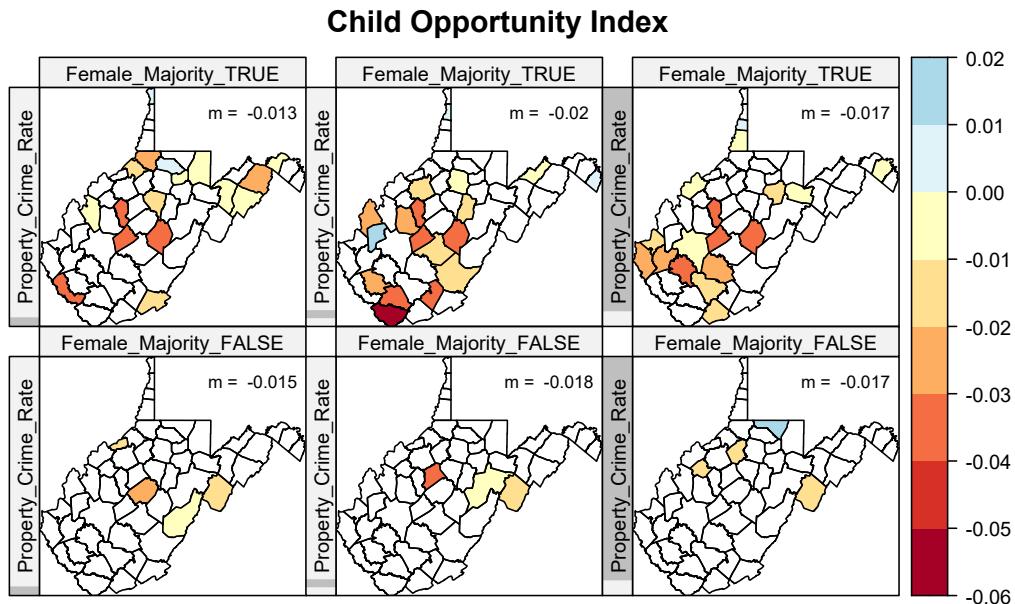


FIGURE 10.11 Conditioned choropleth map 3 for 1 quantitative and 1 categorical variable (order switched).

```
pop_growth <- as.factor(wv$POP2000 > wv$POP1986)
levels(pop_growth)

## [1] "FALSE" "TRUE"

levels(pop_growth) <- c("Population_Growth_FALSE", "Population_Growth_TRUE")
levels(pop_growth)

## [1] "Population_Growth_FALSE" "Population_Growth_TRUE"

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Female_Majority" = female_majority,
```

```

"Pop_Growth" = pop_growth
),
at = seq(-0.06, 0.02, by = 0.01),
col.regions = brewer.pal(11, "RdYlBu")[1:8],
main = "Child Opportunity Index",
layout = c(2, 2),
pstat.function = "median",
pstat.loc = "UR",
pstat.digits = 3
)

```

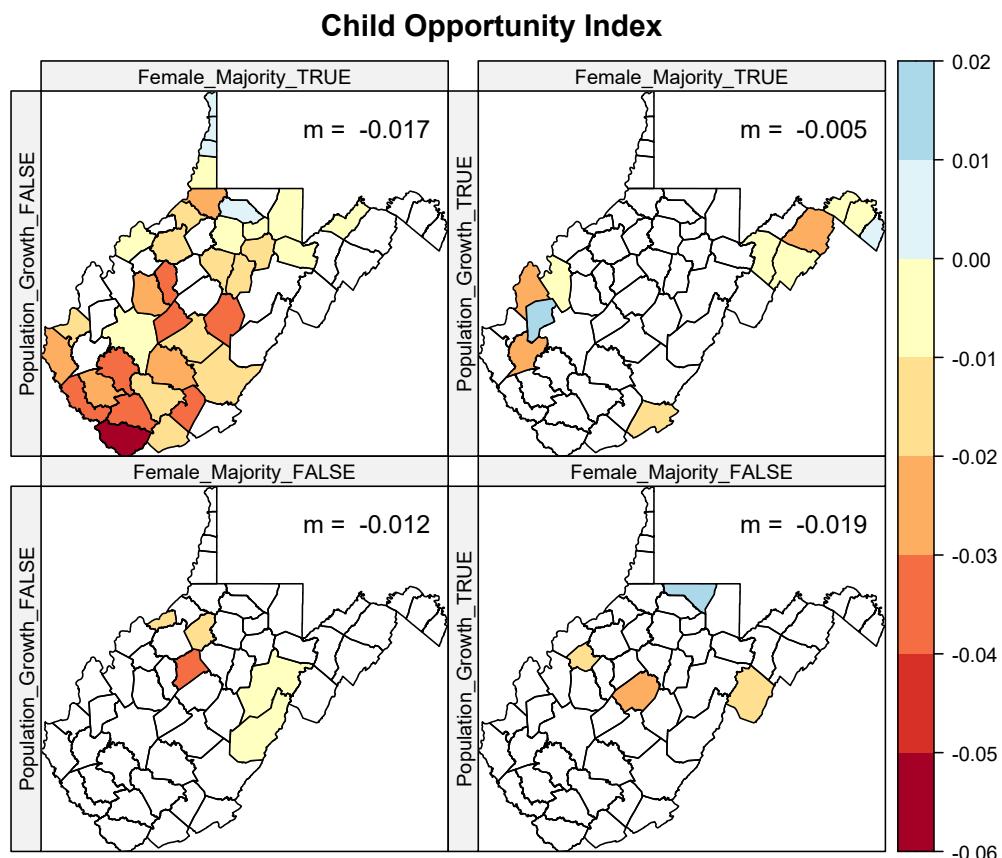


FIGURE 10.12 Conditioned choropleth map 4 for 2 categorical variables.

```

pop_growth3 <- factor(
  ifelse(wv$POP1986 - wv$POP2000 > 1000,
    "Major_Pop_Decrease",
    ifelse(wv$POP1986 - wv$POP2000 < -1000,
      "Major_Pop_Increase",
      "Minor_Pop_Change"
    )
  )
)

```

```

    )
),
levels = c("Major_Pop_Decrease", "Minor_Pop_Change", "Major_Pop_Increase")
)

levels(pop_growth3)

## [1] "Major_Pop_Decrease" "Minor_Pop_Change"   "Major_Pop_Increase"

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",
  cvar = list(
    "Female_Majority" = female_majority,
    "Pop_Growth3" = pop_growth3
  ),
  at = seq(-0.06, 0.02, by = 0.01),
  col.regions = brewer.pal(11, "RdYlBu")[1:8],
  main = "Child Opportunity Index",
  layout = c(3, 2),
  pstat.function = "mean",
  pstat.loc = "UR",
  pstat.digits = 3,
  pstat.cex = 0.9
)
}

xlim <- sp::bbox(wv)[1, ]
ylim <- sp::bbox(wv)[2, ]
textloc <- c(xlim[1] + 0.1 * (xlim[2] - xlim[1]), ylim[2] - 0.1 * (ylim[2] - ylim[1])) # UL
textloc2 <- c(xlim[2] - 0.2 * (xlim[2] - xlim[1]), ylim[1] + 0.1 * (ylim[2] - ylim[1])) # LR

sp.layout.external <- list(
  list("sp.text", textloc, "(a)", which = 4),
  list("sp.text", textloc, "(b)", which = 5),
  list("sp.text", textloc, "(c)", which = 6),
  list("sp.text", textloc, "(d)", which = 1),
  list("sp.text", textloc, "(e)", which = 2),
  list("sp.text", textloc, "(f)", which = 3),
  list("sp.text", textloc2, "WV")
)

myCCmaps(
  obj = wv,
  # Note that COI variable does not exist in local data set; use z_COI_nat instead
  # zcol = "COI",
  zcol = "z_COI_nat",

```

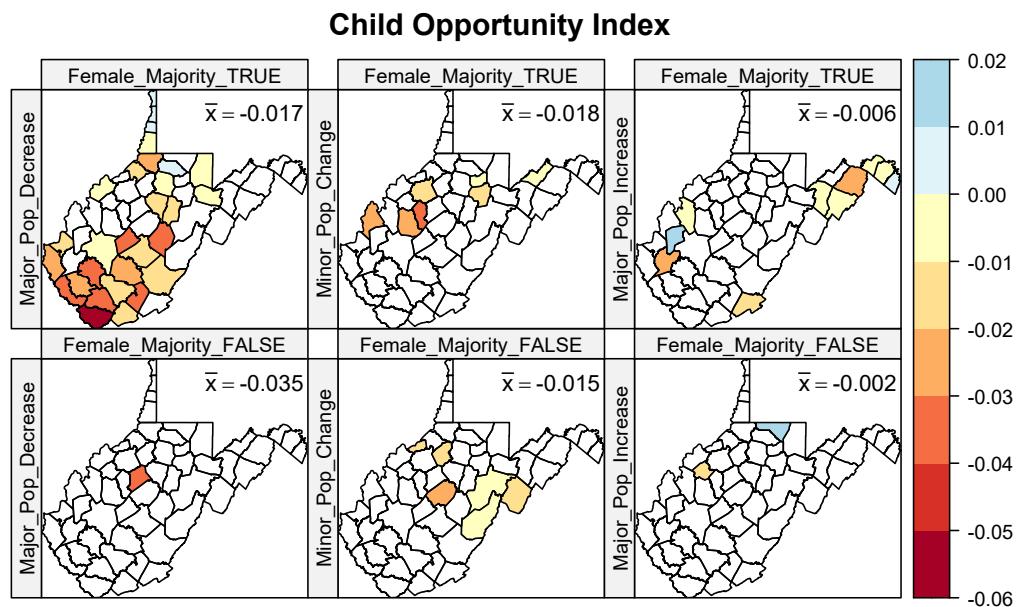


FIGURE 10.13 Conditioned choropleth map 5 for 2 categorical variables (with more than 2 factor levels).

```
cvar = list(
  "Female_Majority" = female_majority,
  "Pop_Growth3" = pop_growth3
),
at = seq(-0.06, 0.02, by = 0.01),
col.regions = brewer.pal(11, "RdYlBu")[1:8],
main = "Child Opportunity Index",
layout = c(3, 2),
sp.layout = sp.layout.external,
pstat.function = "mean",
pstat.loc = "UR",
pstat.digits = 3,
pstat.cex = 0.9
)
```

10.5 DC Examples from Brent Mast

```
# Original file: DC_Youth_Crime.R
# Original output: DC_1way_CCmap.jpeg
```

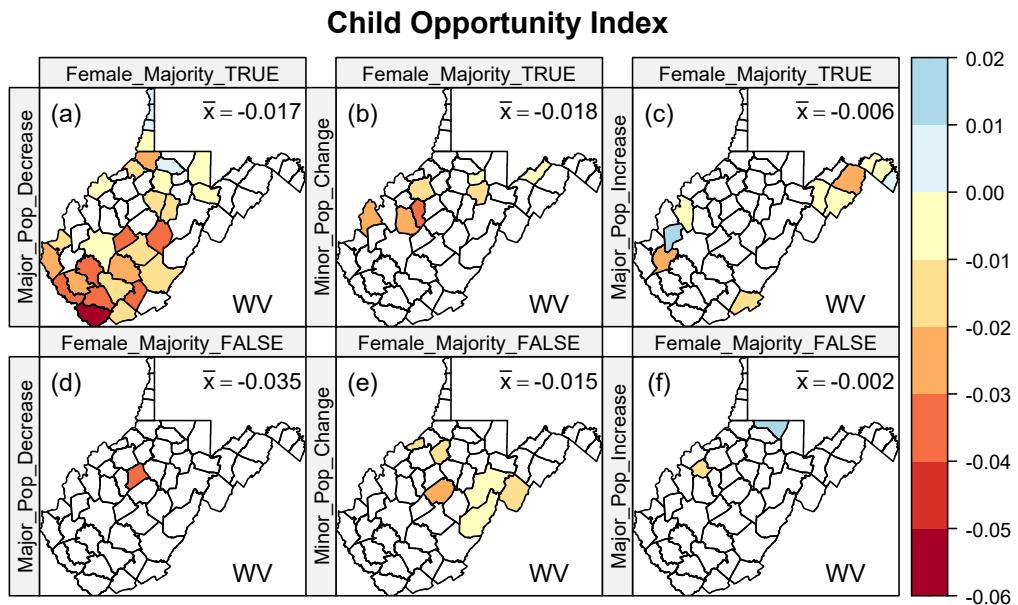


FIGURE 10.14 Conditioned choropleth map 6 for 2 categorical variables (with more than 2 factor levels). Also with externally provided text information.

```
# Original data: DC_HPN.shp

# setwd("C:\\Users\\bmast\\GD\\Chapters\\Crime")
library(lattice)
library(sf)
library(dplyr)
library(sp)

poly <- st_read("data/Ch08/DC_HPN.shp",
  crs = "+proj=longlat +ellps=clrk66"
)

## Reading layer `DC_HPN` from data source
## `D:\\Dropbox\\JUE\\Papers\\2022_MicromapBook\\MicromapRBookSource\\data\\Ch08\\DC_HPN.shp'
##   using driver `ESRI Shapefile'

## Simple feature collection with 51 features and 16 fields
## Geometry type: POLYGON
## Dimension: XY
## Bounding box: xmin: -77.12 ymin: 38.79 xmax: -76.91 ymax: 39
## Geodetic CRS: +proj=longlat +ellps=clrk66

# summary(poly)
```

```

dc <- as_Spatial(poly)
# summary(dc)
rownames(slot(dc, "data")) <- dc$GIS_ID

youth_ec <- equal.count(dc$PCT_0_17, 6, overlap = 0)
youth_ec

##
## Data:
## [1] 21.798 28.001 25.725 26.453 22.301 20.685 7.819 20.467 23.233 19.275
## [11] 20.597 11.888 9.345 15.751 15.519 20.343 16.910 20.184 10.383 9.449
## [21] 12.439 14.449 7.424 3.492 19.560 6.978 13.474 1.964 6.220 0.000
## [31] 22.375 1.385 27.294 18.790 27.076 8.908 16.650 29.720 38.697 38.294
## [41] 35.483 28.903 28.078 24.590 21.530 19.451 20.245 28.260 27.195 15.337
## [51] 19.617
##
## Intervals:
##      min     max count
## 1 -0.02859 7.847     8
## 2  9.31607 15.366     8
## 3 15.49029 19.645     9
## 4 19.58817 21.826     9
## 5 22.27274 27.224     8
## 6 27.97276 38.725     8
##
## Overlap between adjacent intervals:
## [1] 0 0 1 0 0

```

```

myCCmaps(
  obj = dc,
  zcol = "V_R_2021",
  cvar = list(
    "Percent_Youth" = youth_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(18),
  main = "Violent Crimes Per 10,000 Population",
  layout = c(3, 3)
)

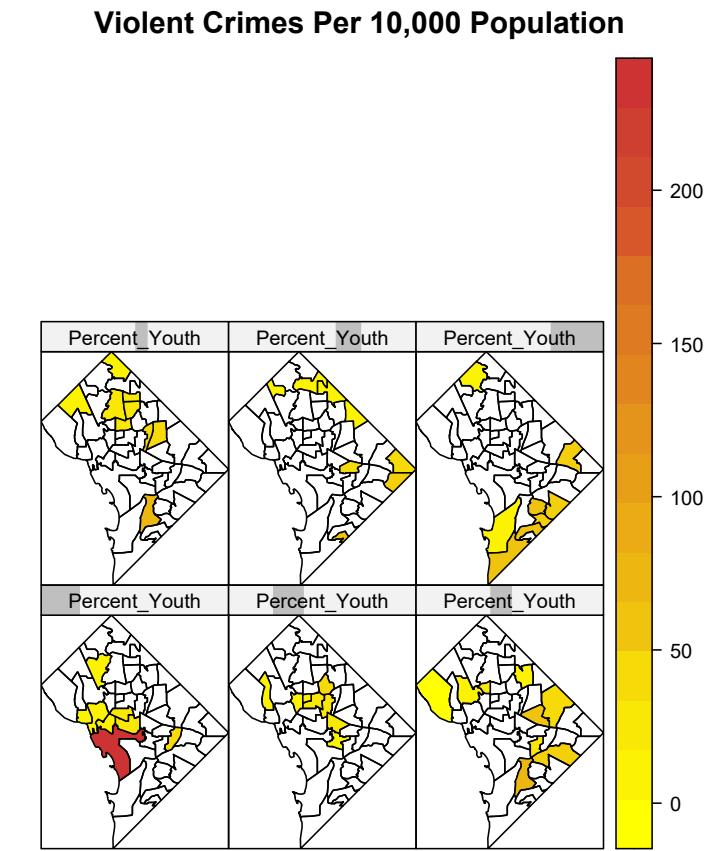
```

```

# modified

myCCmaps(
  obj = dc,
  zcol = "V_R_2021",
  cvar = list(
    "Percent_Youth" = youth_ec
  ),
  at = seq(0, 250, by = 25),

```

**FIGURE 10.15** Conditioned choropleth map 3.

```
col.regions = colorRampPalette(c("yellow1", "red"))(10),
main = "Violent Crimes Per 10,000 Population",
layout = c(3, 2),
pstat.function = "median",
pstat.loc = "UR",
pstat.cex = 0.8
)
```

```
# Original file: DC_Youth_Crime2.R
# Original output: DC_1way_CCmap2.jpeg
# Original data: DC_HPN.shp

# setwd("C:\\\\Users\\\\bmast\\\\GD\\\\Chapters\\\\Crime")
library(lattice)
library(sf)
library(dplyr)
library(sp)
```

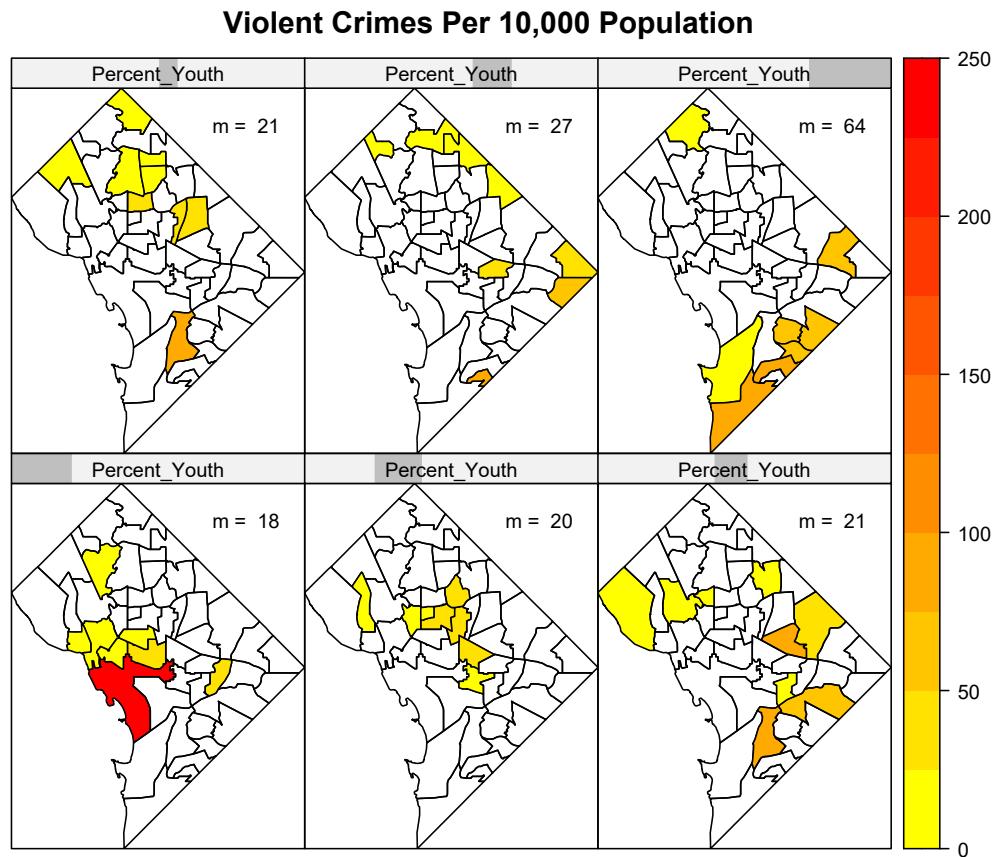


FIGURE 10.16 Modified conditioned choropleth map 3.

```

poly <- st_read("data/Ch08/DC_HPN.shp",
  crs = "+proj=longlat +ellps=clrk66"
)

## Reading layer `DC_HPN' from data source
##   'D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\DC_HPN.shp'
##     using driver `ESRI Shapefile'

## Simple feature collection with 51 features and 16 fields
## Geometry type: POLYGON
## Dimension:      XY
## Bounding box:  xmin: -77.12 ymin: 38.79 xmax: -76.91 ymax: 39
## Geodetic CRS:  +proj=longlat +ellps=clrk66

# summary(poly)

sub <- subset(poly, !(CODE %in% "N0"))

```

```
# summary(sub)

dc <- as_Spatial(sub)
# summary(dc)
rownames(slot(dc, "data")) <- dc$GIS_ID

youth_ec <- equal.count(dc$PCT_0_17, 6, overlap = 0)
youth_ec

## Data:
## [1] 21.798 28.001 25.725 26.453 22.301 20.685 7.819 20.467 23.233 19.275
## [11] 20.597 11.888 9.345 15.751 15.519 20.343 16.910 20.184 10.383 9.449
## [21] 12.439 14.449 7.424 3.492 19.560 6.978 13.474 1.964 6.220 22.375
## [31] 1.385 27.294 18.790 27.076 8.908 16.650 29.720 38.697 38.294 35.483
## [41] 28.903 28.078 24.590 21.530 19.451 20.245 28.260 27.195 15.337 19.617
##
## Intervals:
##      min     max count
## 1 1.356 8.936     8
## 2 9.316 15.547     9
## 3 15.723 19.645     8
## 4 20.156 21.826     8
## 5 22.273 27.323     9
## 6 27.973 38.725     8
##
## Overlap between adjacent intervals:
## [1] 0 0 0 0 0
```

```
myCCmaps(
  obj = dc,
  zcol = "V_R_2021",
  cvar = list(
    "Percent_Youth" = youth_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(18),
  main = "Violent Crimes Per 10,000 Population",
  layout = c(3, 3)
)
```

```
# modified

myCCmaps(
  obj = dc,
  zcol = "V_R_2021",
  cvar = list(
    "Percent_Youth" = youth_ec
  ),
```

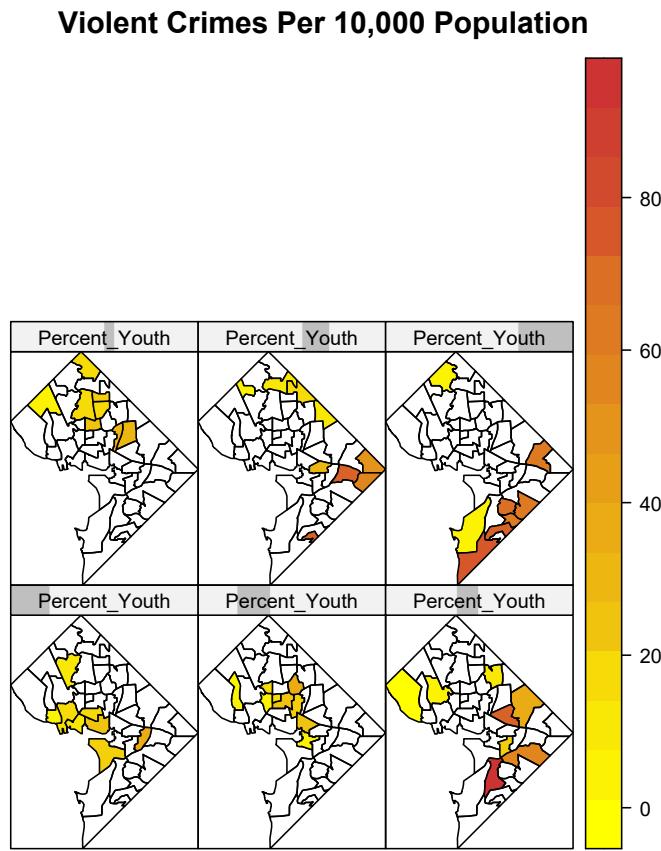


FIGURE 10.17 Conditioned choropleth map 8.

```

at = seq(0, 100, by = 10),
col.regions = colorRampPalette(c("yellow1", "brown3"))(10),
main = "Violent Crimes Per 10,000 Population",
layout = c(2, 3),
pstat.function = "median",
pstat.loc = "UR",
pstat.cex = 0.6
)

```

10.6 NJ Examples from Brent Mast

```

# Original file: NJ_Counties2.R
# Original output: NJ_One_Way_CCMap.jpeg

```

Violent Crimes Per 10,000 Population

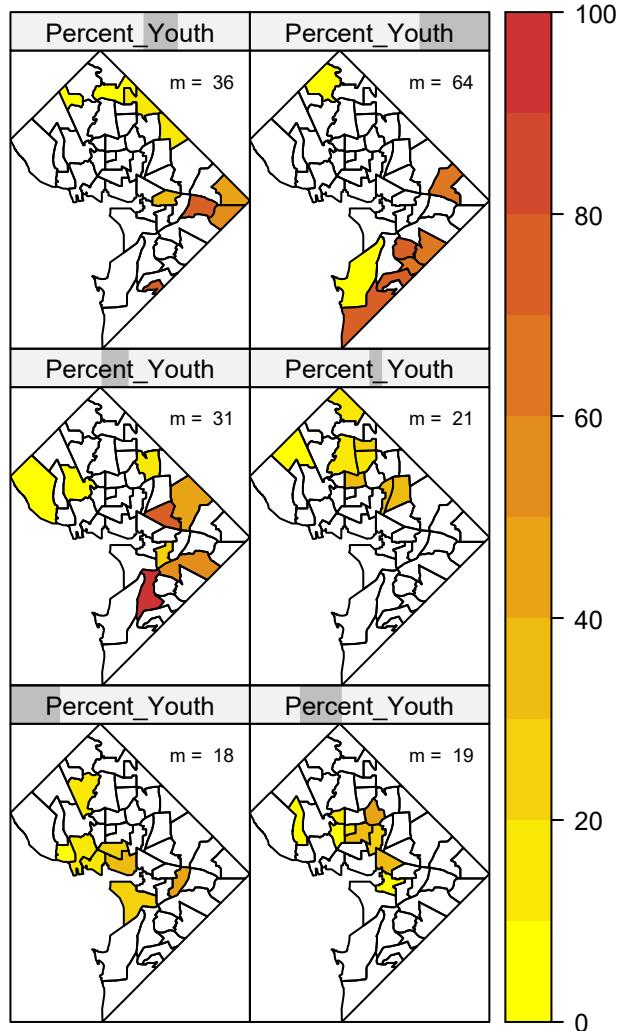


FIGURE 10.18 Modified conditioned choropleth map 8.

```
# Original data: nj_ctny_Disab_gaps.csv

# setwd("C:\\Users\\bmast\\GD\\NJ_Counties")
library(stringr)
library(tidyr)
library(sf)
library(dplyr)
library(cdlTools)
library(lattice)
library(sf)
```

```

library(sp)

# Note that tl_2024_us_county.shp is > 100MB and can't be stored on GitHub.
# Moreover, these are the political boundaries of the states.
# Work with smaller state boundary file that contains the water boundaries.
#
# poly <- st_read("data/Ch08/tl_2024_us_county.shp",
#   crs = "+proj=longlat +ellps=clrk66"
# )
# summary(poly)

# nj <- subset(poly, STATEFP == "34")
# head(nj$GEOID)

nj <- st_read("data/Ch08/NJ_Counties_3424.shp")

## Reading layer `NJ_Counties_3424` from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\NJ_Counties_3424.shp'
## using driver `ESRI Shapefile'

## Simple feature collection with 21 features and 20 fields
## Geometry type: MULTIPOLYGON
## Dimension: XY
## Bounding box: xmin: 193700 ymin: 34950 xmax: 657100 ymax: 919600
## Projected CRS: NAD83 / New Jersey (ftUS)

head(nj$COUNTY)

## [1] "ATLANTIC"    "BERGEN"       "BURLINGTON"  "CAMDEN"      "CAPE MAY"
## [6] "CUMBERLAND"

head(nj$FIPSSTCO)

## [1] "34001" "34003" "34005" "34007" "34009" "34011"

data <- read.csv("data/Ch08/nj_ctny_disab_gaps.csv")
# summary(data)

# data$GEOID <- as.character(data$FIPS_CNTY_CD)
data$FIPSSTCO <- as.character(data$FIPS_CNTY_CD)

data$abs_gap_read <- abs(data$gap_read)
data$abs_gap_math <- abs(data$gap_math)
# summary(data)

# poly2 <- inner_join(nj, data, join_by(GEOID))
poly2 <- inner_join(nj, data, join_by(FIPSSTCO))
# summary(poly2)

```

```

nj2 <- as_Spatial(poly2)
# rownames(slot(nj2, "data")) <- nj2$GEOID
rownames(slot(nj2, "data")) <- nj2$FIPSSTCO
# save(nj2,file="nj2.Rdata")

reading_gap_ec <- equal.count(nj2$abs_gap_read, 6, overlap = 0)
reading_gap_ec

##
## Data:
## [1] 28.33 34.01 30.98 31.82 25.22 19.47 38.50 23.06 33.30 37.25 36.95 38.92
## [13] 28.91 31.45 24.36 35.35 32.80 32.53 26.51 32.79 39.36
##
## Intervals:
##      min   max count
## 1 19.47 25.23     4
## 2 25.22 28.91     4
## 3 30.97 31.83     3
## 4 32.79 33.30     3
## 5 34.01 37.25     4
## 6 37.24 39.37     4
##
## Overlap between adjacent intervals:
## [1] 1 0 0 0 1

```

```

myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3))(18),
  main = "Math Achievement Gap",
  layout = c(3, 3)
)

```

```

# modified

myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec
  ),
  at = seq(10, 35, by = 5),
  col.regions = brewer.pal(5, "YlOrBr"),
  main = "Math Achievement Gap",
  layout = c(3, 2),

```

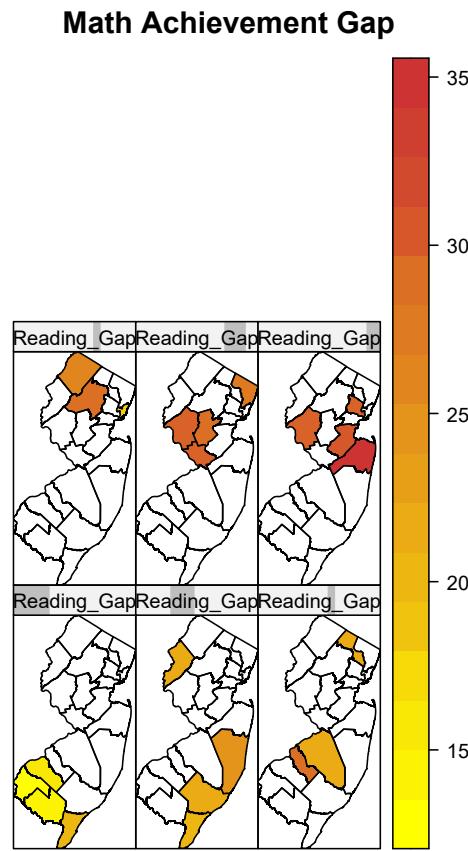


FIGURE 10.19 Conditioned choropleth map 4.

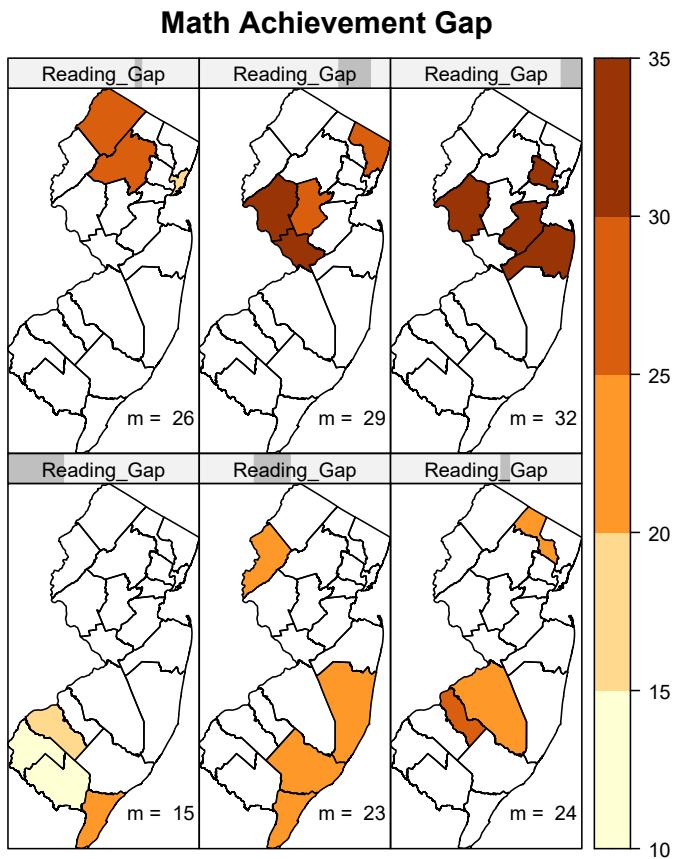
```

pstat.function = "median",
pstat.loc = "LR",
pstat.cex = 0.8
)

# Original file: NJ_Counties3.R
# Original output: NJ_One_Way_CCMap.jpeg
# Original data: No new data file

# setwd("C:\\\\Users\\\\bmast\\\\GD\\\\NJ_Counties")
library(stringr)
library(tidyr)
library(sf)
library(dplyr)
library(cdlTools)
library(lattice)
library(sf)

```

**FIGURE 10.20** Modified conditioned choropleth map 4.

```

library(sp)

# Note that tl_2024_us_county.shp is > 100MB and can't be stored on GitHub.
# Moreover, these are the political boundaries of the states.
# Work with smaller state boundary file that contains the water boundaries.
#
# poly <- st_read("data/Ch08/tl_2024_us_county.shp",
#   crs = "+proj=longlat +ellps=clrk66"
# )
# summary(poly)

# nj <- subset(poly, STATEFP == "34")
# head(nj$GEOID)

nj <- st_read("data/Ch08/NJ_Counties_3424.shp")

## Reading layer `NJ_Counties_3424` from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\NJ_Counties_3424.shp'

```

```

##   using driver 'ESRI Shapefile'
## Simple feature collection with 21 features and 20 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 193700 ymin: 34950 xmax: 657100 ymax: 919600
## Projected CRS: NAD83 / New Jersey (ftUS)

head(nj$COUNTY)

## [1] "ATLANTIC"    "BERGEN"       "BURLINGTON"  "CAMDEN"       "CAPE MAY"
## [6] "CUMBERLAND"

head(nj$FIPSSTCO)

## [1] "34001" "34003" "34005" "34007" "34009" "34011"

data <- read.csv("data/Ch08/nj_ctny_disab_gaps.csv")
# summary(data)

# data$GEOID <- as.character(data$FIPS_CNTY_CD)
data$FIPSSTCO <- as.character(data$FIPS_CNTY_CD)

data$abs_gap_read <- abs(data$gap_read)
data$abs_gap_math <- abs(data$gap_math)
# summary(data)

# poly2 <- inner_join(nj, data, join_by(GEOID))
poly2 <- inner_join(nj, data, join_by(FIPSSTCO))
# summary(poly2)

nj2 <- as_Spatial(poly2)
# rownames(slot(nj2, "data")) <- nj2$GEOID
rownames(slot(nj2, "data")) <- nj2$FIPSSTCO
# save(nj2, file="nj2.Rdata")

reading_gap_ec <- equal.count(nj2$abs_gap_read, 6, overlap = 0)
reading_gap_ec

## 
## Data:
## [1] 28.33 34.01 30.98 31.82 25.22 19.47 38.50 23.06 33.30 37.25 36.95 38.92
## [13] 28.91 31.45 24.36 35.35 32.80 32.53 26.51 32.79 39.36
## 
## Intervals:
##   min   max count
## 1 19.47 25.23     4
## 2 25.22 28.91     4
## 3 30.97 31.83     3
## 4 32.79 33.30     3

```

```

## 5 34.01 37.25      4
## 6 37.24 39.37      4
##
## Overlap between adjacent intervals:
## [1] 1 0 0 0 1

```

```

myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(18),
  main = "Math Achievement Gap",
  layout = c(3, 3)
)

```

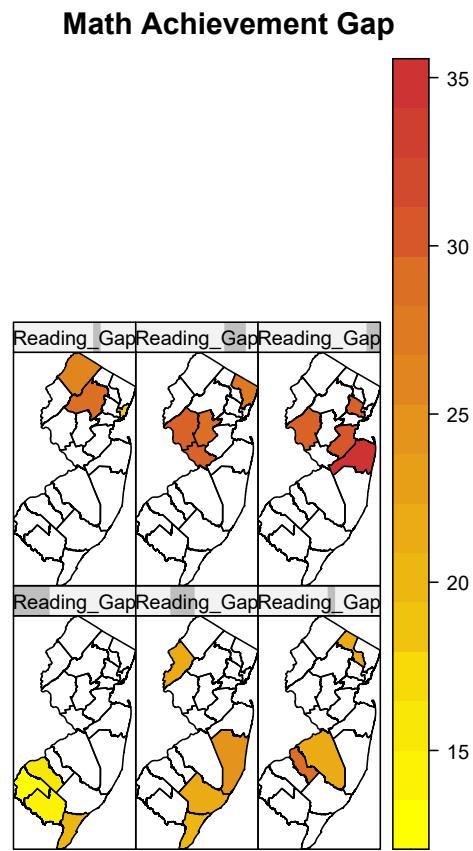


FIGURE 10.21 Conditioned choropleth map 5.

```
# modified

myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec
  ),
  at = seq(10, 35, by = 2.5),
  col.regions = brewer.pal(10, "BrBG"),
  main = "Math Achievement Gap",
  layout = c(3, 2),
  pstat.function = "median",
  pstat.loc = "LR",
  pstat.cex = 0.6
)
```

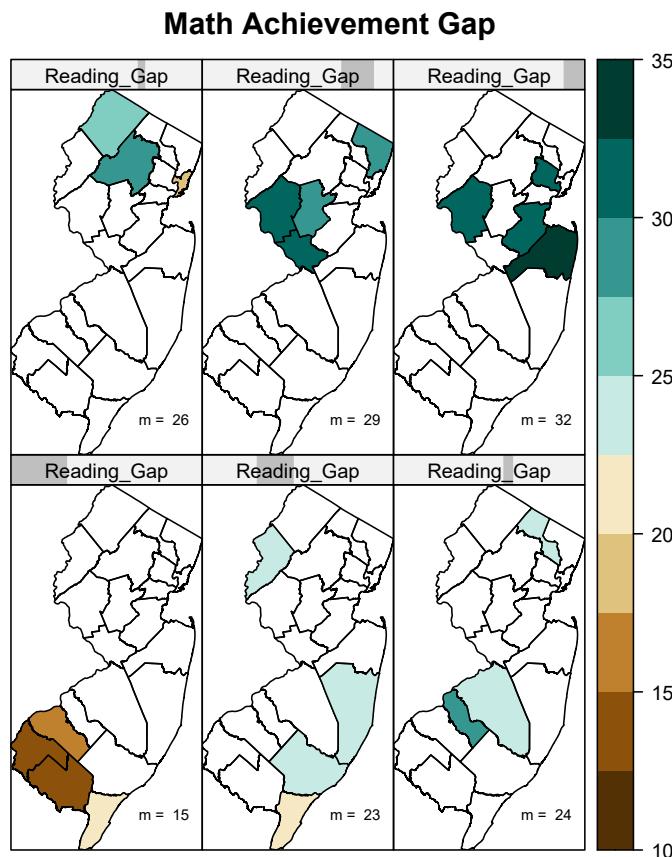


FIGURE 10.22 Modified conditioned choropleth map 5.

```

# Original file: NJ_Counties4.R
# Original output: NJ_Two_Way_CCMMap.jpeg
# Original data: nj_ctny_disab_gaps.csv & nj_mi2.Rdata

# setwd("C:\\Users\\bmast\\GD\\NJ_Counties")
library(stringr)
library(tidyr)
library(sf)
library(dplyr)
library(cdlTools)
library(lattice)
library(sf)
library(sp)

# poly <- st_read("tl_2024_us_county.shp",
#   crs = "+proj=longlat +ellps=clrk66"
# )
# summary(poly)
#
# nj <- subset(poly, STATEFP=="34")
# 
# head(nj$GEOID)

nj <- st_read("data/Ch08/NJ_Counties_3424.shp")

## Reading layer `NJ_Counties_3424` from data source
## `D:\Dropbox\JUE\Papers\2022_MicromapBook\MicromapRBookSource\data\Ch08\NJ_Counties_3424.shp'
##   using driver `ESRI Shapefile'

## Simple feature collection with 21 features and 20 fields
## Geometry type: MULTIPOLYGON
## Dimension:      XY
## Bounding box:  xmin: 193700 ymin: 34950 xmax: 657100 ymax: 919600
## Projected CRS: NAD83 / New Jersey (ftUS)

head(nj$COUNTY)

## [1] "ATLANTIC"    "BERGEN"       "BURLINGTON"  "CAMDEN"       "CAPE MAY"
## [6] "CUMBERLAND"

head(nj$FIPSSTCO)

## [1] "34001" "34003" "34005" "34007" "34009" "34011"

data <- read.csv("data/Ch08/nj_ctny_disab_gaps.csv")
summary(data)

##   FIPS_CNTY_CD      gap_read      gap_math
##   Min. :34001   Min. :-39.4   Min. :-34.1

```

```

## 1st Qu.:34011  1st Qu.:-35.3  1st Qu.:-28.7
## Median :34021  Median :-32.5   Median :-24.7
## Mean   :34021  Mean  :-31.5   Mean  :-24.9
## 3rd Qu.:34031  3rd Qu.:-28.3  3rd Qu.:-22.9
## Max.   :34041  Max.  :-19.5   Max.  :-13.5

# data$GEOID <- as.character(data$FIPS_CNTY_CD)
data$FIPSSTCO <- as.character(data$FIPS_CNTY_CD)

data$abs_gap_read <- abs(data$gap_read)
data$abs_gap_math <- abs(data$gap_math)
summary(data)

##   FIPS_CNTY_CD      gap_read      gap_math      FIPSSTCO
## Min.   :34001   Min.   :-39.4   Min.   :-34.1   Length:21
## 1st Qu.:34011  1st Qu.:-35.3  1st Qu.:-28.7  Class  :character
## Median :34021  Median :-32.5   Median :-24.7   Mode   :character
## Mean   :34021  Mean  :-31.5   Mean  :-24.9
## 3rd Qu.:34031  3rd Qu.:-28.3  3rd Qu.:-22.9
## Max.   :34041  Max.  :-19.5   Max.  :-13.5
## abs_gap_read abs_gap_math
## Min.   :19.5    Min.   :13.5
## 1st Qu.:28.3   1st Qu.:22.9
## Median :32.5    Median :24.7
## Mean   :31.5    Mean  :24.9
## 3rd Qu.:35.3   3rd Qu.:28.7
## Max.   :39.4    Max.  :34.1

load("data/Ch08/nj_mi2.Rdata")
# nj_mi2$GEOID <- nj_mi2$GEO_ID
nj_mi2$FIPSSTCO <- nj_mi2$GEO_ID

# data2 <- inner_join(data, nj_mi2, join_by(GEOID))
data2 <- inner_join(data, nj_mi2, join_by(FIPSSTCO))
summary(data2)

##   FIPS_CNTY_CD      gap_read      gap_math      FIPSSTCO
## Min.   :34001   Min.   :-39.4   Min.   :-34.1   Length:21
## 1st Qu.:34011  1st Qu.:-35.3  1st Qu.:-28.7  Class  :character
## Median :34021  Median :-32.5   Median :-24.7   Mode   :character
## Mean   :34021  Mean  :-31.5   Mean  :-24.9
## 3rd Qu.:34031  3rd Qu.:-28.3  3rd Qu.:-22.9
## Max.   :34041  Max.  :-19.5   Max.  :-13.5
## abs_gap_read abs_gap_math      GEO_ID          NAME
## Min.   :19.5    Min.   :13.5   Length:21        Length:21
## 1st Qu.:28.3   1st Qu.:22.9   Class  :character  Class  :character
## Median :32.5    Median :24.7   Mode   :character  Mode   :character
## Mean   :31.5    Mean  :24.9
## 3rd Qu.:35.3   3rd Qu.:28.7
## Max.   :39.4    Max.  :34.1

```

```

##      mi
## Min.   : 65792
## 1st Qu.: 86760
## Median : 99697
## Mean    :101528
## 3rd Qu.:111457
## Max.   :142413

cor(subset(data2, select = c("abs_gap_read", "abs_gap_math", "mi")))

##           abs_gap_read abs_gap_math     mi
## abs_gap_read      1.0000      0.8880 0.5172
## abs_gap_math       0.8880      1.0000 0.5385
## mi                 0.5172      0.5385 1.0000

# poly2 <- inner_join(nj, data2, join_by(GEOID))
poly2 <- inner_join(nj, data2, join_by(FIPSSTCO))
# summary(poly2)

nj2 <- as_Spatial(poly2)
# rownames(slot(nj2, "data")) <- nj2$GEOID
rownames(slot(nj2, "data")) <- nj2$FIPSSTCO
# save(nj2,file="nj2.Rdata")

reading_gap_ec <- equal.count(nj2$abs_gap_read, 3, overlap = 0)
reading_gap_ec

## 
## Data:
## [1] 28.33 34.01 30.98 31.82 25.22 19.47 38.50 23.06 33.30 37.25 36.95 38.92
## [13] 28.91 31.45 24.36 35.35 32.80 32.53 26.51 32.79 39.36
##
## Intervals:
##      min   max count
## 1 19.47 28.91     7
## 2 30.97 33.30     7
## 3 34.01 39.37     7
##
## Overlap between adjacent intervals:
## [1] 0 0

mi_ec <- equal.count(nj2$mi, 3, overlap = 0)
mi_ec

## 
## Data:
## [1] 80369 117577 103896 82562 93455 65792 82088 99697 90226 138037
## [11] 94086 121486 87669 84743 86760 142413 111457 104311 100048 137668
## [21] 107739
##

```

```

## Intervals:
##      min     max count
## 1  65616  87844      7
## 2  90050 104486      7
## 3 107563 142589      7
##
## Overlap between adjacent intervals:
## [1] 0 0

math_gap_ec <- equal.count(nj2$abs_gap_math, 3, overlap = 0)
math_gap_ec

## 
## Data:
## [1] 23.28 27.94 23.80 28.73 21.76 14.48 30.53 15.80 18.71 31.02 30.10 34.12
## [13] 24.65 23.09 13.51 28.34 25.82 23.80 22.92 28.68 32.36
##
## Intervals:
##      min     max count
## 1 13.51 23.09      7
## 2 23.28 28.34      7
## 3 28.67 34.12      7
##
## Overlap between adjacent intervals:
## [1] 0 0

# Code for "NJ_Two_Way_CCMMap.jpeg"
myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec,
    "Median_Income" = mi_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(18),
  main = "Math Achievement Gap",
  layout = c(3, 3)
)

# modified

myCCmaps(
  obj = nj2,
  zcol = "abs_gap_math",
  cvar = list(
    "Reading_Gap" = reading_gap_ec,
    "Median_Income" = mi_ec
  ),
  at = seq(10, 35, by = 5),
)

```

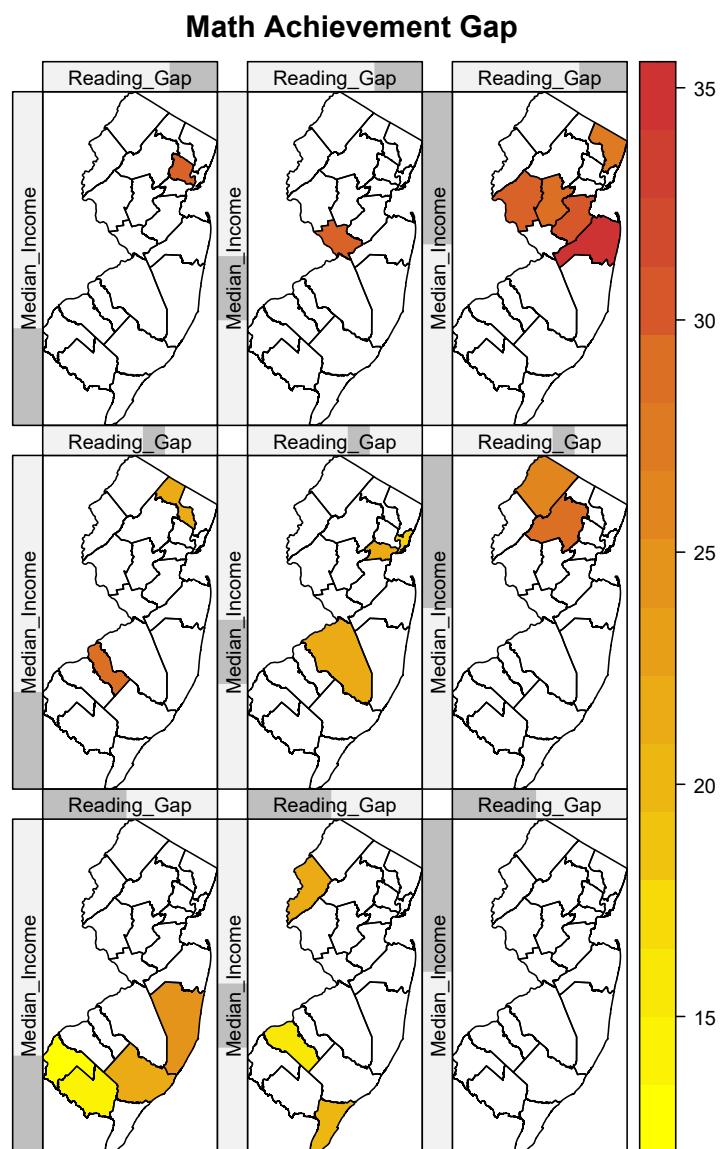


FIGURE 10.23 Conditioned choropleth map 6.

```

col.regions = brewer.pal(5, "YlOrBr"),
main = "Math Achievement Gap",
layout = c(3, 3),
pstat.function = "median",
pstat.loc = "LR",
pstat.cex = 0.8
)

```

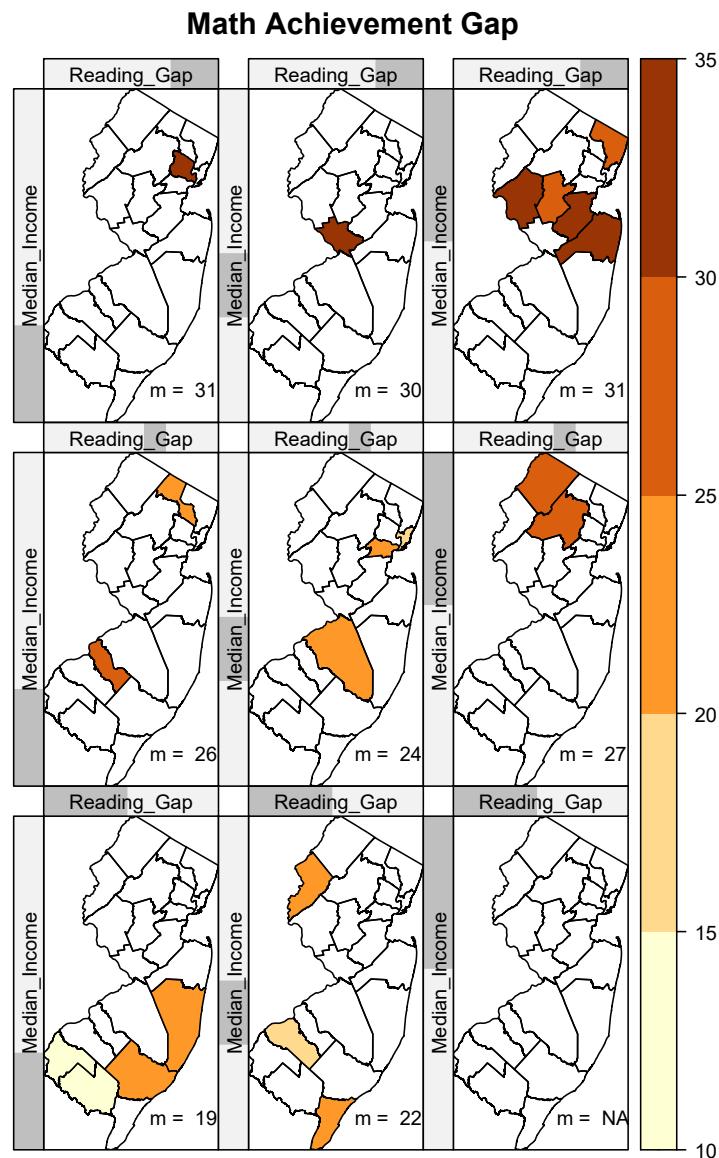


FIGURE 10.24 Modified conditioned choropleth map 6.

```
# Original file: NJ_Counties4.R
# Original output: NJ_Two_Way_CCMap2.jpeg
# Original data: No new data file

# Code for "NJ_Two_Way_CCMap2.jpeg"
myCCmaps(
  obj = nj2,
  zcol = "mi",
  cvar = list(
    "Reading_Gap" = reading_gap_ec,
    "Math_Gap" = math_gap_ec
  ),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(18),
  main = "Median Income",
  layout = c(3, 3)
)
```

```
# modified

myCCmaps(
  obj = nj2,
  zcol = "mi",
  cvar = list(
    "Reading_Gap" = reading_gap_ec,
    "Math_Gap" = math_gap_ec
  ),
  at = seq(60000, 150000, by = 10000),
  col.regions = colorRampPalette(c("yellow1", "brown3"))(9),
  main = "Median Income",
  layout = c(3, 3),
  pstat.function = "median",
  pstat.loc = "LR",
  pstat.cex = 0.6
)
```

10.7 Further Reading

Introduce cross-references to other chapters, e.g., Chapter 1 and Chapter 2, where related work and further examples can be found in this book that match the content of this chapter, that follow up on this chapter, or that are a prerequisite of this chapter.

Also, do some scientific literature review here that is specific to your chapter. Where has this R package been introduced and used before, where have other plot types or different countries been used in micromaps, what were other applications of micromaps that are related to the title and content of your chapter, etc.?

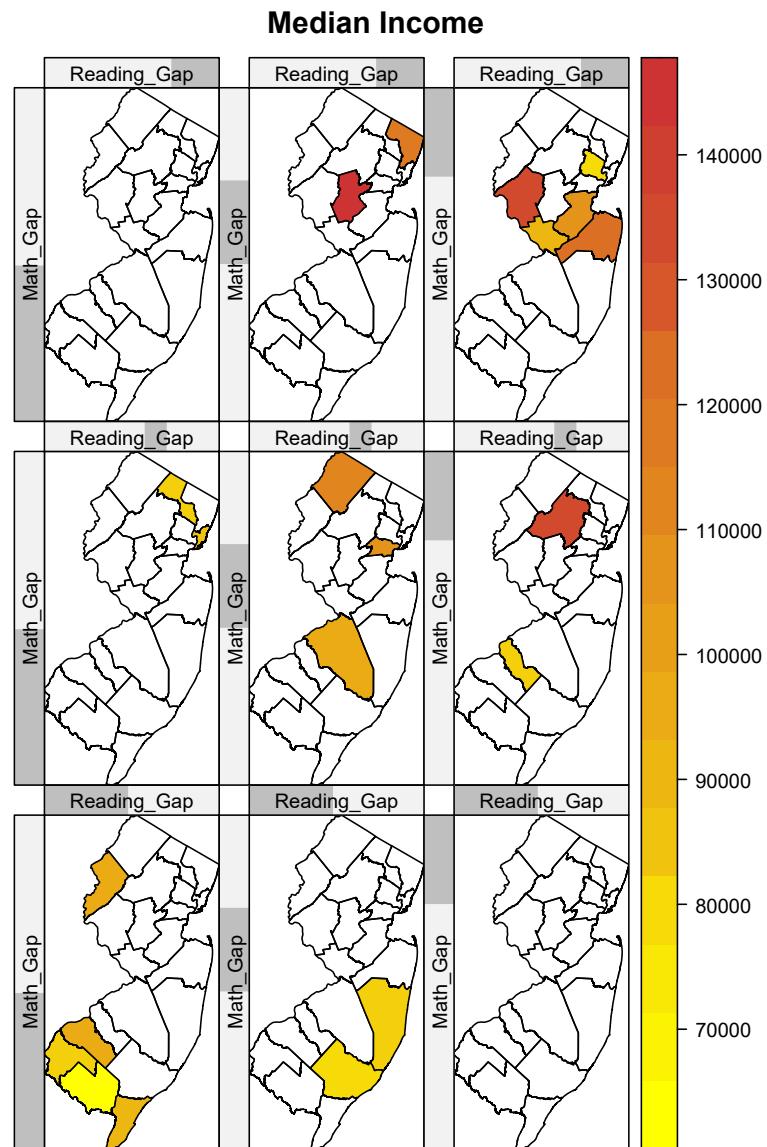


FIGURE 10.25 Conditioned choropleth map 7.

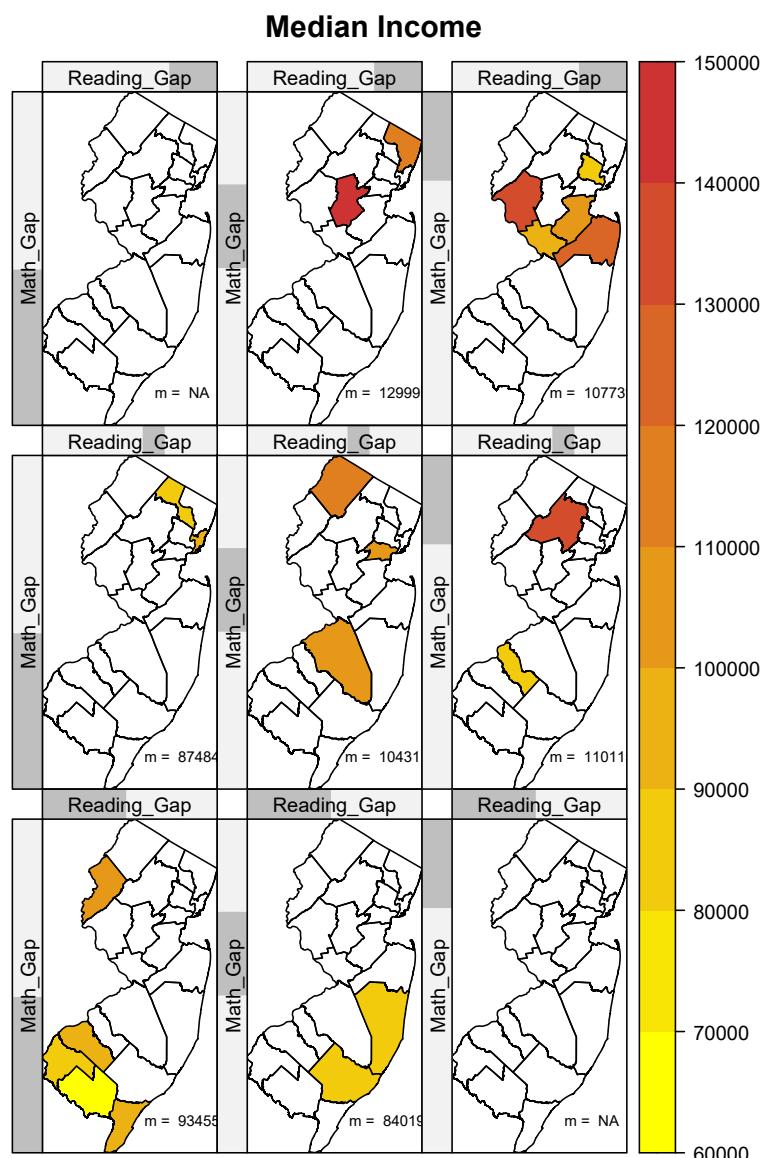


FIGURE 10.26 Modified conditioned choropleth map 7.

References

- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.

11

Comparative Micromaps

BRENT D. MAST

Comparative micromaps are the third of the three main variations of micromap visualizations. They allow a comparison of two groups in a geographic setting or the assessment of a time series of map visualizations. The main focus of these maps is to emphasize the spatial differences of the two groups or visualize the geographic changes (increases and decreases) over time, rather than leaving it to the reader to identify these changes on its own. The reader will learn how to create several variants of comparative micromaps and how to interpret them.

11.1 Introduction

As a reminder, see Chapter 1 for general style requirements for our `Micromap Plots in R` book. In particular, please do the following:

- Introduce meaningful labels for the sections, figures, and tables in your chapter.
 - Create index entries for all R packages (such as the `micromap` R package) and for all datasets (such as the `USstates` and `edPov` datasets) that are used in your chapter.
 - Include references for R packages and publications related to your chapter, such as for the `micromap` (Payton & Olsen, 2015) and `micromapST` (Carr & Pearson Jr., 2015) R packages and some micromap articles, book chapters, and books (Carr, 2001; Carr & Pickle, 2010; Symanzik & Carr, 2008).
 - Also create index entries for main topics such as linked micromap plots, conditioned choropleth maps, perceptual group, color blindness,, and quantile-quantile plot.
-

11.2 Main

Here goes the main content of your chapter. Introduce additional sections as needed.

For convenience, Figure 11.1 shows one linked micromap plot (which is the same as in Figure 1.6), but now formatted in a slightly more meaningful way.

```

library(micromap)

# initial example

data(USstates)
statePolys <- create_map_table(USstates, "ST")
data(edPov)

# basic figure 1
lmplot(
  stat.data = edPov,
  map.data = statePolys,
  panel.types = c("labels", "dot", "dot", "map"),
  panel.data = list("state", "pov", "ed", NA),
  ord.by = "pov",
  grouping = 5,
  median.row = TRUE,
  plot.width = 2,
  plot.height = 6,
  map.link = c("StateAb", "ID")
)

```

11.3 Further Reading

Introduce cross-references to other chapters, e.g., Chapter 1 and Chapter 2, where related work and further examples can be found in this book that match the content of this chapter, that follow up on this chapter, or that are a prerequisite of this chapter.

Also, do some scientific literature review here that is specific to your chapter. Where has this R package been introduced and used before, where have other plot types or different countries been used in micromaps, what were other applications of micromaps that are related to the title and content of your chapter, etc.?

References

- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<http://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].

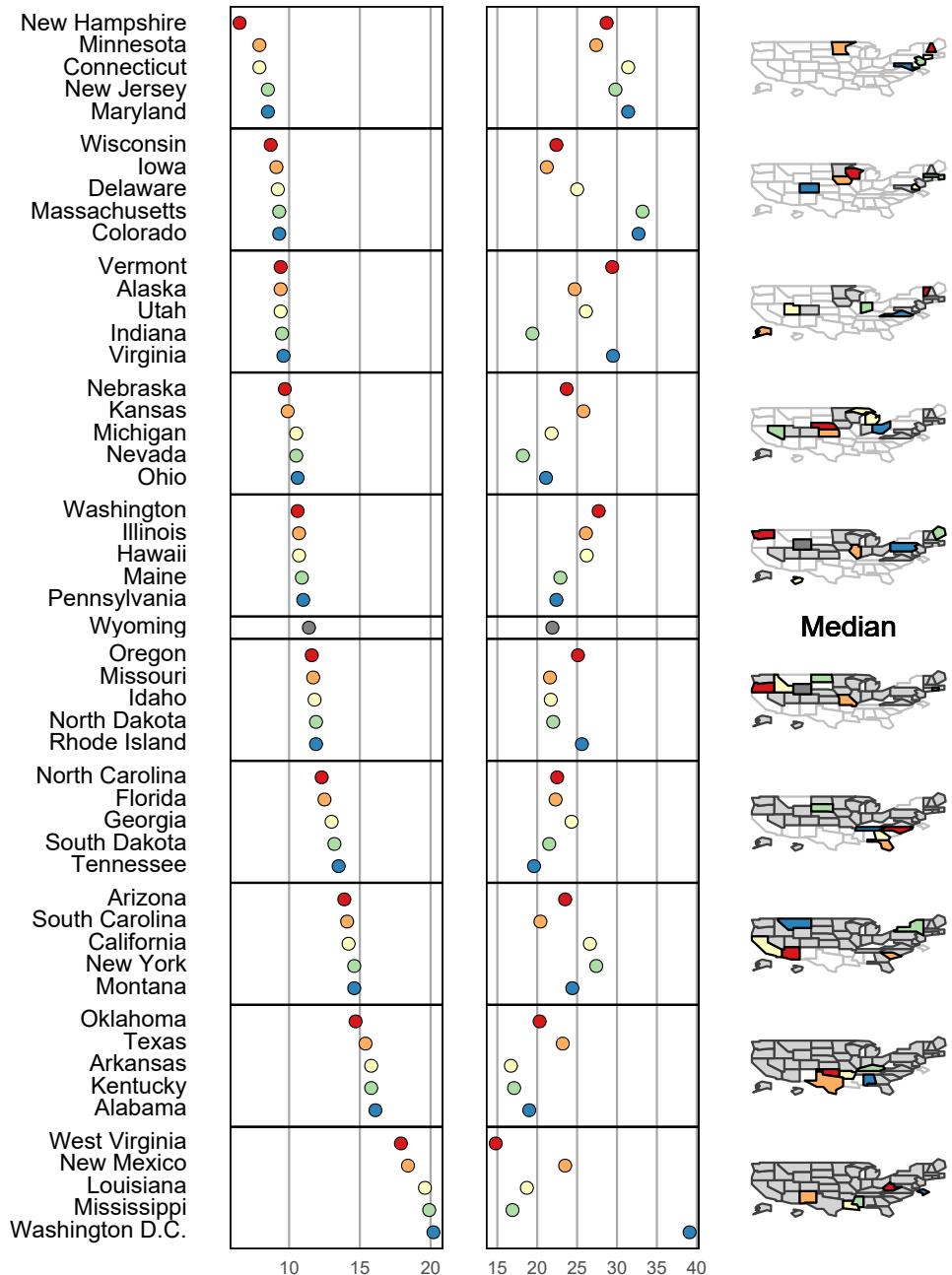


FIGURE 11.1 Here is a first micromap example for this chapter. Note that the figure is formatted in a slightly more meaningful way this time.

- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.

12

Back to the Roots: Non-traditional Micromaps (for Viewing Patterns of Scientific Posters)

CHUNYANG LI

This chapter will demonstrate how to create linked micromap plots from scratch without using any of the existing micromap R packages. This may become necessary when the underlying areas are not supported by any shapefile or other boundary file, e.g., positions on a baseball field or locations of stores in a shopping mall, or when non-standard visualizations are required, e.g., the in-flow or out-flow of a spatial area. This chapter will focus on linked microposter plots, i.e., linked micromap plots where the areas of interest are the main viewing areas of a scientific poster.

12.1 Introduction

Linked micromap plots were first introduced in 1996 to highlight geographic patterns and associations among the variables in a spatial dataset (Carr & Pierson, 1996; Olsen et al., 1996). They have been widely used to display geospatially-indexed summary statistics. For some in-depth discussion of linked micromap plots, the reader is referred to Symanzik and Carr (2008), Carr and Pickle (2010), Symanzik et al. (2014), Symanzik, Carr, McManus, et al. (2017), and additional chapters of this book, in particular Chapters 1, 2, and 3.

According to Carr and Pickle (2010), linked micromap plots can represent any two-dimensional space, not just latitude-longitude on the Earth's surface, such as positions on a baseball field or locations of stores in a shopping mall, or the in-flow or out-flow of a spatial area, etc. Based on this idea, Li and Symanzik (2016) proposed a linked microposter plot to visualize eye tracking data of how people look at scientific posters, considering a poster as a map and the areas of interest (AOIs) of the poster as equivalent to the different countries or states of a geographic map. Linked microposter plots are able to more effectively present eye tracking data compared to some of the commonly used eye tracking visualization methods. Li and Symanzik (2017) proposed **EyeTrackR**, an R package for eye tracking data visualizations including linked microposter plots, and Symanzik, Li, et al. (2017) applied this R package to create linked microposter plots to visualize how people look at posters of yoga postures.

The following sections will illustrate how to create linked micromap plots from scratch using the creation of linked microposter plots as an example. Specifically, Section 12.2 will provide an overview how to process eye tracking data for use in linked microposter plots. Section 12.3 will demonstrate how to create linked microposter plots column by column and panel by panel. Finally, Section 12.4 will provide a brief summary of this chapter and it will also

provide suggestions for further reading. The overall goal of this chapter is to enable users to change the R code from this chapter to create other types of linked micromap plots for their own needs.

12.2 Data Processing and Preparation

Section 12.2.1 will provide a brief summary of eye tracking data that is used as basis for the linked microposter plots. Section 12.2.2 will discuss areas of interest that can be seen as equivalent to shapefiles in regular linked micromap plots. Section 12.2.3 will describe feature extraction from the raw eye tracking data. Readers will have to create similar replacements of shapefiles and data files for the creation of non-traditional linked micromap plots from scratch.

12.2.1 Eye Tracking Data

Eye tracking data is obtained from measuring where people are looking at with an eye tracking device, either a mobile eye tracker or a static eye tracker. Static eye trackers are based on a desktop, hence they are often used to study eye motion on a computer screen. Mobile eye trackers are fixed on a user's head, so they are not limited within a restricted area and can be used for a variety of activities, such as playing soccer, driving, etc. The example data we use to demonstrate linked microposter plots were obtained from a 30 Hz mobile eye tracker that records 30 video frames per second. The two co-authors of Li and Symanzik (2017) looked at a series of statistical and other scientific posters. This video data was first processed with Matlab code. Eventually, x and y coordinates were created that are translations of the gaze points recorded in the video that represent spatial locations in the poster. Li (2017) described the details of processing the raw video data. Other data such as time series of pupil radius data were also obtained from the eye tracker.

The eye tracking data for the poster we use as a basis for the linked microposter plot is based on a controlled experiment where the participant was given instructions to look at the poster in a certain way.

12.2.2 Shapefile Preparation: AOI Definition

The shapefile for the linked microposter plot contains the boundaries of the areas of interest for the poster, such as the title, logos, multiple text areas, images, and tables. Even though this technically is not a shapefile, we use this term for convenience.

The rectangles that represent the AOIs in the poster have been drawn by mouse clicking two vertices using the `locator()` function in base R graphics (R Core Team, 2017), which is able to return x and y coordinates based on the mouse clicks. The AOIs have been named by the analyst. Based on the coordinates of two vertices, we created the boundaries of the AOI rectangles that will be used as the shapefile of the poster. Table 12.1 shows two of the AOIs definitions that will be used for the linked microposter plot creation.

Creators of non-traditional linked micromap plots that follow the principle of linked microposter plots from this chapter can create shapefiles based on their own needs, for example, defining different locations of the stores in a mall or spots in a baseball field. A file similar to the AOIs from Table 12.1 can be directly used to generate the linked micromap plot.

TABLE 12.1 An example of a shapefile for a linked microposter plot that shows x and y coordinates for two AOIs (here the Logo and the Title of the poster).

section	x	y
Logo	16.86	5.166
Logo	164.29	5.166
Logo	164.29	106.603
Logo	16.86	106.603
Logo	16.86	5.166
Logo	NA	NA
Title	172.40	9.223
Title	483.48	9.223
Title	483.48	102.545
Title	172.40	102.545
Title	172.40	9.223
Title	NA	NA

This file contains the names of the AOIs and the x and y coordinates of the vertices of the AOIs. The coordinates start with one vertex and end with the coordinates of the same vertex. Each AOI is separated from the next AOI by a row of NAs. According to Carr and Pickle (2010) and Payton et al. (2015), “the polygons displayed in the maps of micromaps only need to be detailed enough to convey shape and relative position of the areas and provide a spatial framework by identifying neighboring polygons”. The same holds for the AOIs in a poster. Overall, twelve AOIs were defined for the poster image and the shapefile data has been saved as `AOIName.rda`.

Figure 12.1 shows the twelve defined AOIs of the poster. The red bounding boxes outline the defined AOIs. An additional AOI, called `blank`, contains all the empty space between these main AOIs so that the entire poster is covered. The poster is used to test the data processing results and the validity of the linked microposter plot. The participant was timed to look at eight of the AOIs for about six seconds and at four of the AOIs for about two seconds. Overall, the participant looked at the poster for about $48 + 8 = 56$ seconds, resulting in a total of about $56 \cdot 30 = 1680$ video frames. Thus, there are around 1680 rows of x and y coordinates after the video processing described in Section 12.2.1.

12.2.3 Eye Tracking Data Feature Extractions

The processed video data (which is a time series of x and y coordinates) is still considered to be raw eye tracking data. To create a linked microposter plot, the data features had to be extracted based on the previously defined AOIs. Here, different features of eye tracking data have been extracted in terms of the defined AOIs, such as how long the participant has spent looking at each AOI and how many times the participant has visited each AOI, eye movement speed from one focus point to another in pixels within each AOI, and pupil radiiuses for each visit at the corresponding AOI.

The resulting datasets described in the previous sections have been provided as part of the **micromapExtra** R package to create a sample non-traditional linked micromap plot or, more specifically, a linked microposter plot: `AOIName` contains the shapefile; `poster_dataset_all` contains two lists with features extracted from the original eye movement data set. These lists are: 1. A data frame with three columns: *section* (AOI names),

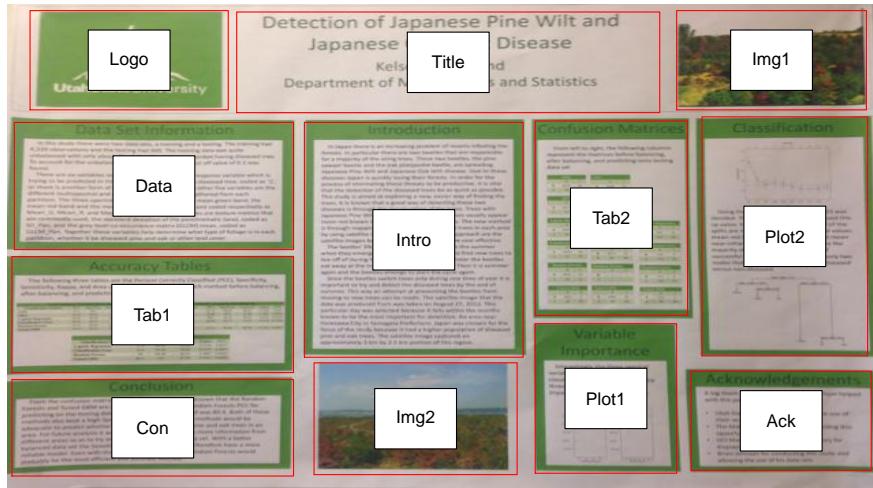


FIGURE 12.1 Poster with twelve defined AOIs.

visits_len or time the participant spent on each AOI (in seconds), and *visits_num* indicating how many times the participant has visited each AOI. 2. A list of pupil radiiuses for each AOI (each AOI is a list).

XXX JS: Simplify the data that is provided (and adjust the text above), e.g., `locations_testing` does not seem to be used later on. As far as I can see, you are also not using eye movement speed in any figure, so it can be deleted. Overall, a simple data frame with AOI name, number of visits, lengths of visits, and pupil radiiuses (and anything else that is needed) may be enough. Others should be allowed to see what is needed for the linked microposter construction and do not need to see the original data structures. XXX CL: removed the description of '`locations_testing`', also removed eye movement speed to simplify the data list

12.3 Linked Microposter Plot Creation

After the eye tracking video data has been processed and the AOI shapefile has been created as discussed in Section 12.2, we can start to create linked microposter plots following the ten steps below.

1. Panel Design Specification.

First, the user needs to figure out a few questions: What columns are there in the plot, i.e., how many and which variables does one want to visualize? What are the plot types one wants to use for the statistical columns? How to arrange the columns, such as which column type goes left and which column type goes right? How does one want to label the columns? How many perceptual groups should there be? And how many rows should be in each perceptual group? Readers who are not familiar with the general design and terminology related to linked micromap plots are encouraged to review Section 1.2.2.

A vector called `panel_types` is created to store the intended panel types in each of the columns (code section 1.1). Then, in 1.2, a list of column attributes is created to specify the order of the columns from left to right using a `col_num`. For the poster column in 1.2.1, `cumulate` is an attribute that determines if one wants cumulative patterns or not. For each column, one needs to specify the header of the column and the font characteristics of the labeling. Code section 1.2.2 does this for the label column. For the statistical graphics columns in 1.2.3-1.2.5, the data needs to be specified, such as `visits_num`, `visits_len`, and `pupil_data`. Below is the code how the linked microposter plot columns are specified. Here, we are creating three statistical graphics columns: two dotplot columns representing number of visits (1.2.3) and length of visits (1.2.4) and one boxplot column (1.2.5) representing pupil data. The poster, legend, and the statistical graphics columns are presented from left to right. After the column types and attributes have been specified, a quick check is conducted to make sure that all the column types one plans to plot have been properly specified (1.3).

```
# 1.1 Specify column types
panel_types <- c("poster", "legend", "dot", "dot", "boxplot")

# 1.2 Specify column attributes
columns_att <- list(
  # 1.2.1 Microposters column
  list(
    col_num = 1,
    cumulate = TRUE
  ),
  # 1.2.2 Label column
  list(
    col_num = 2,
    header = "Areas of Interest"
  ),
  # 1.2.3 Dotplot column to visualize number of visits
  list(
    col_num = 3,
    panel_data = "visits_num",
    header = "Number of Visits",
    axis_ticks = NA,
    axis_labels = NA
  ),
  # 1.2.4 Dotplot column to visualize length of visits
  list(
    col_num = 4,
    panel_data = "visits_len",
    header = "Length of Visits (sec)",
    axis_ticks = NA,
    axis_labels = NA
  ),
  # 1.2.5 Boxplot column to visualize pupil radius
  list(
    col_num = 5,
    header = "Pupil Radius in Pixels",
    axis_ticks = NA,
  )
)
```

```

    axis_labels = NA
  )
}

# 1.3 Check if the length of column types and column attributes match
columns <- length(columns_att)
if (length(panel_types) < columns) {
  warning("Column types are not completely specified.")
}

```

2. Load Images and Data.

After specifying the columns of the linked microposter plot, the background image/poster and data have to be loaded. The poster image is changed to grayscale. We use the **imager** R package (Barthelme, 2022) for the poster image processing such as changing the colored poster to grayscale in order to overlay other colors on top of it. The user can replace the poster with other images and replace eye tracking data with other data types.

```

library(imager)

# 2.1 Load data
load(file = "data/Ch10/posterdat_all.rda")
poster_loc <- "img/Ch10-poster_colored.jpg"
poster_dataset <- posterdat_all$poster_dataset
names(columns_att) <- panel_types

# 2.2 Load image and convert to gray scale
poster_raw <- imager::load.image(poster_loc)
poster <- imager::grayscale(poster_raw)

```

3. Load the Shapefile for the Poster.

The shapefile, i.e., the definitions of the AOIs' boundary data, is loaded in code section 3.1. The AOIs are like the states for the map of the United States and the poster is like the entire nation. The AOI `Blank` is added to the shapefile with its vertices defined as `0s` in 3.2. A polygon for the entire poster is created in 3.3 after the creation and loading of the polygons for all the AOIs.

```

# 3.1 Load shapefile or definitions of AOIs
load("data/Ch10/AOIName.rda")
aoi_vis_borders <- AOIName

# 3.2 Create shape data for blank and add it to the poster shapefile
blank <- as.data.frame(
  matrix(c(rep(0, 5 * 3), NA, NA, NA),
  nrow = 6,
  ncol = 3,
  byrow = TRUE

```

```

)
)
blank[, 1] <- "Blank"
colnames(blank) <- colnames(aoi_vis_borders)
aoi_vis_borders <- rbind(aoi_vis_borders, blank)

# 3.3 Create the polygon for the entire poster
poster_vis_borders <- matrix(
  c(
    0, 0,
    imager::width(poster), 0,
    imager::width(poster), imager::height(poster),
    0, imager::height(poster),
    0, 0
  ),
  nrow = 5,
  ncol = 2,
  byrow = TRUE
)
poster_vis_borders <- as.data.frame(poster_vis_borders)
colnames(poster_vis_borders) <- c("x", "y")

```

4. Sort Rows According to a Specific Data Column.

The user can specify how the rows of the linked microposter plot should be ordered in code section 4.1, i.e., which column to sort by. In this case, one can choose a variable from the three variables we visualize in the statistical columns: number of visits (`visits_num`), length of visits (`visits_len`) or pupil radiiuses (`pupil_data`) to sort by. The code for rearranging the datasets sorted by a specific column is shown in code section 4.2. It is executed using if statements based on the different data structures of the variables. The variable `visits_len` is chosen to sort the rows for the plot in this example.

```

# 4.1 Specify which column to sort by
sortby <- "visits_len"
decreasing <- TRUE
pupil_data <- posterdat_all$pupil_data

# 4.2 Order rows: rearrange the datasets by a specific variable
if (sortby == "pupil_data") {
  # 4.2.1 Order by pupil radius
  unordered_median <- unlist(lapply(pupil_data, stats::median))
  ordered_median <- order(unordered_median,
    decreasing = decreasing
  )
  pupil_data <- pupil_data[ordered_median]
  poster_dataset <- poster_dataset[ordered_median, ]
} else {
  # 4.2.2 Order by other variables,
  # such as number of visits and length of visits
}

```

```

ordered_rows <- order(poster_dataset[, sortby])
poster_dataset <- poster_dataset[ordered_rows, ]
pupil_data <- pupil_data[ordered_rows]
}

```

XXX JS: remove 4.2.2 from above as we are not doing anything with speed - pupil radius seems to be enough (and renumber) XXX CL: Done

5. Determine the Panel Layout.

The quickest approach to specify the panel layout for the entire linked microposter plot is to use an automatic layout function and choose a suitable partitioning (see Symanzik and Carr (2008) or Table 1.1 in Chapter 1). There are 11 such helper functions overall (See help pages of the **micromapExtra** R package for further details). There is no need to make changes to these functions, unless more changes on the layout are desired. We use the functions **panelLayout()**, **DetermineMedianRow()**, and **ArrangePanels()** to determine the panel layout. The **ArrangePanels()** function automatically creates layout, and we select Partitioning 1 (according to Symanzik and Carr (2008) or Table 1.1 in Chapter 1). The user can further specify the layout of the rows and whether a median row is wanted or not. Then the sizes of all rows are calculated based on the overall plot size and the number of rows, based on the aesthetics of the linked microposter plot.

The data is arranged into three perceptual groups. In code section 5.4, the entire panel layout frame is created. The start and end row numbers for each plot groups are automatically determined by **ArrangePanels()** function.

XXX JS: Can you mention that there are 11 functions overall. Introduce those that are needed for your code in 1 or 2 sentences. Mention that there are other ones and refer to the help pages of the **micromapExtra** R package for further details. Mention that the functions starting with panel... have been released as part of the Carr and Pickle (2010) book and are based on the original S-Plus functions (I found S-Plus code on my side from 2003, but there should be even older code). The three other functions likely have been created on my side, but I could not find when. XXX CL: updated the description based on the suggestion. I didn't mention S-Plus functions, I am not sure if it might be a little redundant for readers to read, as the purpose is to be able to create the plot. But if it helps explain the history of the code and would benefit for the big picture, feel free to add a sentence or so

```

# 5.1 Load source code for panel definitions and layout
source("R/Ch10_SharedFunctions.R")
layout <- NA
median_row <- TRUE
automatic_layout <- TRUE
partitioning <- 1

# 5.2 Determine the layout
if (automatic_layout) {
  rows <- ArrangePanels(
    posterdat = poster_dataset,
    partitioning = partitioning
}

```

```

) [[3]]

median_row <- DetermineMedianRow(
  n = nrow(poster_dataset),
  partitioning = partitioning
)
} else {
  rows <- length(layout)
  if (length(color_scheme) < max(layout)) {
    color_scheme <- rep(
      color_scheme,
      ceiling(max(layout) / length(color_scheme)))
  }
  warning("There are too few colors provided.")
}
}

if (!automatic_layout && median_row) {
  if ((rows %% details_layout$ngroups == 0) || (layout[ceiling(length(layout) / details_layout$ngroups)] != median_row))
    median_row <- FALSE
  warning("There is no median row.")
}
}

# 5.3 Determine the plot size of the rows
if (!median_row) {
  rowsize <- rep(details_layout$pheight / rows, rows)
} else {
  rowsize <- c(
    rep(details_layout$pheight / rows, floor(rows / details_layout$ngroups)),
    details_layout$pheight / (rows * details_layout$mheight),
    rep(details_layout$pheight / rows, floor(rows / details_layout$ngroups)))
}
}

# 5.4 Determine panel layout
panel_width <- rep(details_layout$pwidth, columns)
panels <- panelLayout(
  nrow = rows,
  ncol = columns,
  leftMargin = details$left,
  rightMargin = details$right,
  topMargin = details$top,
  bottomMargin = details$bot,
  colSize = panel_width,
  rowSize = rowsize,
  colSep = rep(details_layout$pcspace, (columns + 1)), # space between panels in columns
  rowSep = rep(details_layout$prspace, rows + 1) # space between panels in rows
}

```

```

)
# 5.5 Set panel row indexes
row_indexes <- ArrangePanels(
  posterdat = poster_dataset,
  partitioning = partitioning,
  AutomaticLayout = automatic_layout,
  Layout = layout,
  MedianRow = median_row
)
row_begin <- row_indexes[[1]]
row_end <- row_indexes[[2]]
n_groups <- length(row_end)

```

XXX JS: Place 5.4 earlier on and then add 17.2, 2, and 3.2 as additional constants and work with these constant names in 5.3, rather than using the numeric values in that code section. This means you have to rearrange some of your code and description in the main text. Same with the 2.9 from code section 6.1. Ideally, have one place (i.e., list) where you define all constants for the aesthetics of the plot and then use those constants. Same with the 0.1 and 0.01 for the rowSep and colSep further below. There may be more numeric and color constants used below. But, notice that Black may not be the same as Black if one refers to a font color and the next to a line color - so define fontcolor = Black and linecolor = Black in your list above (if needed). XXX CL: moved 5.4 to section 1

XXX JS: You defined/used details and the **ArrangePanels()** function already in step 5. above. Describe what you are doing here in code sections 6.1 and 6.2. XXX CL: I got rid of step 6 and merged them into step 5

6. Plot the Microposters Column.

After all settings have been specified in the previous steps, we can now start to create the linked microposter plot one column at a time, by starting from the leftmost column, i.e., the microposters column. The R code for linked microposter plots is structured similar to the **mmpplot()** function provided in the **micromap** R package (Payton & Olsen, 2015). We introduce a series of functions (called **plot_microposters()**, **plot_labels()**, **plot_dotplot()**, and **plot_boxplot()**) in the next three steps that plot the various panels in the columns of the linked microposter plot.

The **plot_microposters()** function has been created to plot the panels in the microposters column. If the user plans to use similar data structures as for the eye tracking data in this chapter, i.e., similar data frames, column names, and image types, then there is no need to adjust this function to visualize other topics of interest. Otherwise, it is necessary to adjust the data and column names accordingly.

The microposters column with small microposters in its panels is plotted with highlighting and coloring of the AOIs according to context and user-defined labels. **panel_col** is the column number of this column that is defined with the column attributes. The rest of the arguments for this function are the attributes specified in the previous steps. First, in code sections 6.1 and 6.2, the labels used in this column are specified. These labels typically are related to above and below median data and to cumulative microposters. One can adjust the text of the labels directly inside the function. Then in code section 6.3, the microposters

with specific AOIs highlighted are created for each of the perceptual groups via a for-loop. In this example, there are three perceptual groups and no median row. For each perceptual group, we color the AOIs that are present in that perceptual group. The remaining AOIs are either colored in gray as background or colored in light yellow if they have been highlighted in the previous perceptual groups given that the cumulative option is `TRUE`. The borders of each AOI and the entire poster are plotted with thicker black lines. Explanations of the most important arguments used within the `plot_microposters()` function are given below:

- `gsubs`: the row indexes for all AOIs that are in the current perceptual group;
- `back`: the row indexes for the AOIs that are not highlighted in the current perceptual group, i.e., those in the background;
- `fore`: the complement of `back`, i.e., the row indexes for the AOIs that are highlighted in the current perceptual group, i.e., those in the foreground;
- `pen`: the index to determine which color to use from the selected color scheme `color_scheme`;
- `past`: the indexes for cumulative AOIs shown in perceptual groups that have been highlighted above the current perceptual group.

In code sections 6.3.1, 6.3.2, and 6.3.5, the panels and indexes are set. Then the microposters (6.3.3) and `back` AOIs (6.3.4) are plotted. In code sections 6.3.6 and 6.3.7, the `fore` AOIs are plotted. In 6.3.8, the microposters are finalized with coloring the cumulative AOIs, and in 6.3.9, plotting the outlines of the borders for the poster.

```
plot_microposters <- function(panel_col,
                                cumulate = TRUE,
                                poster_dataset,
                                aoi_vis_borders,
                                poster_vis_borders,
                                poster,
                                panels,
                                n_groups,
                                rows,
                                median_row,
                                row_begin,
                                row_end,
                                color_scheme) {
  # 6.1 Assign AOI ids (one per AOI)
  aoi_ids <- aoi_vis_borders$section[is.na(aoi_vis_borders$x)]

  # 6.2 Plot panel titles
  panelSelect(layout = panels, i = 1, j = panel_col)
  panelScale()
  # If there is a median row,
  # then there will be additional labels and a median panel plotted;
  # if not, it will be skipped.
  if (median_row) {
    graphics::mtext(
      text = "Above Median Visits",
      side = 3,
      line = details$line1,
      cex = details$cex
    )
  }
}
```

```

if (cumulate) {
  graphics::mtext(
    text = "Cumulative Microposters",      # change this to flexible
    side = 3,
    line = details$line2,
    cex = details$cex
  )
}
} else {
  if (cumulate) {
    graphics::mtext(
      text = "Cumulative Microposters",
      side = 3,
      line = details$line1,
      cex = details$cex
    )
  }
}
}

panelSelect(layout = panels, i = n_groups, j = panel_col)
panelScale()

if (median_row) {
  graphics::mtext(
    text = "Below Median Visits",
    side = 1,
    line = details$line3,
    cex = details$cex
  )
}

rxpath <- range(poster_vis_borders$x, na.rm = TRUE)
rypoly <- range(poster_vis_borders$y, na.rm = TRUE)

# 6.3 Plot by perceptual groups
for (i in 1:n_groups) {
  if (median_row && i == ceiling(n_groups / 2)) {
    # 6.3.1 Map not drawn, median region plotted in adjacent panel
    panelSelect(layout = panels, i = (n_groups + 1) / 2, j = panel_col)
    panelScale()
    panelFill(col = details$wgray)
    panelOutline(col = details_layout$ocol)
    graphics::text(
      x = details$text_x,
      y = details$text_y,
      labels = "Median",
      cex = details$cex
    )
  }
  next
}

```

```

}

# 6.3.2 Select panels and set indexes
panelSelect(layout = panels, i = i, j = panel_col)
panelScale(rx = rxpoly, ry = rypoly)
gsubs <- row_begin[i]:row_end[i]
median_group <- row_begin[ceiling(n_groups / 2)]
if (median_row) {
  # If there is a median row, include median row in the index
  if (i == ceiling(n_groups / 2) - 1) {
    gsubs <- c(gsubs, median_group)
  }

  if (i == ceiling(n_groups / 2) + 1) {
    gsubs <- c(gsubs, median_group)
  }
}

panel_names <- poster_dataset$section[gsubs]

# 6.3.3 Plot background images (the poster)
graphics::rasterImage(
  image = poster,
  xleft = 0,
  ybottom = 0,
  xright = imager::width(poster),
  ytop = imager::height(poster)
)

# 6.3.4 Plot background (out of contour) regions
# in gray with white outlines
# "front_regions" are the regions in the group, the rest are "back"
front_regions <- poster_dataset$section[1:row_end[i]]
back <- is.na(match(x = aoi_vis_borders$section, table = front_regions))
graphics::polygon(
  x = aoi_vis_borders$x[back],
  y = aoi_vis_borders$y[back],
  border = TRUE
)

# 6.3.5 Set the indexes: plot foreground regions
# for the panel in their special colors pens 1:5
# and others in contour regions in light yellow if cumulative is TRUE
fore <- !is.na(match(x = aoi_vis_borders$section, table = panel_names))
pen <- match(
  x = aoi_ids,
  table = panel_names,
  nomatch = 10
)[!is.na(match(x = aoi_ids, table = panel_names))]
```

```

# 6.3.6 Outline regions
# If there is a median row,
# then the median row color is defined here
if (median_row) {
  pen[which(pen == median_group)] <- length(color_scheme)
}

graphics::polygon(
  x = aoi_vis_borders$x[fore],
  y = aoi_vis_borders$y[fore],
  col = scales::alpha(colour = color_scheme[pen], alpha = details_layout$alpha),
  border = FALSE
)

# 6.3.7 Plot borders
graphics::polygon(
  x = aoi_vis_borders$x[fore],
  y = aoi_vis_borders$y[fore],
  col = details_layout$ocol,
  density = details_layout$density,
  lwd = details_layout$lwd
)

# 6.3.8 Plot and color cumulative regions if it is defined to be TRUE
if (cumulate) {
  past <- !back & !fore
  graphics::polygon(
    x = aoi_vis_borders$x[past],
    y = aoi_vis_borders$y[past],
    col = scales::alpha(colour = details$wyellow, alpha = details_layout$alpha),
    border = FALSE
  )
  graphics::polygon(
    x = aoi_vis_borders$x[past],
    y = aoi_vis_borders$y[past],
    col = details_layout$ocol,
    density = details_layout$density,
    lwd = details_layout$lwd
  )
}

# 6.3.9 Plot poster outline
graphics::polygon(
  x = poster_vis_borders$x,
  y = poster_vis_borders$y,
  col = details_layout$ocol,
  density = details_layout$density,
  lwd = details_layout$lwd
)

```

```

    }
}
```

Now we can apply the `plot_microposters()` function to create our first column in the linked microposter plot, i.e., the microposters column. The color palette is selected from the **RColorBrewer** R package (Neuwirth, 2022). In this example, five colors are used from the Paired color palette, since we have five rows at most in each perceptual group. Multiple microposters columns could be created if needed.

```

color_scheme <- rev(RColorBrewer::brewer.pal(n = 5, name = "Paired"))
if (sum(panel_types == "poster") >= 1) {
  posters <- which(panel_types == "poster")

  for (i in 1:length(posters)) {
    plot_microposters(
      panel_col = columns_att[posters[i]]$poster$col_num,
      cumulate = columns_att[posters[i]]$poster$cumulate,
      poster_dataset = poster_dataset,
      aoi_vis_borders = aoi_vis_borders,
      poster_vis_borders = poster_vis_borders,
      poster = poster,
      panels = panels,
      n_groups = n_groups,
      rows = rows,
      median_row = median_row,
      row_begin = row_begin,
      row_end = row_end,
      color_scheme = color_scheme
    )
  }
}
```

7. Plot the Color-Coded Legend and Labels Column.

After plotting the microposters column, we will plot the column with the color-coded legend and the labels with the subregion names. While these are considered to be two different columns as outlined in Section 1.2.2, they typically have been created in one step in the original R code for linked micromap plots. Similarly to the `plot_microposters()` function introduced in the previous step, a `plot_labels()` function has been created to make the plotting process cleaner and easier to reuse. This function is structured similarly to the `plot_microposters()` function. Also, the variable naming schemes are similar. First, we define the label content and aesthetics such as font and size. The panels and indexes are set in code section 7.1 and 7.2. Then we create the color-coded legend and the labels with a for-loop in the code based on the perceptual groups in code section 7.3. Within each perceptual group, the colors of the color-coded legend are set to match the corresponding AOIs by using the same color scheme and plotting order. The blank AOI is defined separately as we want it to be of white color which is not in the color scheme we chose for the AOIs (code section 7.3.3).

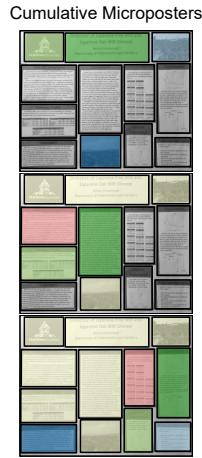


FIGURE 12.2 Microposters column of the linked microposter plot.

```
plot_labels <- function(panel_col,
                       title,
                       poster_dataset,
                       aoi_vis_borders,
                       poster_vis_borders,
                       panels,
                       n_groups,
                       median_row,
                       row_begin,
                       row_end,
                       color_scheme) {

  # 7.1 Define panels
  panelSelect(layout = panels, i = 1, j = panel_col)
  panelScale()

  # 7.2 Plot median row panel if there is a median row
  if (median_row) {
    graphics::mtext(
      text = title,
      side = 3,
      line = details$line2,
      cex = details_labels$header_size
    )
  } else {
```

```

graphics::mtext(
  text = title,
  side = 3,
  line = details$line1,
  cex = details_labels$header_size
)
}

# 7.3 Plot labels by perceptual groups
for (i in 1:n_groups) {
  # 7.3.1 Set indexes and panels
  gsubs <- row_begin[i]:row_end[i]
  gnams <- poster_dataset$section[gsubs]
  nsubs <- length(gnams)
  pen <- 1:nsubs
  laby <- nsubs:1
  panelSelect(layout = panels, i = i, j = panel_col)
  panelScale(rx = c(0, 1), ry = c(1 - details$ypad, nsubs + details$ypad))

  # 7.3.2 Define the index for the median row if there is a median row
  if (median_row && i == ((n_groups + 1) / 2)) {
    pen <- length(color_scheme)
  }

  # 7.3.3 Plot labels and color-coded legend corresponding to
  # the colors of the AOIs in the microposters
  for (j in 1:length(pen)) {
    graphics::points(
      x = details_labels$points_x,
      y = laby[j],
      pch = details_labels$points_pch1,
      col = color_scheme[pen[j]],
      cex = details_labels$dce
    )
    graphics::points(
      x = details_labels$points_x,
      y = laby[j],
      pch = details_labels$points_pch2,
      col = "black",
      cex = details_labels$dce
    )
    graphics::text(
      x = details_labels$text_x,
      y = laby[j] + details$nameShift,
      labels = gnams[j],
      cex = details_labels$cex,
      adj = 0,
      col = "black",
      font = details_labels$font
    )
  }
}

```

Now the `plot_labels()` function can be executed and the color-coded legend and labels column can be plotted.

```

if (sum(panel_types == "legend") >= 1) {
  legends <- which(panel_types == "legend")

  for (i in 1:length(legends)) {
    plot_labels(
      panel_col = columns_att[legends[i]]$legend$col_num,
      title = columns_att[legends[i]]$legend$header,
      poster_dataset = poster_dataset,
      aoi_vis_borders = aoi_vis_borders,
      poster_vis_borders = poster_vis_borders,
      panels = panels,
      n_groups = n_groups,
      median_row = median_row,
      row_begin = row_begin,
      row_end = row_end,
      color_scheme = color_scheme
    )
  }
}

```

8. Plot the Statistical Columns.

In our case, two dotplot columns and one boxplot column are used to visualize the number

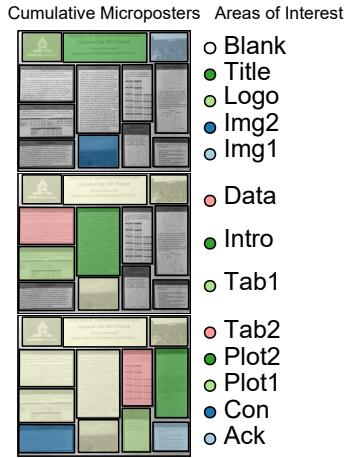


FIGURE 12.3 Similar to Figure 12.2, now with the color-coded legend and labels column added.

of visits and length of visits to the AOIs and the pupil radiiuses in pixels during these visits. Similar to the `plot_microposters()` and `plot_labels()` functions, we first set up indexes and panels, specify labels (axis, axis ticks, and texts), and then plot each dotplot or boxplot one-by-one with the corresponding colors from the AOIs within each perceptual group. Since we chose a white color for `Blank`, it is coded as a special case again.

For the `plot_dotplot()` function, the axis range as well as the panel grid are automatically determined by the range of the input data in code section A.9.1 if they are not provided by the user. For the automatic setup, axis range is set to be slightly below the minimum and above the maximum. Then the grids in the panels are determined by evenly dividing the panel width with the predefined functions. Finally, the dots are plotted within each perceptual group by using the `points()` function. Moreover, from A.9.3.1 to A.9.3.4, the indexes, panels, and labels are created for each group. In code section A.9.3.5, the dots are connected with black lines,

XXX JS: Instead of listing A.9.3.1 to A.9.3.4 at the end of the previous sentence, add this to the preceding text, similar to the description of the steps in Step 7. XXX CL: addressed

For the `plot_boxplot()` function, the data frame is specified within the data list in code section B.9.1. Then in B.9.2, the plot parameters are defined for the boxplot. In B.9.3, the summary statistics of the data for each AOI are calculated for the creation of the boxplots. Similar to the `plot_dotplot()` function, in B.9.4, the panel grid and axis labels are created, and in B.9.5, the groups panels are set up. Finally, the boxplots are created component-wise within each perceptual group by first plotting the outliers and then the boxes.

```

# A Dotplot
plot_dotplot <- function(var,
                         poster_dataset,
                         panel_num,
                         title,
                         axis_ticks = NA,
                         axis_labels = NA,
                         panels,
                         n_groups,
                         median_row,
                         row_begin,
                         row_end,
                         draw_line = TRUE,
                         color_scheme) {
  # A.9.1 Define grids based on the data to be visualized
  var <- poster_dataset[, var]
  aoi_names <- poster_dataset$section
  countRange <- range(var)
  countRange <- mean(countRange) + details_dotplot$rmulti * diff(countRange) * c(-details_dotplot$rmarks, details_dotplot$marks)
  if (is.na(axis_ticks)) {
    # If axis is not defined by the user, grids are determined automatically and
    # pretty values that are within bounds are used
    countGrid <- panelInbounds(bnds = countRange)
  } else {
    # If axis is defined by the user
    countGrid <- axis_ticks
  }
  # A.9.2 Set up panels as a big frame for the entire plot
  panelSelect(layout = panels, i = 1, j = panel_num)
  panelScale()
  graphics::mtext(
    text = title,
    side = 3,
    line = details;line1,
    cex = details$cex
  )
  # A.9.3 Plot by perceptual groups
  for (i in 1:n_groups) {
    # A.9.3.1 Set up indexes
    gsubs <- row_begin[i]:row_end[i] # group subscript
    gnams <- aoi_names[gsubs] # group names
    nsups <- length(gsubs) # number of group subscripts
    pen <- 1:nsups # index counts forward
    laby <- nsups:1 # index counts backward
    # A.9.3.2 Set up group panels
    panelSelect(layout = panels, i = i, j = panel_num)
    countRange2 <- c(
      min(countGrid) - mean(countGrid) * details_dotplot$gmulti,
      max(countGrid) + mean(countGrid) * details_dotplot$gmulti
    )
  }
}

```

```

)


panelScale(rx = countRange2, ry = c(1 - details$ypad, nsubs + details$ypad))
panelFill(col = details$wgray)
panelGrid(x = countGrid, col = "white")
panelOutline(col = "white")
# A.9.3.3 Label axis
if (i == n_groups) {
  if (is.na(axis_labels)) {
    # If axis labels are not defined by the user
    graphics::axis(
      side = 1,
      at = countGrid,
      labels = as.character(countGrid),
      col = "black",
      mgp = c(1, 0, 0),
      tck = details_dotplot$tck, #### details
      cex.axis = details$cex
    )
  } else {
    # If axis labels are defined by the user
    graphics::axis(
      side = 1,
      at = countGrid,
      labels = as.character(axis_labels),
      col = "black",
      mgp = c(1, 0, 0),
      tck = details_dotplot$tck,
      cex.axis = details$cex
    )
  }
}
# A.9.3.4 Define median row index if there is a median row
if (median_row & i == ((n_groups + 1) / 2)) {
  pen <- length(color_scheme)
}

if (draw_line == TRUE){
  graphics::lines(
    x = var[gsubs],
    y = laby,
    col = "black",
    lwd = 1
  )
}

# A.9.3.5 Plot dots using corresponding data points
for (j in 1:length(pen)) {
  graphics::points(
    x = var[gsubs[j]],
    y =
  )
}


```

```

y = laby[j],
pch = details_dotplot$pch1,
cex = details$dcex,
col = color_scheme[pen[j]]
)
graphics::points(
  x = var[gsubs[j]],
  y = laby[j],
  pch = details_dotplot$pch2,
  cex = details$dcex,
  col = "black"
)
if (gnams[j] == "Blank") {
  graphics::points(
    x = var[gsubs[j]],
    y = laby[j],
    pch = details_dotplot$pch1,
    cex = details$dcex,
    col = "white"
  )
  graphics::points(
    x = var[gsubs[j]],
    y = laby[j],
    pch = details_dotplot$pch2,
    cex = details$dcex,
    col = "black"
  )
}
}

# B. Boxplot
plot_boxplot <- function(dat,
                         poster_dataset,
                         panel_num,
                         panels,
                         title,
                         axis_ticks = NA,
                         axis_labels = NA,
                         n_groups,
                         median_row,
                         row_begin,
                         row_end,
                         color_scheme) {
  # B.9.1 Prepare data to visualize
  aoi_names <- poster_dataset$section

  # B.9.2 Set up boxplot parameters
}

```

```

# y boxplot scaling
# standard - horizontal box - no vertical (y) dimensions
py <- c(-details_boxplot$pmulti, -details_boxplot$pmulti, details_boxplot$pmulti, details_boxplot$pmulti)
thiny <- details_boxplot$thin_box * py
thicky <- details_boxplot$thick_box * py
medy <- details_boxplot$median_line * c(-details_boxplot$pmulti, details_boxplot$pmulti)
ry <- c(0, 1) # used in y scaling for grid lines

# B.9.3 Gather boxplot statistics and put in AOI order
boxlist <- graphics:::boxplot(dat, plot = FALSE)
# For the moment match on names
# Boxlist: names, stats, out, group
# Name: 1-low, 2-25%, 3-median, 4-75%, 5-high
# 5 variables for each AOI
stats <- boxlist$stats
# a column for each AOI
# - thin line - outliers - columns in boxlist (1,5,5,1)
thin <- stats[c(1, 5, 5, 1), ]
# a column for each AOI - thick line - 25% to 75% -
# columns in boxlist(2, 4, 4, 2)
thick <- stats[c(2, 4, 4, 2), ]
med <- stats[3, ] # a single value for each AOI (median)
nam <- boxlist$names # AOI names
# Define outliers
outlier <- rep(FALSE, length(med))
if (!is.null(boxlist$out)) {
  out <- boxlist$out
  group <- boxlist$group
  outlier[unique(group)] <- TRUE
}

# B.9.4 Put data in order and set up grids
ord <- match(x = aoi_names, table = nam)

if (is.null(out)) {
  rx <- range(stats)
} else {
  rx <- range(stats, out)
}
countRange <- details_boxplot$sc * diff(rx) * c(-details_boxplot$pmulti, details_boxplot$pmulti) + mean(rx)
if (is.na(axis_ticks)) {
  # If axis is not defined by the user, grids are determined automatically and
  # pretty values that are within bounds are used
  countGrid <- panelInbounds(bnds = countRange)
} else {
  # If axis is defined by the user
  countGrid <- axis_ticks
}

```

```

# B.9.5 Set up group panels
panelSelect(layout = panels, i = 1, j = panel_num)
panelScale()
graphics::mtext(
  text = title,
  side = 3,
  line = details$line1,
  cex = details_boxplot$text_size
)

# B.9.6 Plot by perceptual groups
for (i in 1:n_groups) {
  # B.9.6.1 Set up plotting index
  gsubs <- row_begin[i]:row_end[i]
  nsubs <- length(gsubs)
  gnams <- aoi_names[gsubs]
  pen <- 1:nsubs
  laby <- nsubs:1
  # B.9.6.2 Set up group panels
  panelSelect(layout = panels, i = i, j = panel_num)
  countRange2 <- c(
    min(countGrid) - stats::median(countGrid) * details_boxplot$mmulti,
    max(countGrid) + stats::median(countGrid) * details_boxplot$mmulti
  )
  panelScale(rx = countRange2, ry = c(1 - details$ypad, nsubs + details$ypad))
  panelFill(col = details$wgray)
  panelGrid(x = countGrid, col = "white")
  panelOutline(col = "white")
  # B.9.6.3 Plot axis labels
  if (i == n_groups) {
    if (is.na(axis_labels)) {
      # If axis labels are not defined by the user
      graphics::axis(
        side = 1,
        at = countGrid,
        labels = as.character(countGrid),
        col = "black",
        mgp = c(1, 0, 0),
        tck = details_boxplot$tck,
        cex.axis = details_boxplot$text_size
      )
    } else {
      # If axis labels are defined by the user
      graphics::axis(
        side = 1,
        at = countGrid,
        labels = as.character(axis_labels),
        col = "black",
        mgp = c(1, 0, 0),
        cex.axis = details_boxplot$text_size
      )
    }
  }
}

```

```

tck = details_boxplot$tck,
cex.axis = details_boxplot$text_size
)
}
}

# B.9.6.4 Define median row index if there is a median row
if (median_row && i == ((n_groups + 1) / 2)) {
  pen <- length(color_scheme)
}

for (k in 1:length(pen)) {
  # Set up index for each AOI
  m <- ord[gsubs[k]] # m is the location of the AOIs in boxlist
  if (is.na(m)) next
  kp <- pen[k] # color number
  ht <- laby[k]

  # Plot outliers
  if (outlier[m]) {
    vals <- out[group == m]
    if (gnams[k] == "Blank") {
      # Plot "Blank" AOI separately in white color
      graphics::points(
        x = vals,
        y = rep(ht, length(vals)),
        pch = details_boxplot$pch_outlier,
        col = ifelse(details_boxplot$use_black, "black", "#FFFFFF"),
        cex = details_boxplot$cex_outlier,
        lwd = details_boxplot$lwd_outlier
      )
    } else {
      # Plot other AOIs with colors in the selected color scheme
      graphics::points(
        x = vals,
        y = rep(ht, length(vals)),
        pch = details_boxplot$pch_outlier,
        col = ifelse(details_boxplot$use_black, "black", color_scheme[kp]),
        cex = details_boxplot$cex_outlier,
        lwd = details_boxplot$lwd_outlier
      )
    }
  }
}

# Plot boxplots
if (gnams[k] == "Blank") {
  # Plot "Blank" AOI separately in white color
  graphics::polygon(
    x = thin[, m],
    y = rep(ht, 4) + thiny,
    col = "#FFFFFF",

```

```

    border = "black"
)
graphics::polygon(
  x = thick[, m],
  y = rep(ht, 4) + thicky,
  col = "#FFFFFF",
  border = "black"
)
graphics::segments(
  x0 = med[m],
  y0 = ht + medy[1],
  x1 = med[m],
  y1 = ht + medy[2],
  col = "black",
  lwd = details_boxplot$lwd_median
)
} else {
# Plot other AOIs with colors in the selected color scheme
graphics::polygon(
  x = thin[, m],
  y = rep(ht, 4) + thiny,
  col = color_scheme[kp],
  border = "black"
)
graphics::polygon(
  x = thick[, m],
  y = rep(ht, 4) + thicky,
  col = color_scheme[kp],
  border = "black"
)
graphics::segments(
  x0 = med[m],
  y0 = ht + medy[1],
  x1 = med[m],
  y1 = ht + medy[2],
  col = "black",
  lwd = details_boxplot$lwd_median
)
}
}
}
}
```

XXX JS: Remove the if-statements related to speed and pupil radius - this is too specific. Instead pass on poster_dataset[[2]] or poster_dataset[[3]]. Apparently, in line 151, you specify header = "Pupil Radius in Pixels" - so only pupil radius makes sense here. XXX

There are two statistical columns that make use of dotplot visualizations: the length of time spent in each AOI and the number of times each AOI has been visited. These two columns are plotted by calling the `plot_dotplot()` function twice within the for-loop.

```
# Plot number of visits and length of visits
if (sum(panel_types == "dot") >= 1) {
  dots <- which(panel_types == "dot")

  for (i in 1:length(dots)) {
    plot_dotplot(
      var = columns_att[dots[i]]$dot$panel_data,
      poster_dataset = poster_dataset,
      panel_num = columns_att[dots[i]]$dot$col_num,
      title = columns_att[dots[i]]$dot$header,
      axis_ticks = columns_att[dots[i]]$dot$axis_ticks,
      axis_labels = columns_att[dots[i]]$dot$axis_labels,
      panels = panels,
      n_groups = n_groups,
      median_row = median_row,
      row_begin = row_begin,
      row_end = row_end,
      draw_line = FALSE,
      color_scheme = color_scheme
    )
  }
}
```

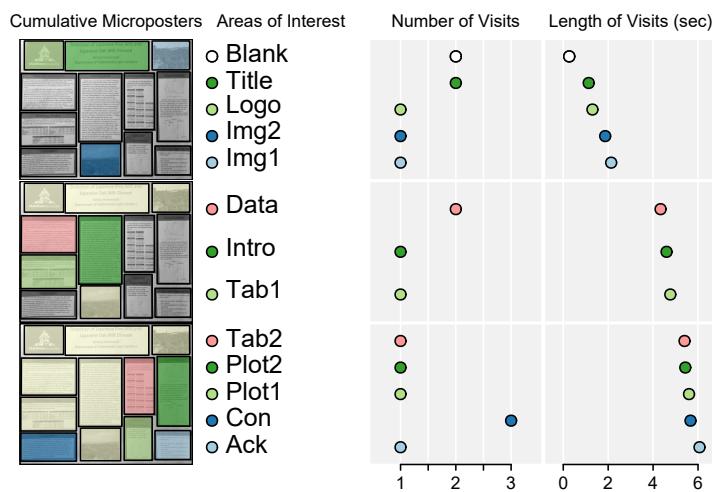


FIGURE 12.4 Similar to Figure 12.3, now with two dotplot columns added.

XXX JS: The connecting lines between the dots should be optional. Can you modify the `plot_dotplot()` function and add an argument to draw / not draw these lines. Set the default to FALSE. Here, draw the line for the number of visits column, but omit it for the

length of visits column. This line makes most sense when there are big jumps in the data as is the case for the number of visits, but it makes less sense for the column with sorted data or minor jumps only. XXX CL: addressed with the function, I removed the line for both, I personally think it looks fine

Then the third statistical column, the boxplot column to visualize pupil sizes for each AOI, is added.

```
# Plot pupil radius (in pixels)
if (sum(panel_types == "boxplot") >= 1) {
  bplot <- which(panel_types == "boxplot")

  for (i in 1:length(bplot)) {
    plot_boxplot(
      dat = pupil_data,# columns_att[bplot[i]]$boxplot$panel_data,
      poster_dataset = poster_dataset,
      panel_num = columns_att[bplot[i]]$boxplot$col_num,
      axis_ticks = columns_att[bplot[i]]$boxplot$axis_ticks,
      axis_labels = columns_att[bplot[i]]$boxplot$axis_labels,
      panels = panels,
      title = columns_att[bplot[i]]$boxplot$header,
      n_groups = n_groups,
      median_row = median_row,
      row_begin = row_begin,
      row_end = row_end,
      color_scheme = color_scheme
    )
  }
}
```

XXX JS: Something goes wrong with the coloring in the `plot_boxplot()` function. Note that `Blank` is colored in pink while `Ack` is colored in white at the bottom. As a side note, I was able to add black outlines around the boxes to make the white one better visible. That was easy. But I did not spot immediately why `Blank` and `Ack` do not get the proper colors. Is this a coding issue or was `Blank` assigned to the wrong AOI? In your dissertation in Fig 4.5, `Blank` is the AOI at the bottom. Here, it is at the top. So, a problem with the data / sorting of the data may be the issue (and not the code in `plot_boxplot()` function). XXX

10. Add the Title.

By now, all the columns in the linked microposter plot have been created. The plot can be finalized by adding the main title.

```
main_title <- "Eye Tracking: Statistics from Looking at a Scientific Poster "
title_cex <- 1.08

panelSelect(layout = panels, margin = "top")
invisible(panelScale())
graphics::text(
  x = 0.5,
```

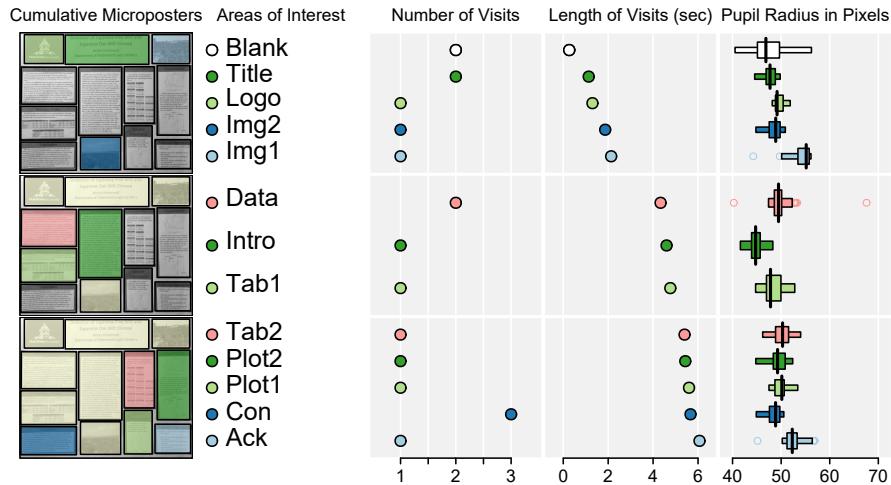


FIGURE 12.5 Similar to Figure 12.4, now with one boxplot column added.

```

y = 0.75,
labels = main_title,
cex = title_cex
)

panelSelect(layout = panels, margin = "bottom")
invisible(panelScale(inches = TRUE))

```

Now we have finished creating a non-traditional linked microposter plot with our own data, boundary (shapefile) information, and background images from a different domain compared to the other chapters in this book. When one has become familiar with the steps on how to bring in data, boundary (shapefile) information, background images or maps, and different plot types and map overlays, the above steps and functions can be incorporated into an alternative linked micromap plot function to simplify the creation of a certain non-traditional linked micromap plot.

12.4 Summary and Further Reading

Readers can find additional plot types for linked micromap plots in Chapter 6 and Chapter 7 and implement those for non-traditional linked micromap plots similar to how the `plot_dotplot()` and `plot_boxplot()` functions have been incorporated in this chapter. To understand more of the linked micromap plot principles and basics of their creation via the

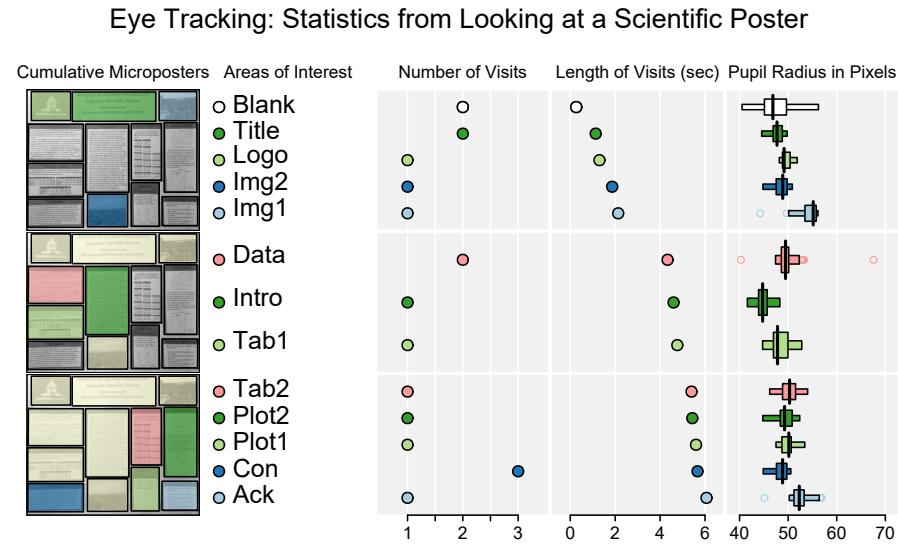


FIGURE 12.6 Similar to Figure 12.5, now with the main title added.

micromap and **micromapST** R packages, readers should refer to Chapters 1, 2, and 3, respectively.

Further, all steps demonstrated in this chapter can be found in the `DrawEyeLMPlot()` function in the **EyeTrackR** R package that is available on GitHub (<https://github.com/Chunyan-gCLi/EyeTrackR.git>). If the user wants to put overlays on the poster in a different way without using the AOI polygons, the `DrawEyeLMPPlot()` function provides a linked scanpath microposter plot (Li, 2017) without predefined AOIs. If interested in such a functionality, the user can directly refer to that R code on GitHub, while the non-traditional linked microposter plot creation approach from the **EyeTrackR** R package closely matches the R code demonstrated in this chapter.

References

- Barthelme, S. (2022). *imager: Image Processing Library Based on 'CImg'* [R package version 0.42.13 (<http://CRAN.R-project.org/package=imager>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Carr, D. B., & Pierson, S. M. (1996). Emphasizing Statistical Summaries and Showing Spatial Context with Micromaps. *Statistical Computing and Statistical Graphics Newsletter*, 7(3), 16–23.

- Li, C. (2017). *Extracting and Visualizing Data from Mobile and Static Eye Trackers in R and Matlab* (Doctoral dissertation) [<https://doi.org/10.26076/5c8c-d8a5>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Li, C., & Symanzik, J. (2016). The Linked Microposter Plot as a New Means for the Visualization of Eye Tracking Data [(CD)]. *2016 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Li, C., & Symanzik, J. (2017). EyeTrackR: An R Package for Extracting and Visualizing Data from Mobile and Static Eye Trackers [(CD)]. *2018 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Neuwirth, E. (2022). *RColorBrewer: ColorBrewer Palettes* [R package version 1.1-3 (<https://CRAN.R-project.org/package=RColorBrewer>)].
- Olsen, A. R., Carr, D. B., Courbois, J.-Y. P., & Pierson, S. M. (1996). Presentation of Data in Linked Attribute and Geographic Space. *1996 Abstracts, Joint Statistical Meetings, Chicago, Illinois* (p. 271). American Statistical Association, Alexandria, VA.
- Payton, Q. C., McManus, M. G., Weber, M. H., Olsen, A. R., & Kincaid, T. M. (2015). micromap: A Package for Linked Micromaps [<https://doi.org/10.18637/jss.v063.i02>]. *Journal of Statistical Software*, 63(2), 1–16.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- R Core Team. (2017). *R: A Language and Environment for Statistical Computing* [<http://www.R-project.org/>]. R Foundation for Statistical Computing. Vienna, Austria.
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.
- Symanzik, J., Carr, D. B., McManus, M. G., & Weber, M. H. (2017). Micromaps [<https://doi.org/10.1002/9781118445112.stat07938>]. *Wiley StatsRef: Statistics Reference Online*. Wiley Online Library.
- Symanzik, J., Dai, X., Weber, M. H., Payton, Q., & McManus, M. G. (2014). Linked Micromap Plots for South America — General Design Considerations and Specific Adjustments [<https://doi.org/10.15446/rce.v37n2spe.47949>]. *Revista Colombiana de Estadística*, 37(2), 451–469.
- Symanzik, J., Li, C., Zhang, B., Studenka, B., & McKinnney, E. (2017). Eye-Tracking in Practice: A First Analysis of a Study on Human Postures [(CD)]. *2017 JSM Proceedings*. American Statistical Association, Alexandria, VA.

13

Applications for the Environmental Field

MICHAEL G. McMANUS, MARCUS W. BECK, JÜRGEN SYMANZIK

13.1 Introduction

This chapter will give the purpose and description of linked micromaps as applied to environmental data using the R package **micromap**. We will ask what types of environmental data are most suited for summarizing and displaying as linked micromaps. Additionally, we will show how linked micromaps can be used in spatial data analysis by combining workflows with spatial sampling designs as a unique application to environmental data. Our linked micromap examples are drawn from federal and state environmental agencies in the United States.

The purpose of a linked micromap is to show spatial locations corresponding to statistical estimates (Carr, Olsen, Courbois, et al., 1998). Linked micromaps simultaneously summarize and display both statistical and geographic distributions by linking statistical summaries of polygons, or areal units, to a series of small maps (Payton et al., 2015). Implicit with this purpose and description is that space, defined as spatial contiguity of polygons, provides some context to the statistical estimates, and that the statistical estimates are representative of the polygons. This spatial context to statistical estimates can be one of three patterns: spatial similarity among neighbors, spatial dissimilarity among neighbors, or no spatial pattern among neighbors. McManus et al. (2016) illustrated that first pattern with data from the West Virginia Department of Environmental Protection (WVDEP), while the examples below by the Virginia Department of Environmental Quality (VDEQ) and Florida Department of Environmental Protection (FDEP) illustrate the second and third patterns, respectively (Silvanima et al., 2018; Virginia Department of Environmental Quality, 2020). All three of those examples also addressed the issue of obtaining statistical estimates of environmental data that were representative of watersheds (polygons) used in the analysis.

Public health statistics are often reported by administrative boundaries, such as counties, states, or countries, whereas environmental statistics are often reported by boundaries of natural features, such as watersheds, estuaries, or ecoregions. A census of environmental characteristics from such natural features is not possible. Consequently, environmental statisticians have recommended probability-based surveys or sampling to obtain representative estimates of natural resources (Peterson et al., 1999). Probability-based surveys have three steps: 1) identify the total or target population of interest (e.g. all wadeable streams in Virginia), 2) select a random sample to ensure representativeness with the known probability of including any member of the target population, and 3) apply the sample weight, the inverse of the probability, to the environmental measurements so that inferences can

be made to the target populations based on the samples (Peterson et al., 1999). Stevens Jr. and Olsen (2004) introduced spatially balanced probabilistic surveys as having sample sites distributed that correspond to the spatial distribution of the features in the target population of the natural resource and can be more efficient than a simple random sample of sites, in which clumping of sites may occur. The design and analysis of spatially balanced probabilistic surveys is done using the R package **spsurvey** (Dumelle et al., 2022; Olsen et al., 2012). An example is provided to demonstrate how **spsurvey** is used for creating samples as a precursor to collecting data and summarizing them with linked micromaps. Statistical estimates of aquatic natural resources have been made from spatially balanced probabilistic surveys of watersheds within a state to the ecoregions conterminous of the United States, as part US EPA's National Aquatic Resource Surveys. The estimates from these surveys can be visualized with linked micromaps.

13.2 Recent Geovisualizations with Environmental Data

Virginia Department of Environmental Quality and Florida Department of Environmental Protection both implemented spatially balanced probabilistic surveys of aquatic resources and visualized the results of the surveys using linked micromaps. Under the Clean Water Act in the United States, states must report on the condition or status of their lakes, rivers, streams, and estuaries. To meet this requirement, VDEQ has conducted such surveys of perennial, non-tidal, wadeable streams and rivers from 2001-2018 and has collected aquatic macroinvertebrates at stream sites for biological monitoring (Virginia Department of Environmental Quality, 2020). The aquatic macroinvertebrates indicate the ecological conditions of a stream and are expressed as a Stream Condition Index (SCI) on a numeric scale. From the stream surveys, estimates of the SCI score can be made among the 11 major river basins in Virginia and compared to a statewide threshold value (Virginia Department of Environmental Quality, 2020). Higher SCI scores suggest better ecological condition of the streams.

Unlike a chloropleth map that simply maps a summary statistic to color aesthetics of polygons, the SCI linked micromap produced by VDEQ displays *both* a measure of central tendency and a measure of variation for each of the 11 major river basins (Figure 13.1). Specifically, the statistical graphics column of the linked micromap shows the estimated median SCI score and the interquartile range (IQR), as the estimated 25th and 75th percentiles, with the basins ranked by their medians. Those statistics are compared to a statewide threshold SCI of 60 as shown by the vertical, black, dashed line. Close inspection of the linked micromap reveals a spatial dissimilarity for the southwestern basins of Holston, Clinch-Powell, and Big Sandy, all of which are part of the Tennessee River drainage. Holston has excellent condition based on its SCI score, Clinch-Powell has good condition, and Big Sandy has severe stress. Also, worth noting, is that the 75th percentile of the poorest scoring basin, Rappahannock, extends past the statewide threshold indicating that some sites might benefit from conservation efforts as opposed to poorer scoring sites, which might need remediation to improve their condition. The VDEQ Report included additional linked micromaps having two statistical graphics column, with one column showing the basin estimates of a water quality stressor, paired with another column of the SCI (Virginia Department of Environmental Quality, 2020). Pairing a potential stressor with the SCI response provided a visualization of stressors that might have a larger impact on aquatic macroinvertebrates relative to other basins (Virginia Department of Environmental Quality,

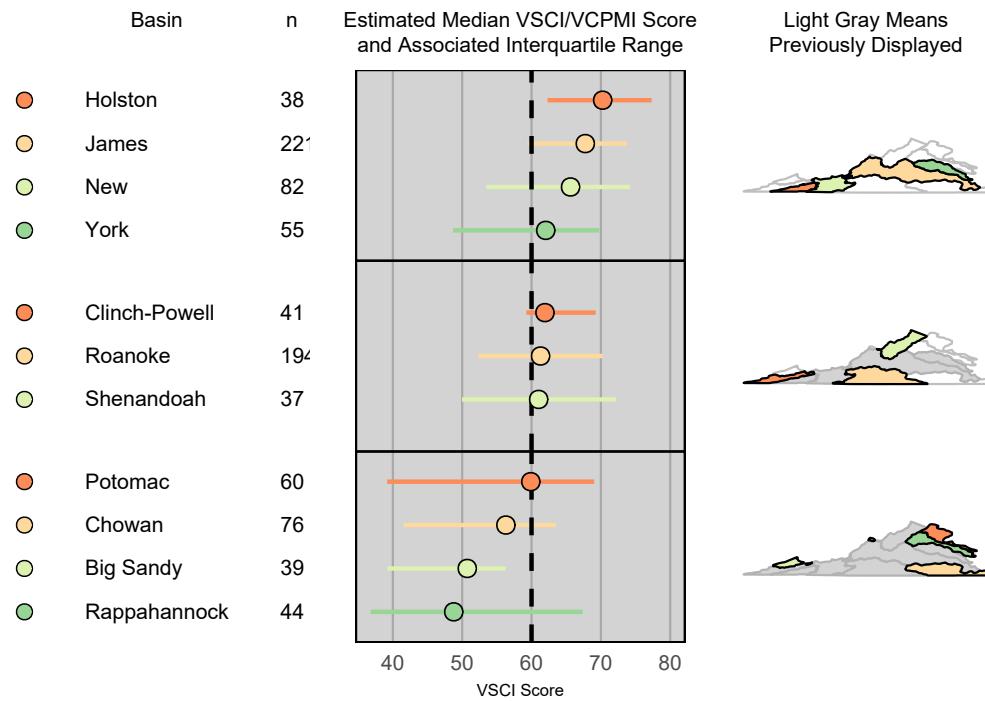


FIGURE 13.1 A linked micromap of stream condition for 11 basins from spatially balanced probabilistic surveys done by Virginia Department of Environmental Quality. Higher scores for Virginia Stream Condition Index (VSCI) and Virginia Coastal Plains Macroinvertebrate Index (VCPMI) indicate better ecological condition of the streams. Sample size, *n*, is indicated for each basin and median estimates and their interquartile ranges are shown in the statistical graphics column. The dashed vertical line at 60 is the threshold for biological impairment.

2020). The VDEQ linked micromap provided a means to show measures of variation, along with comparison to a reference value.

While the VDEQ R code builds off Chapter 2, some distinct arguments and layouts are worth mentioning. The VDEQ linked micromap features asymmetrical perceptual grouping as specified by `grouping = c(4, 3, 4)`, followed by `vertical.align = "center"` that aligns the rows within a perceptual group, rather than the default alignment of “top”. The center alignment is helpful when perceptual groups contain different number of rows. This is a five panel linked micromap showing a unique aspect of two label panels by using this syntax: `panel.types = c("dot_legend", "labels", "labels", "dot_cl", "map")`. The first label specification corresponds to the basin in the `panel.data`, while the second label specification corresponds to the sample size, *n*. From both of those calls, the linked micromap displays the basin names and their sample size. The VDEQ ecologists applied a novel use of `dot_cl` by reporting a three-number summary of estimates of the median, first quartile, and third quartile in their linked micromap. The `dot_cl` argument is typically used to report a central tendency estimate, such as a mean or median, and then the lower and upper confidence limits, as McManus et al. (2016) did. Finally, in that same statistical graphics column, the

overall statewide median estimate of SCI was displayed by using these arguments: `add.line = 60, add.line.col = "black", add.line.typ = "dashed"`.

Finally, the R code used to create Figure 13.1 is shown below, which uses the *stats* and *VA_map_table* datasets.

```
library(micromap)

VA_stats <- readRDS("data/stats.RDS")
VA_map_table <- readRDS("data/map.table.RDS")

mmpplot(
  stat.data = VA_stats,
  map.data = VA_map_table,
  map.link = c("Basin", "ID"),
  panel.types = c("dot_legend", "labels", "labels", "dot_cl", "map"),
  panel.data = list(NA, "Basin", "n", list("x50", "x25", "x75"), NA),
  ord.by = "x50",
  grouping = c(4, 3, 4),
  vertical.align = "center",
  colors = brewer.pal(3, "Spectral"),
  rev.ord = TRUE,
  panel.att = list(
    list(
      1,
      point.type = 20,
      point.border = TRUE,
      point.size = 2
    ),
    list(
      2,
      header = "Basin",
      panel.width = .6,
      align = "left",
      text.size = .9
    ),
    list(
      3,
      header = "n",
      panel.width = .2,
      align = "left",
      text.size = .9
    ),
    list(
      4,
      header = "Estimated Median VSCI/VCPMI Score\\nand Associated Interquartile Range",
      graph.bgcolor = "lightgray",
      point.size = 1.5,
      xaxis.ticks = list(40, 50, 60, 70, 80),
      xaxis.labels = list(40, 50, 60, 70, 80),
      yaxis.ticks = list(40, 50, 60, 70, 80),
      yaxis.labels = list(40, 50, 60, 70, 80)
    )
  )
)
```

```
add.line = 60,
add.line.col = "black",
add.line.typ = "dashed",
xaxis.title = "VSCI Score",
panel.width = 1.2
),
list(
  5,
  header = "Light Gray Means\nPreviously Displayed",
  map.all = TRUE,
  fill.regions = "aggregate",
  active.border.color = "black",
  active.border.size = 1.0,
  inactive.border.color = gray(.7),
  inactive.border.size = 1,
  panel.width = 1.0
)
)
```

The next example is from FDEP and uses several statistical graphics columns to summarize water quality and land cover data among watersheds. Studies of freshwater aquatic ecosystems often need to summarize water quality results from watersheds, as VDEQ did, along with land cover in those watersheds, which is how FDEP used linked micromaps. Spatially balanced probabilistic surveys were done by FDEP to study emerging contaminants from different types of waterbodies, canals, rivers, streams, small and large lakes, and unconfined aquifers, in 29 drainage basins (Silvanima et al., 2018). The results for one of the contaminants, the widely used pesticide imidacloprid, were pooled across waterbody types and summarized with a 3-statistical graphics column linked micromap of the 29 drainage basins displaying: 1) percentage of surveyed sites having detections of imidacloprid, 2) percentage of urban and agricultural land cover in a basin, and 3) a five-number summary, minimum, first quartile, median, third quartile, and maximum, of imidacloprid concentrations shown as a box plot (Silvanima et al., 2018). As an exploratory spatial data analysis tool, Silvanima et al. (2018) noted from the linked micromap that neighboring drainage basins showed no geographic patterning in percentages of detections, which was confirmed by a nonsignificant spatial autocorrelation using Moran’s Index. This application of linked micromaps was novel in two regards: first, by summarizing data from different sources (i.e., land use, water quality) for the polygons of interest, and second, by formally testing if there was geographic patterning in the results.

In Figure 13.2, the syntax: `panel.types = c("dot_legend", "labels", "labels", "dot", "dot", "box_summary", "map")` is used in the creation of boxplots. The `box_summary` requires that `FL_pesticide` dataset have columns containing the following summary statistics for imidacloprid concentrations for each drainage basin: minimum, first quartile, median, third quartile, and maximum. The boxplots are displayed in the third statistical panel, with the width of the boxplot specified with the argument: `graph.bar.size = 0.4`. The R code used to create Figure 13.2 is below. The `FL_pesticide` dataset had an NA entry for the Florida Keys regarding the percentage of samples having detectable imidacloprid concentrations as none of the freshwater bodies sampled by FDEP occurred there. Users may have to explicitly code how to handle NA values. In this case, the syntax `FL_pesticide$ranked_imida <-`

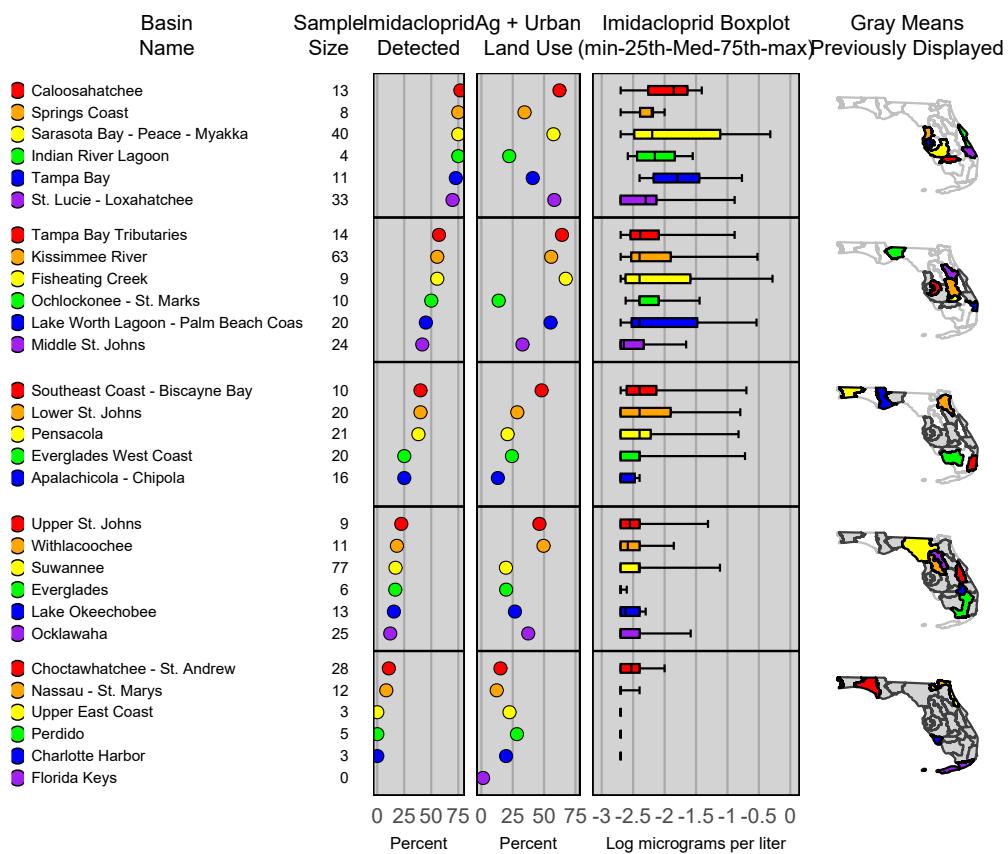


FIGURE 13.2 A linked micromap summarizing results for the pesticide Imidacloprid for 29 Florida basins from spatially balanced probabilistic surveys of freshwater bodies by Florida Department of Environmental Protection. Data are sorted by percentage of samples in a basin having detectable concentrations of the pesticide. The second statistical graphics column shows percent of agricultural and urban land cover in a basin. The third statistical graphics column displays boxplots of concentrations of the pesticide, with single vertical bars indicating samples having concentrations below instrument detection limits.

`rank(FL_pesticide$Imidacloprid, na.last = FALSE, ties.method = "first")` is used to create a new variable that ranked NA values last. The new variable was then used with the `ord.by` argument. A challenge mentioned in Chapter 2 and illustrated with the Florida linked micromap is that of small subareas, which is the Perdido basin, the westernmost basin in Florida.

```
library(micromap)

# data frames
FL_pesticide <- readRDS("data/FL_pesticide.RDS")
FL_basins <- readRDS("data/Florida_Basins.rds")
```

```
# code to handle NA values in rankings
FL_pesticide$Ranked_Imida <- rank(
  FL_pesticide$Imidacloprid,
  na.last = FALSE,
  ties.method = "first"
)

mmpplot(
  stat.data = FL_pesticide,
  map.data = FL_basins,
  map.link = c("GROUP_NAME", "ID"),
  panel.types = c(
    "dot_legend", "labels", "labels", "dot",
    "dot", "box_summary", "map"
  ),
  panel.data = list(
    NA, "GROUP_NAME", "Sum_PNT_COUNT", "Imidacloprid", "Percent_Ag_Ur_Tr",
    list("Imida_Min", "Imida_25", "Imida_Med", "Imida_75", "Imida_Max"),
    NA
  ),
  ord.by = "Ranked_Imida",
  rev.ord = TRUE,
  grouping = c(6, 6, 5, 6, 6),
  vertical.align = "center",
  plot.panel.spacing = 2,
  colors = c("red", "orange", "yellow", "green", "blue", "purple"),
  panel.att = list(
    list(
      1,
      point.type = 20,
      point.border = TRUE,
      panel.width = 1,
      point.size = 2
    ),
    list(
      2,
      header = "Basin\nName",
      panel.width = 1.1,
      align = "left",
      left.margin = -1.6,
      text.size = .65
    ),
    list(
      3,
      header = "Sample\nSize",
      panel.width = .1,
      align = "right",
      left.margin = -1.6,
      text.size = .65
    )
  )
)
```

```
),
list(
  4,
  header = "Imidacloprid\nDetected",
  graph.bgcolor = "lightgray",
  point.size = 1,
  xaxis.ticks = list(0, 25, 50, 75),
  xaxis.labels = list(0, 25, 50, 75),
  xaxis.title = "Percent",
  left.margin = -1.0,
  panel.width = .4
),
list(
  5,
  header = "Ag + Urban\nLand Use",
  graph.bgcolor = "lightgray",
  point.size = 1,
  xaxis.ticks = list(0, 25, 50, 75),
  xaxis.labels = list(0, 25, 50, 75),
  xaxis.title = "Percent",
  left.margin = -1.4,
  panel.width = .4
),
list(
  6,
  header = "Imidacloprid Boxplot\n(min-25th-Med-75th-max)",
  graph.bgcolor = "lightgray",
  xaxis.ticks = list(-3, -2.5, -2, -1.5, -1, -0.5, 0),
  xaxis.labels = list(-3, -2.5, -2, -1.5, -1, -0.5, 0),
  xaxis.title = "Log micrograms per liter",
  graph.bar.size = 0.4,
  left.margin = -1.4,
  panel.width = .85
),
list(
  7,
  header = "Gray Means\nPreviously Displayed",
  map.all = TRUE,
  fill.regions = "aggregate",
  active.border.color = "black",
  active.border.size = 1,
  left.margin = -0.8,
  panel.width = .7
)
)
```

13.3 Spatial Surveys and Linked Micromaps

Spatially balanced probabilistic surveys are often used to draw samples for creating summary statistics for areal units, such as watersheds or ecoregions, and we will illustrate the drawing of such a survey sample. For more details on the design and analysis of those surveys, the readers should consult Stevens Jr. and Olsen (2004), Olsen et al. (2012), and Dumelle et al. (2022). A spatially balanced probabilistic survey is connected to a linked micromap in that the survey provides the representative samples of the natural feature (lakes, rivers, streams, estuaries, ecoregions) from which estimates are made for the areal units. This is a unique application to environmental data by demonstrating how the **spsurvey** package can be used to draw samples that can be used to collect and summarize data with micromaps.

Three steps are required to draw a spatially balanced probabilistic survey using the R package **spsurvey**: 1) reading in the sample frame, which is a GIS file in the format of shapefile or an R simple feature, or sf, object, 2) specifying the design, and 3) selecting the sample (Olsen et al., 2012). The example sample frame we will use contains points for 3,190 lakes in the state of Oregon in the United States and is the same sample frame used by Olsen et al. (2012), where readers can get more details on the Generalized Random Tessellation Stratified (GRTS) algorithm. Unlike other spatially balanced sampling algorithms, GRTS can draw samples from sample frames of points (such as lakes), linear networks (such as rivers and streams), or polygons (such as estuaries) (Dumelle et al., 2022). For the Oregon lakes, we will draw an equal probability sample of 50 lakes where each site has an equal inclusion probability. This equal probability sample can more accurately represent the lake target population by accounting for the natural spatial aggregation of the lakes on the landscape.

We begin by plotting the Oregon border and lakes together in Figure 13.3. Note that both spatial layers are simple features objects and the same coordinate reference system or CRS is required to plot them together. We ensure the CRS of each sf object using the syntax `st_crs(OR) == st_crs(OR_lakes)`.

The *OR_lakes* dataset is the sample frame that will be used with the `grts` function from the **spsurvey** R package to draw a spatially balanced probabilistic sample. That function explicitly incorporates spatial locations into the survey design. Consequently, the sample frame needs to be in a projected coordinate reference system such as an Albers or UTM projection so a spatially balanced sample can be drawn (Olsen et al., 2012). An equal-probability draw of 50 lakes is done by specifying `eqprob <- grts(OR_lakes, n_base = 50)`. That sample of 50 lakes is shown in Figure 13.4.

We demonstrated a practical example of how to import a target population dataset and used the **spsurvey** to create a spatially balanced sample. This sample can then be used to collect and summarize data that can be further displayed with a linked micromap. Next, we demonstrate the value of using **spsurvey** for increasing confidence in summary statistics with an example from West Virginia Department of Environment Protection (WVDEP) (McManus et al., 2016).

The survey designs by VDEQ and FDEP that led to the linked micromaps (Figures 13.1 and 13.2) were more sophisticated than the example of an equal probability draw. Using a spatially balanced probabilistic survey drawn from **spsurvey** has two advantages. First, from a design perspective, one obtains a good spatial coverage of the features to be sampled. As Olsen et al. (2012) noted, users designing a spatially balanced survey often underestimate

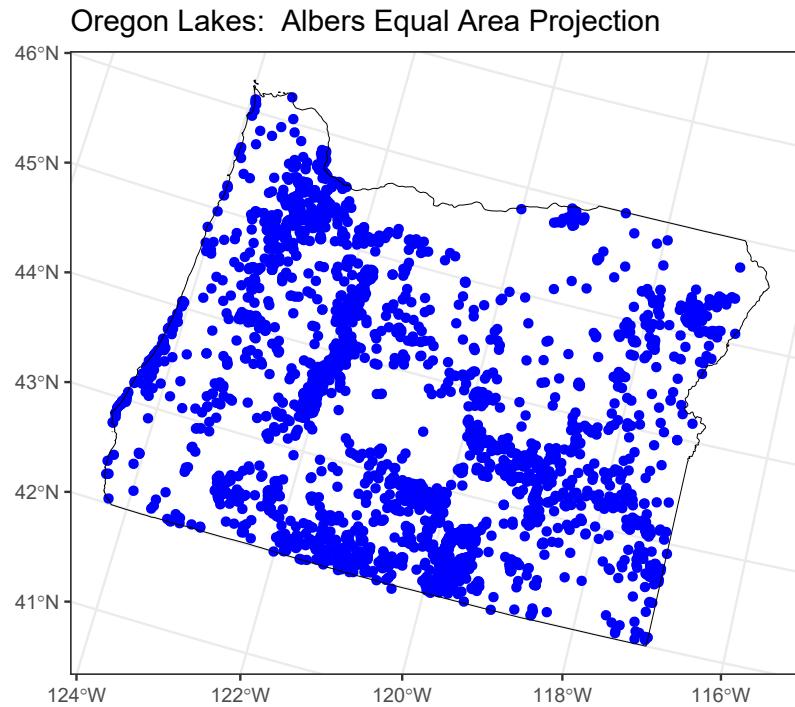


FIGURE 13.3 Map of Oregon lakes, with lakes represented as their centroids. The simple features object of the 3190 lakes is the sample frame from which the Generalized Random Tessellation Stratified function draws an $n = 50$ lakes for a spatially balanced probabilistic survey.

the effort required to build a single GIS layer having all the attributes needed for such a design. Second, from an analytical perspective, if a spatially patterned response is observed among the sample sites then the local neighborhood variance estimator in **spsurvey** tends to outperform other alternatives by providing smaller variances, and consequently, confidence limits around the estimated means, totals, percentiles, and cumulative distribution functions from the survey (Olsen et al., 2012; Stevens Jr. & Olsen, 2004). This advantage of the local neighborhood variance estimator over an independent random sample variance estimator has been shown in river and stream survey estimates of a fish community index of biotic integrity and a qualitative habitat evaluation index, and in reservoir survey estimates of chlorophyll-a (a measure of productivity), temperature, secchi depth (a measure of water clarity), and methane emission rates (a greenhouse gas) (Beaulieu et al., 2016; Stevens Jr. & Olsen, 2003, 2004). Consequently, in a linked micromap of spatially balanced probabilistic survey results, one could have smaller confidence limits displayed around the means or percentiles because of the improved performance of the local neighborhood variance estimate. This result is seen below for 25 basins in West Virginia with median estimates and 95% confidence limits for specific conductance, a measure of the ionic content of the sampled rivers and streams (Figure 13.5). The left statistical panel has confidence limits using the local neighborhood variance estimator, and the right panel used the simple random sample variance estimator. Some of the percentage reductions in confidence limits were quite modest. The Lower New basin had only ~ a 3% reduction; whereas others reductions were quite large, such as the Upper Ohio North/Upper Ohio South basin having a ~56% reduc-

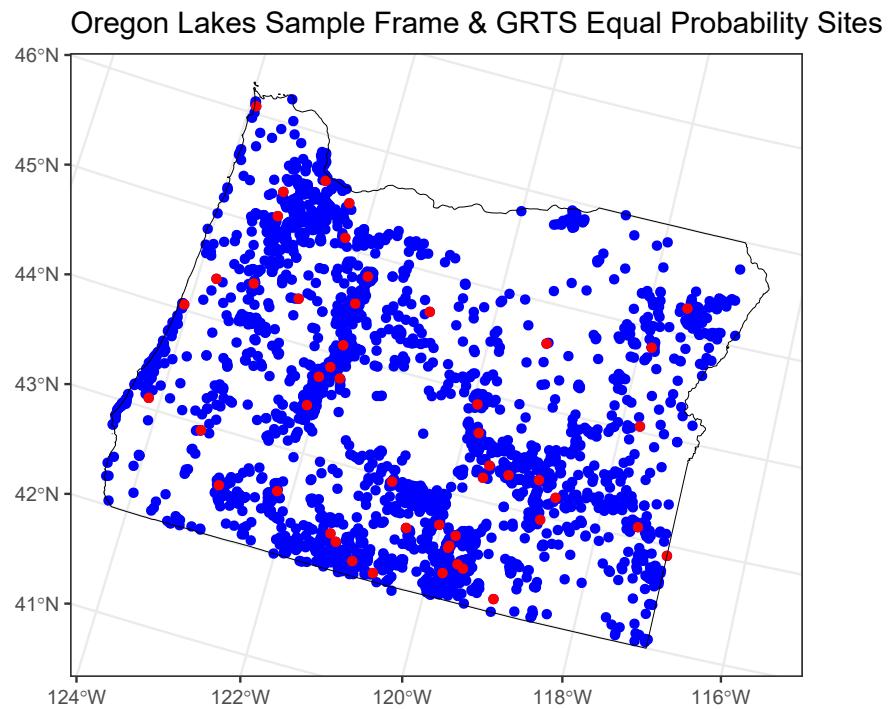


FIGURE 13.4 Oregon lakes Sample Frame & Equal Probability Sites in Projected Coordinates. Red points are the GRTS sample and blue points are the sample frame.

tion. The average percentage reduction among the 25 basins was ~28%. The independent random sample variance estimator does not take into account any spatial patterning in the sampled survey responses (Stevens Jr. & Olsen, 2003). If there is spatial patterning, such that responses at two sites close to each other are more similar than responses at two sites further apart, then that patterning is incorporated into the local neighborhood variance estimator (Stevens Jr. & Olsen, 2003).

Two final points about the WVDEP linked micromap should be noted. First, we used a simple features object, `sf`, that contained both the geospatial data and the statistical summaries of the basins. Consequently, we only had to specify the `map.data` argument, in contrast to the Virginia and Florida examples, where we used both the `map.data` and `stat.data` syntax. Second, the basins in the linked micromap were smoothed; that is their polygon boundaries generalized, so they would draw faster as mentioned in Chapter 2.

13.4 Summary and Further Reading

Building off of the foundation in Chapter 2, this chapter covered environmental applications of linked micromaps. Such applications have two unique aspects. One is the use of geographic boundaries from natural features, such as ecoregions or watersheds, instead of administrative or political boundaries that are more commonly used in other applications. See also Chapter 14 for medical applications that deal with non-standard geographic boundaries, such as

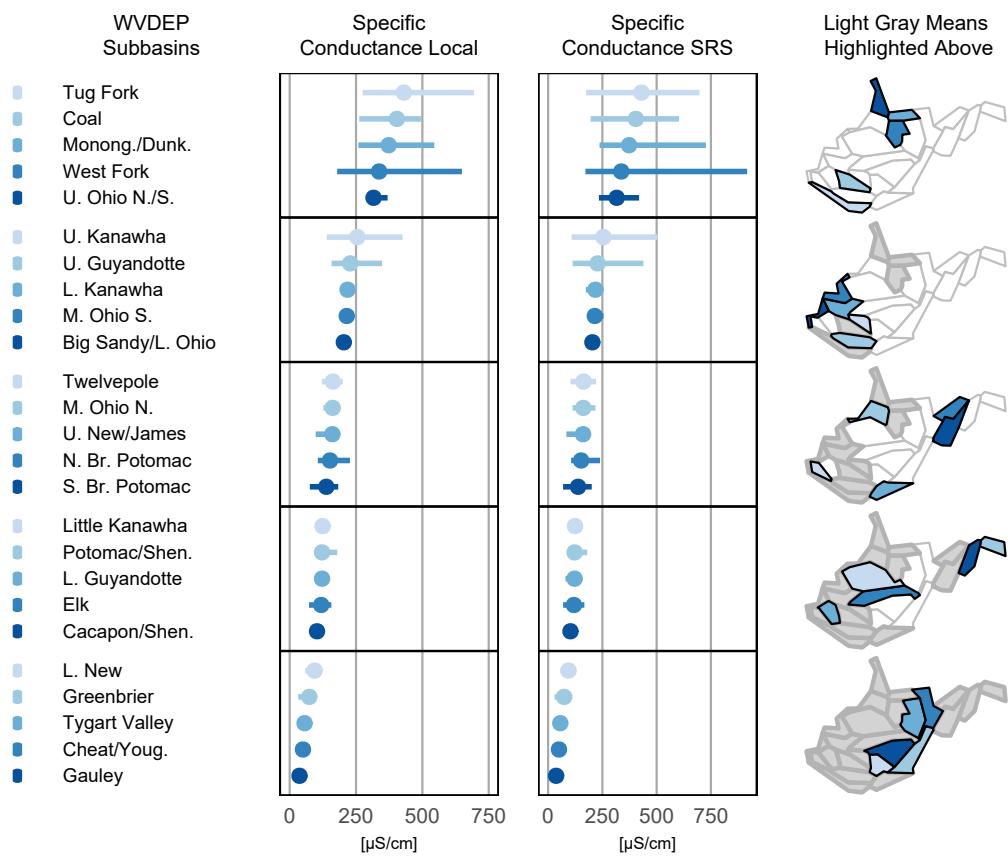


FIGURE 13.5 A linked micromap of specific conductance, a measure of the ionic content of the sampled rivers and streams, for 25 basins from spatially balanced probabilistic stream surveys done by West Virginia Department of Environmental Protection. The first statistical panel displays the median and 95% confidence limits based on the default variance local neighborhood estimator. The second statistical panel displays the median and 95% confidence limits based on specifying the simple random sample (SRS) variance estimator.

health service areas and census tracts. The other is the strong link between spatial survey design to obtain a representative sample from the natural features so that survey estimates and sample statistics can be displayed in linked micromaps. Data collected from spatially balanced probabilistic surveys by state environmental agencies were used in the linked micromap examples from this chapter. The FDEP example used a linked micromap for initial exploratory spatial data analysis and subsequent testing for spatial autocorrelation in pesticide detection among basins (Silvanima et al., 2018). McManus et al. (2016) used linked micromaps for a multivariate spatial analysis of WVDEP water quality data. The examples in this chapter were of static linked micromaps, but Chapter 9 covers interactive exploration of spatial data, including one by VDEQ.

Acknowledgements

The manuscript has been subjected to the U.S. Environmental Protection Agency's (USEPA) peer and administrative review and approved for publication. However, the views expressed are those of the authors and do not represent the views or policies of the USEPA. The mention of trade names or commercial products does not constitute endorsement or recommendation for use. We appreciate reviews provided by Michael Dumelle and Marc Weber at USEPA. We also want to acknowledge the data and input provided by Jason Hill and Emma Jones, both at VDEQ, James Silvanima at FDEP, and Michael Whitman at WVDEP.

References

- Beaulieu, J. J., McManus, M. G., & Nietch, C. T. (2016). Estimates of Reservoir Methane Emissions Based on a Spatially Balanced Probabilistic-Survey [<https://doi.org/10.1002/lno.10284>]. *Limnology and Oceanography*, 61(S1), S27–S40.
- Carr, D. B., Olsen, A. R., Courbois, J.-Y. P., Pierson, S. M., & Carr, D. A. (1998). Linked Micromap Plots: Named and Described. *Statistical Computing and Statistical Graphics Newsletter*, 9(1), 24–32.
- Dumelle, M., Kincaid, T. M., Olsen, A. R., & Weber, M. H. (2022). *spsurvey: Spatial Sampling Design and Analysis* [R package version 5.3.0 (<https://CRAN.R-project.org/package=spsurvey>)].
- McManus, M. G., Pond, G. J., Reynolds, L., & Griffith, M. B. (2016). Multivariate Condition Assessment of Watersheds with Linked Micromaps [<https://doi.org/10.1111/1752-1688.12399>]. *Journal of the American Water Resources Association*, 52(2), 494–507.
- Olsen, A. R., Kincaid, T. M., & Payton, Q. (2012). Spatially Balanced Survey Designs for Natural Resources [<https://doi.org/10.1017/CBO9781139022422.010>]. In R. A. Gitzen, J. J. Millspaugh, A. B. Cooper, & D. S. Licht (Eds.), *Design and Analysis of Long-Term Ecological Monitoring Studies* (pp. 126–150). Cambridge University Press, Cambridge, U.K.
- Payton, Q. C., McManus, M. G., Weber, M. H., Olsen, A. R., & Kincaid, T. M. (2015). micromap: A Package for Linked Micromaps [<https://doi.org/10.18637/jss.v063.i02>]. *Journal of Statistical Software*, 63(2), 1–16.
- Peterson, S. A., Urquhart, N. S., & Welch, E. B. (1999). Sample Representativeness: A Must for Reliable Regional Lake Condition Estimates [<https://doi.org/10.1021/es9807111>]. *Environmental Science & Technology*, 33(10), 1559–1565.
- Silvanima, J., Woeber, A., Sunderman-Barnes, S., Copeland, R., Sedlacek, C., & Seal, T. (2018). A Synoptic Survey of Select Wastewater-Tracer Compounds and the Pesticide Imidacloprid in Florida's Ambient Freshwaters [<https://doi.org/10.1007/s10661-018-6782-4>]. *Environmental Monitoring and Assessment*, 190(7), 435.
- Stevens Jr., D. L., & Olsen, A. R. (2003). Variance Estimation for Spatially Balanced Samples of Environmental Resources [<https://doi.org/10.1002/env.606>]. *Environmetrics*, 14(6), 593–610.
- Stevens Jr., D. L., & Olsen, A. R. (2004). Spatially Balanced Sampling of Natural Resources [<https://doi.org/10.1198/016214504000000250>]. *Journal of the American Statistical Association*, 99(465), 262–278.

Virginia Department of Environmental Quality. (2020). Chapter 4.4 Freshwater Probabilistic Monitoring Results [<https://www.deq.virginia.gov/home/showpublisheddocument/2221/637436316328230000>]. *2020 305(b)/303(d) Water Quality Assessment Integrated Report, EPA Approved, December 9, 2020* (pp. 112–152). Virginia DEQ, Richmond, VA.

14

Application of Linked Micromaps in the Medical Field

LINDA WILLIAMS PICKLE, JAMES BLACKWOOD PEARSON, JR.

14.1 Introduction

Public health studies often begin with an exploratory analysis of available data in order to generate hypotheses about a medical problem noticed in the population. Increasing disease rates over time or geospatial clusters of high rates typically trigger this sort of analysis. Questions asked in this preliminary stage include what lifestyle and environmental factors seem associated with the rates and where the problem seems to be concentrated. The latter “cluster analysis” leads epidemiologists to locations where a more in-depth study could be undertaken with sufficient numbers of cases for analysis. These questions lend themselves well to the use of linked micromaps.

In the early chapters of this book, the reader learned about the structure and use of linked micromap plots and how to create them using two different R packages. In this chapter, the linked micromap plot will be applied to health data to illustrate how it can be useful in an exploratory analysis and subsequent communication of results to the public.

14.2 Background

One of the successes in cancer screening has been the continuing decline of colorectal cancer (CRC) rates in the U.S. since screening methods became available about 50 years ago (Siegel et al., 2017). A colonoscopy can find and remove pre-cancerous polyps before they become malignant, thus reducing both incidence and mortality of CRC. However, it has become apparent recently that rates among people under age 50, the recommended age to start screening, have started to rise not just in the U.S. but around the world (National Cancer Institute (NCI Staff), 2020). As a result, many countries have recently lowered the initial screening age to 45. Researchers are puzzled by the increase in rates in the younger group (National Cancer Institute (C. Phillips), 2025). Risk factors identified for CRC occurrence in older adults include obesity, a sedentary lifestyle, smoking, excessive alcohol use and first degree relatives who also had CRC. Aspirin use, hormonal therapy and pre-cancerous polyp removal during screening are thought to reduce the risk of CRC. (National Cancer Institute, 2025a). In this chapter, we will explore CRC mortality data using the **micromapST** R package (Carr & Pearson Jr., 2015), accessible at <https://cran.r-project.org/package=micromapST>.

project.org/web/packages/micromapST/index.html to see if the plots can shed light on this new trend.

14.2.1 Data

Mortality data for U.S. states were obtained from the Centers for Disease Control and Prevention (CDC) online Wonder system (Centers for Disease Control and Prevention, 2025); rates are directly age-adjusted using the U.S. 2000 standard population. The rate in any strata, e.g., by state, age (<50, 50+), sex, race or year, was suppressed (shown as NA) in the original files if the number of cases was less than 16. Mortality data by state were available for 1999-2021, which were grouped into periods 1999-2001, 2002-2006, 2007-2011, 2012-2016, 2017-2021 for trend analysis. This aggregation over time eliminated most of the suppressed state counts. However, the District of Columbia (DC) still had suppressed counts and so was eliminated from the data file. Annual U.S. mortality rates for 1990-2022 were obtained from the NCI State Cancer Profiles web site (National Cancer Institute, 2025b). All rates are shown as the number of deaths per 100,000 population.

Not all of the risk and protective factors of interest were available from a nationwide survey that was reasonably representative of each state's population. However, the CDC's Behavioral Risk Factor Surveillance System (BRFSS) (National Cancer Institute, 2025b) provided state-level survey data for:

- Smoking: Ever smoked at least 100 cigarettes in their lifetime
- Binge drinking: On one occasion in the last 30 days had 5 or more alcoholic beverages (4 for women)
- Sedentary lifestyle: During the past month, had no leisure time physical activity
- Obesity (BMI ≥ 30): calculated from self-reported height and weight.
- CRC Screening: Received at least 1 recommended CRC screening test
- Poverty: persons living below the federal poverty level in each state. (National Cancer Institute, 2025b; United States Census Bureau, 2025)

Poverty was included because of its possible association with access to screening or other unmeasured socio-economic factors. All of these factors are represented as percentages in the linked micromap plots.

Risk factor data were available for 2022 but not earlier. Note that using this most recent risk factor data violates a major epidemiologic principle, i.e., that the putative risk factor must have occurred well prior to the initiation of the cancer. Nevertheless, these data will serve to illustrate the use of linked micromaps but we should be cautious about the interpretation of the results.

Data for Kentucky Area Development Districts were provided directly by Drs. Bin Huang and Thomas Tucker of the Kentucky Cancer Registry.

14.2.2 Preliminary U.S. Analysis

First, let's look at the U.S. CRC mortality rates over time, comparing rates among those under age 50 and those 50 and older.

```
library(micromapST)

# Input data files.
```

```
# Data Set # 1
mort_ratesUS <- read.csv("data/Ch12-data/USMortRateTrends.csv", skip = 1, header=TRUE)
head(mort_ratesUS, n = 3L)

##      Year WMFYoung WMYoung WFYoung WMOld WFOld WMFYoungINC PctWMFYoung
## 1 1990     1.6    1.8    1.4   83.9 105.9   69.6          NA
## 2 1991     1.6    1.7    1.5   82.0 102.6   68.6          NA       0.00%
## 3 1992     1.6    1.8    1.3   80.4 100.8   67.0          NA       0.00%
##   PctWMFOld AllMAllAges AllFAllAges AllMFAllAges WMFAllAges AllMFYoung AllMFOld
## 1           30.8        20.6        24.6      24.3     1.7     84.7
## 2      -5.90%        29.6        20.3        24.0      23.8     1.7     82.5
## 3      5.60%        29.3        19.9        23.6      23.4     1.7     81.1
##   BMFAllAges BMFYoung BMFOld
## 1     30.9      3.0 104.0
## 2     29.9      2.6 101.3
## 3     29.6      2.8  99.9

plot(mort_ratesUS$Year,mort_ratesUS$AllMFOld,type = 'l',xlab = 'Year',
      ylab = 'Age-adjusted rate', ylim = c(0,85))
lines(mort_ratesUS$Year,mort_ratesUS$AllMFYoung, lty = 'dashed', col = 'blue')
legend(x = 'topright', legend = c('ages 50+', 'ages <50'), col = c('black','blue'),
       lty = c('solid','dashed'))
```

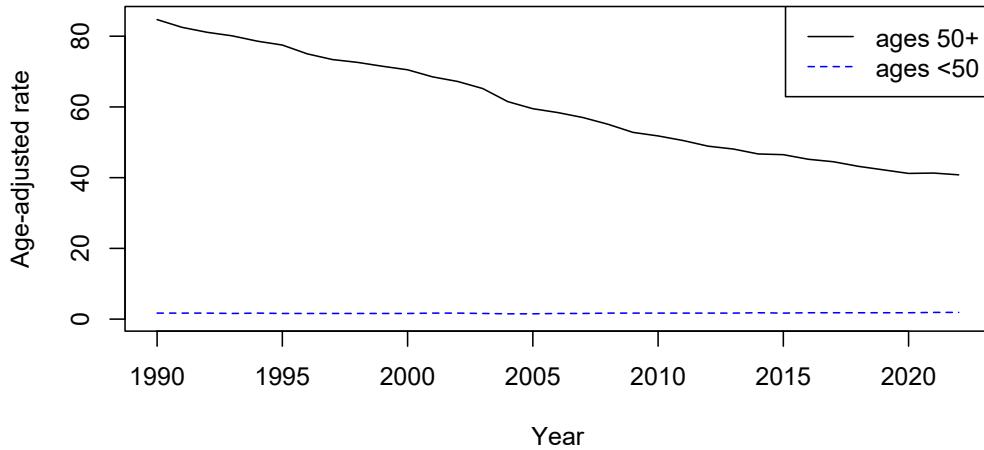


FIGURE 14.1 U. S. Colorectal cancer mortality rates, 1990-2022 by age group <50 and 50+

Because of the scale differences between the rates for the two age groups, we don't notice that the younger rates increased 20% while the older rates declined over 50% over this time span. From this point on, we will focus on the younger group. The numbers of deaths are

relatively small, despite increasing rates, on the order of 1.5 per 100,000 population. How can the data be aggregated to provide reasonably stable rates and to avoid too much data suppression for a visual analysis of state rates?

Aggregation over race/ethnic categories is a possibility, but because the non-Hispanic white category is so much larger than others in most states, aggregate patterns will largely reflect the patterns of whites. Therefore further analyses are restricted to non-Hispanic whites.

Exploratory analyses are often conducted separately on males and females but, in the interest of avoiding small numbers in the strata, could we combine the two groups?

```
plot(mort_ratesUS$Year,mort_ratesUS$WMYoung,type = 'l',lty = 'dotted', lwd = 2,
      col = 'blue', xlab = 'Year', ylab = 'Age-adjusted rate', ylim = c(0,3))
lines(mort_ratesUS$Year,mort_ratesUS$WFYoung, lty = 'dashed', col = 'green', lwd = 2)
lines(mort_ratesUS$Year,mort_ratesUS$WMFYoung, lty = 'solid', col = 'black', lwd = 2)
legend(x = 'bottomright', legend = c('Male','Female', 'Both'),
       col = c('blue','green','black'), lty = c('dotted','dashed','solid'), lwd = 2)
```

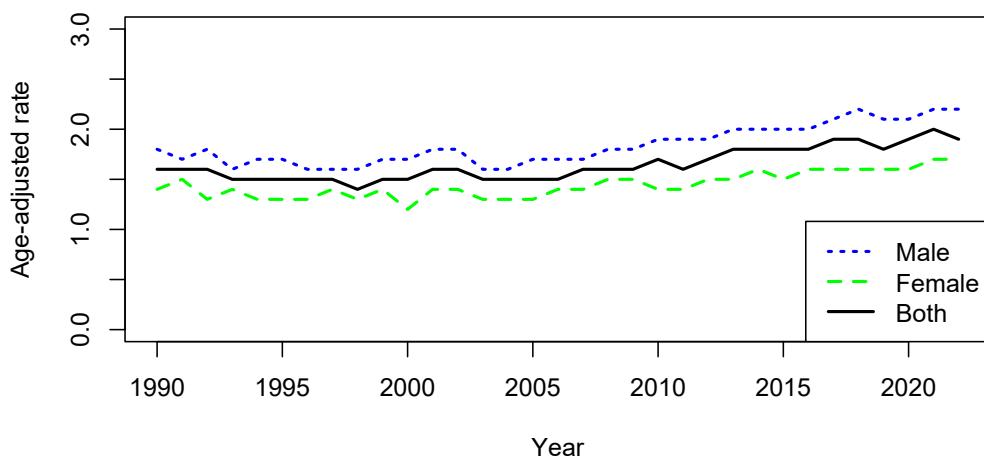


FIGURE 14.2 U. S. Colorectal cancer mortality rates among white non-Hispanics ages <50, 1990-2022 for males, females and both

Although male rates are slightly higher than female rates, the trend lines are fairly parallel. A trend analysis showed that rates in all three groups began to increase approximately 1.6% per year since 2004, compared to a decline of 2% or more per year for whites over age 50 (National Cancer Institute, 2025b). Subsequent analyses will combine white males and females.

One danger of any mortality analysis is that patterns in death rates might not reflect patterns in the underlying occurrence (incidence) due to changes in treatment or screening for that disease. For example, prostate cancer mortality rates dropped rapidly when the PSA screening test became widely available, even as incidence rates continued to rise. For CRC though, incidence rates for those under age 50 have been rising on pace with the

mortality rates. This suggests that changes in screening or treatment for CRC are not the causes of the rising mortality rates.

14.3 State-level Analysis

First, the mortality and risk factor data files are merged into a single working file. Column names for the mortality rates indicate the category for:

- sex (B, M, F for both, male, female respectively);
- year (all and 1-5 for 1999-2001, 2002-2006, 2007-2011, 2012-2016, 2017-2021);
- age (Y for ages <50; O for ages 50+)

Note that data for ages 50+ are excluded from the working file, named AllData. After calculating the percent change of rates over time, data for the first three states are shown.

```
#read csv files for mortality and risk factors, merge together.
#
#dDir <- "c:/projects/statnet/HealthChapter/"
#dDir <- "c:/MicromapBook/JuergenBase/MicromapPlotsInR-master/"
dDir <- "output/Ch12/"
File1 <- "data/Ch12-data/CRCMortData.csv"
File2 <- "data/Ch12-data/CRCRiskFactors.csv"

MortData <- read.csv(File1, header=TRUE)
RiskData <- read.csv(File2, header=TRUE)
MAbbr <- MortData[,2]
RAbbr <- RiskData[,1]
Mord <- order(MAbbr)
Rord <- order(RAbbr)

# exclude mortality data for age 50+ group
MortData2 <- MortData[Mord,-c(4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38)]
RiskData2 <- RiskData[Rord,]
names(MortData2)[2] <- "MAbbr"
names(RiskData2)[1] <- "RAbbr"
row.names(MortData2) <- MortData2$MAbbr
row.names(RiskData2) <- RiskData2$RAbbr
AllData <- cbind(MortData2,RiskData2)

stAbbr <- MortData2$MAbbr

# Linked MM Columns: latest time period rate, % time change from period 1 to 5, full time series
# Compute % change,
SexBYrRates <- MortData2[,c("MAbbr","SexBYrAllAgeY","SexBYr1AgeY","SexBYr2AgeY","SexBYr3AgeY","SexBYr4AgeY")]
row.names(SexBYrRates) <- stAbbr
names(SexBYrRates) <- c("Abbr","Yr0","Yr1","Yr2","Yr3","Yr4","Yr5") # Yr0 is the ALL data over 1 to 5.
ColNames <- c("Yr0", "Yr1", "Yr2", "Yr3", "Yr4", "Yr5")
```

```

SexByrRatesArr <- array(dim=c(50,5,2),dimnames=list(stAbbr, ColNames[2:6]) )

for (ind in row.names(SexByrRatesArr)) {
  y <- as.numeric(SexBYrRates[ind,c("Yr1","Yr2","Yr3","Yr4","Yr5")])
  SexByrRatesArr[ind,,1] <- c(1,2,3,4,5) # x value for each y
  SexByrRatesArr[ind,,2] <- y
}

AllData <- cbind(MortData2,RiskData2)
AllData$Delta51 <- ( AllData$SexBYr5AgeY - AllData$SexBYr1AgeY )
AllData$PctChange <- ( AllData$SexBYr5AgeY - AllData$SexBYr1AgeY ) / AllData$SexBYr1AgeY * 100
head(AllData, n = 3L)

##      StateName MAabbr SexBYrAllAgeY SexMYrAllAgeY SexFYrAllAgeY SexBYr1AgeY
## AK      Alaska   AK          1.1          1.2          1.0        NA
## AL     Alabama   AL          2.0          2.2          1.8        1.5
## AR    Arkansas   AR          2.2          2.4          1.9        1.9
##      SexMYr1AgeY SexFYr1AgeY SexBYr2AgeY SexMYr2AgeY SexFYr2AgeY SexBYr3AgeY
## AK          NA          NA          NA          NA          NA        1.4
## AL          1.8          1.3          2.0          2.2          1.7        2.0
## AR          2.2          1.7          2.1          2.5          1.6        2.2
##      SexMYr3AgeY SexFYr3AgeY SexBYr4AgeY SexMYr4AgeY SexFYr4AgeY SexBYr5AgeY
## AK          NA          NA          1.3          NA          NA        1.5
## AL          2.3          1.6          2.2          2.2          2.2        2.4
## AR          2.3          2.1          2.3          2.3          2.3        2.2
##      SexMYr5AgeY SexFYr5AgeY RAabbr PctBinge PctEverSmk PctNoExer PctCRCScreen
## AK          NA          NA       AK        17.0        41.7        20.9       62.2
## AL          2.6          2.2       AL        12.9        38.2        29.1       67.7
## AR          2.7          1.8       AR        13.5        42.7        32.5       64.0
##      PctObese PctPov Delta51 PctChange
## AK      35.8    10.7      NA        NA
## AL      39.5    16.7      0.9      60.00
## AR      40.8    17.0      0.3      15.79

# set up micromapST
#
#Columns: latest time period rate, % time change from period 1 to 5, full time series.

panelHC01 <- data.frame(type=c("mapcum","id","dot","dot","ts"),
  lab1=c(NA, NA, "Rate", "Pct Change", "Time Series"),
  lab2=c(NA, NA, "2017-2021", "1999-2001 to 2017-2021", "1999-2021"),
  col1=c(NA, NA, "SexBYr5AgeY", "PctChange", NA),
  panelData=c(NA,NA, NA,           "SexByrRatesArr"),
  refVals=c(NA,NA, 1.8,            20,           NA),
  refTexts=c(NA,NA, "US 2017-2021", "US Pct Chg", NA)
)

micromapST(AllData,panelHC01,sortVar="SexBYr5AgeY",ascend=FALSE,

```

```
rowNames="ab", rowNamesCol="MAbbr"
```

```
)
```

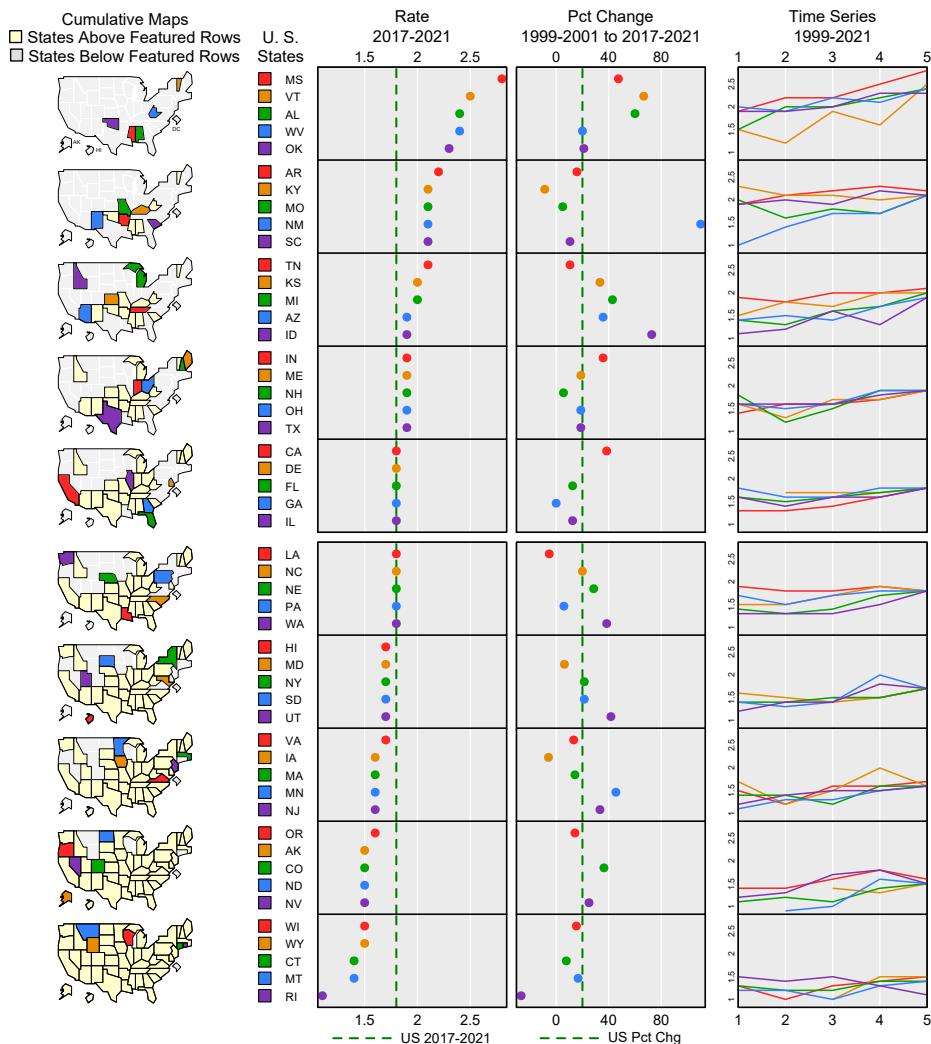


FIGURE 14.3 Time trends and recent rates of colorectal cancer mortality among whites ages <50 by state

The first linked micromap plot, Figure 14.3, displays state mortality rates for the latest

time period (2017-2021), the calculated percent change in rates from the first to last time period, and a time series plot of all of the period rates. This plot is sorted in descending order by the latest period rates. Most of the highest rates for 2017-2021 are in the southern U.S. The U.S. rate (1.8) is shown as a vertical green line for reference. As we can see from the cumulative shading on the maps (second map from the top), 9 of the 10 highest-rate states cluster in the south-central region. The time series plots of the five time-period rates show that changes have been gradual over time, not a sudden recent spike.

Resorting by the percent change in rates (Figure 14.4), we can also see that many states with the highest recent rates have also had the most rapid increases over the past 20 years.

```
micromapST(AllData,panelHC01,sortVar="PctChange",ascend=FALSE,
            rowNames="ab", rowNamesCol="MAbbr"
          )
```

The next two linked micromap plots display the most recent mortality rates alongside the measurable risk factors (Figure 14.5) and screening and poverty (Figure 14.6). Since the states are sorted by recent rates in descending order, we can look for associations between rates and the plotted factors by looking at the shape of the dot pattern reading from top to bottom. A dot pattern that parallels the rate dots suggests a positive correlation while negative correlation appears as the opposite dot pattern (i.e., factor rates are low where mortality rates are high and vice versa).

The recent CRC rates appear to be positively correlated with obesity, sedentary lifestyle (no exercise) and poverty, consistent with findings in older adults. However, binge drinking appears to be somewhat negatively correlated with CRC rates, unlike previous findings in those over age 50. Screening rates appear to have no relationship with the mortality rates, not surprising since asymptomatic patients under age 50 have not been screened regularly until recently.

```
panelHC03 <- data.frame(type=c("mapcum","id","dot",      "dot",      "dot",      "dot",      "dot"),
                         lab1=c(NA,    NA, "Rate", "Pct Ever", "Pct",     "Pct Binge",   "Pct"),
                         lab2=c(NA,    NA, "2017–2021", "Smoked",   "Obese",   "Drinking",   "No Exercise"),
                         col1=c(NA,NA,"SexBYr5AgeY","PctEverSmk","PctObese","PctBinge","PctNoExer"),
                         refVals=c(NA,NA,  1.8,           NA,        NA,        NA,        NA),
                         refTexts=c(NA,NA, "US Rate",    NA,        NA,        NA,        NA)
                       )

Title<-c("Colorectal Cancer Mortality Among whites, < 50,","and related
lifestyle factors.")

micromapST(AllData,panelHC03,sortVar="SexBYr5AgeY",ascend=FALSE,
            rowNames="ab", rowNamesCol="MAbbr", staggerLab=TRUE      )
```



```
panelHC05 <- data.frame(type=c("mapcum","id","dot",      "dot",      "dot"),
                         lab1=c(NA,    NA, "Rate", "Pct CRC",     "Pct"),
                         lab2=c(NA,    NA, "2017–2021", "Screening", "Poverty"),
                         col1=c(NA,    NA, "SexBYr5AgeY", "PctCRCScreen", "PctPov"),
                         refVals=c(NA,NA,  1.8,           NA,        NA),
                         refTexts=c(NA,NA, "US Rate",    NA,        NA))
```

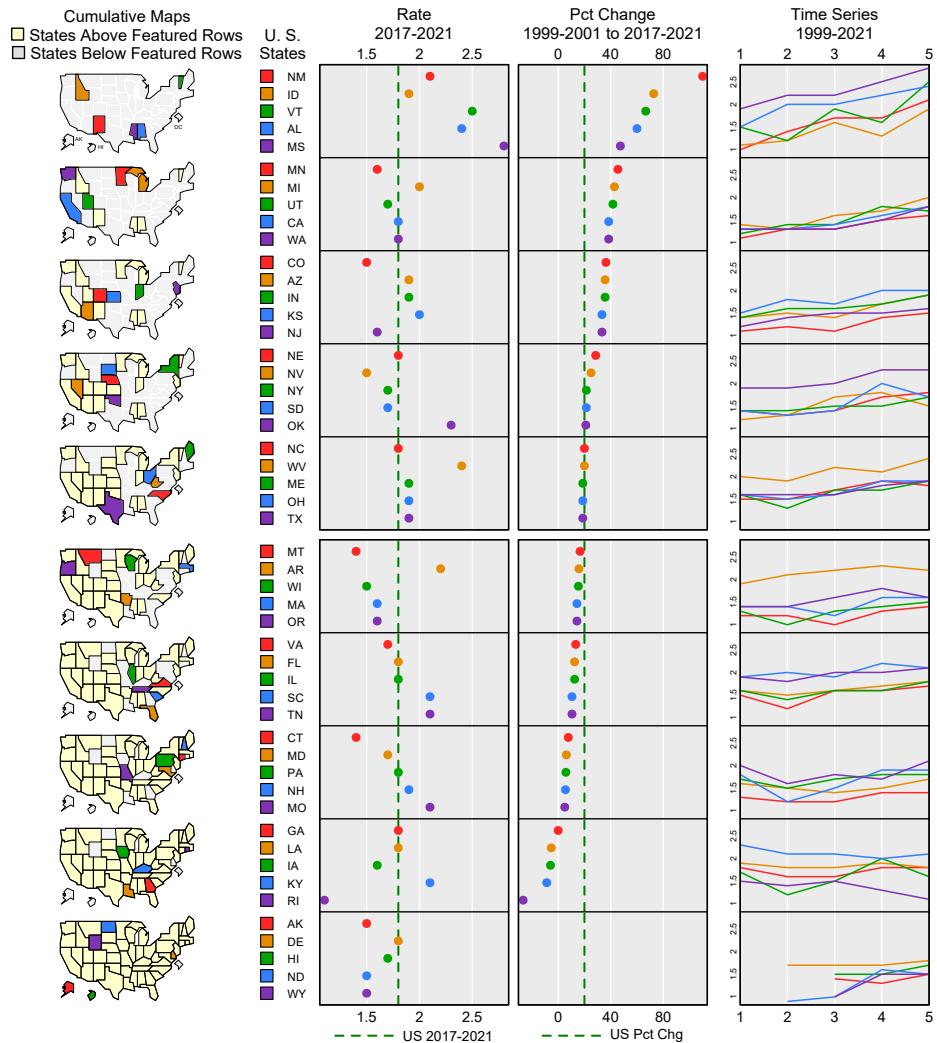


FIGURE 14.4 Time trends of colorectal cancer mortality, sorted by percent change, among whites ages < 50, by state

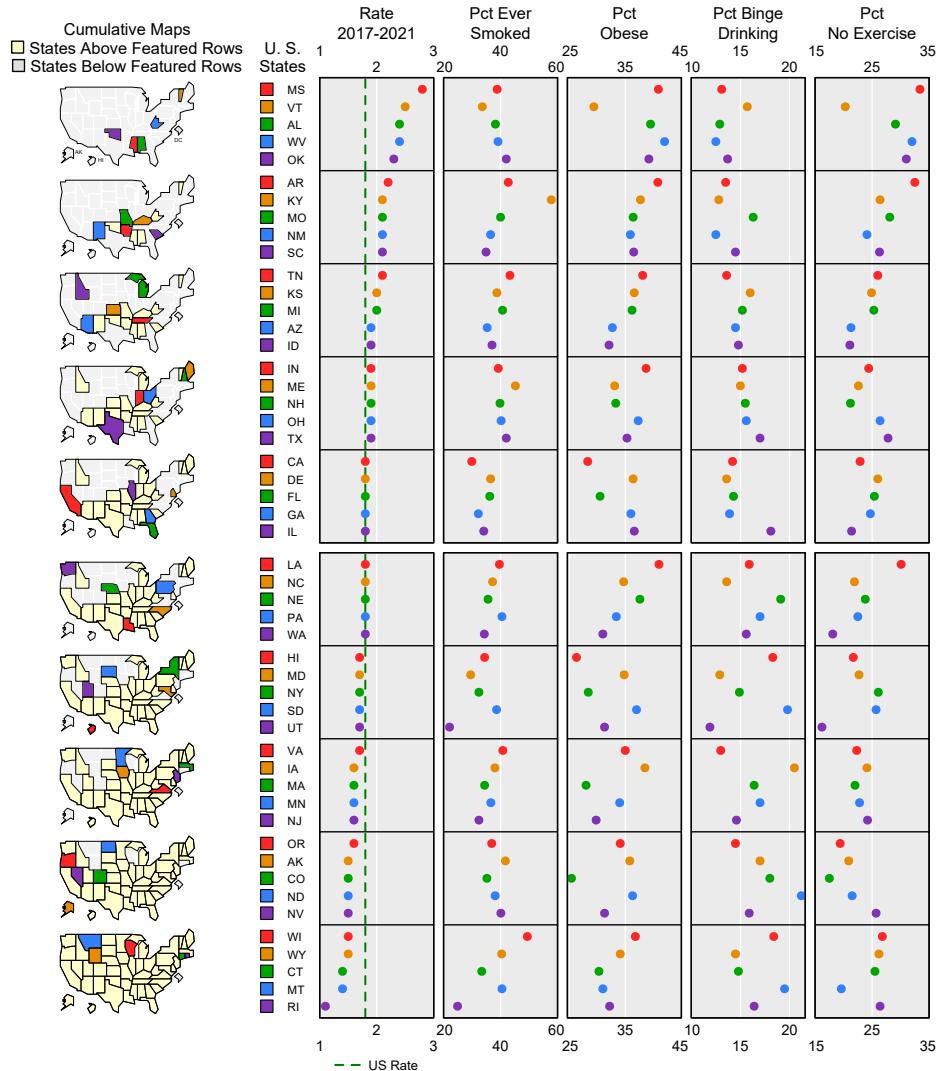


FIGURE 14.5 Colorectal cancer mortality among whites ages < 50, and related lifestyle factors

```

    )

micromapST(AllData,panelHC05,sortVar="SexBYr5AgeY",ascend=FALSE,
            rowNames="ab", rowNamesCol="MAbbr", staggerLab=TRUE
)

```

In addition to data exploration, we can use the linked micromap plots to communicate our findings to others. The next two plots illustrate different ways to do that - using simple dot plots or comparing state rates to each risk factor by means of a scatter plot. A new feature of micromapST, the locally weighted scatterplot smoothing line (LOWESS; (Cleveland, 1981)), is added to aid trend visualization.

```

panelHC07 <- data.frame(type=c("mapcum","id","dot",      "dot",      "dot",      "dot",      "dot"),
                         lab1=c(NA,     NA,   "Rate",      "Pct Chg","Pct",      "Pct",      "Pct"),
                         lab2=c(NA,     NA,   "2017-2021", "from 1999-2001", "Poverty", "Obesity",   "No Exerc",
                         col1=c(NA,     NA,   "SexBYr5AgeY", "PctChange", "PctPov",   "PctObese",  "PctNoExer"),
                         refVals=c(NA,NA,   1.8,        20,          NA,         NA,         NA),
                         refTexts=c(NA,NA,  "USYr5",    "USPctC",    NA,         NA,         NA))

micromapST(AllData,panelHC07,sortVar="SexBYr5AgeY",ascend=FALSE,
            rowNames="ab", rowNamesCol="MAbbr", staggerLab=TRUE      )

```



```

micromapST(AllData,panelHC07,sortVar="PctObese",ascend=FALSE,
            rowNames="ab", rowNamesCol="MAbbr", staggerLab=TRUE
)

```

Eliminate X and Y pairs and States that have an NA as a value.

```

xyx <- is.na(AllData$SexBYr5AgeY)
xx1 <- is.na(AllData$PctPov)
xx2 <- is.na(AllData$PctObese)
xx3 <- is.na(AllData$PctNoExer)
xBad <- xyx | xx1 | xx2 | xx3
xGood <- !xBad

AllData2 <- AllData[xGood,]

panelHC10 <- data.frame(type=c("mapcum", "id", "dot","scatdot", "scatdot", "scatdot"),
                          lab1=c(NA,     NA,   "Rate","Rate vs.", "Rate vs.", "Rate vs."),
                          lab2=c(NA,     NA,   NA,   "Pct Poverty","Pct Obese","Pct No Exer."),
                          col1=c(NA,     NA,   "SexBYr5AgeY", "PctPov", "PctObese", "PctNoExer"),
                          col2=c(NA,     NA,   NA,   "SexBYr5AgeY", "SexBYr5AgeY", "SexBYr5AgeY"))

)

```

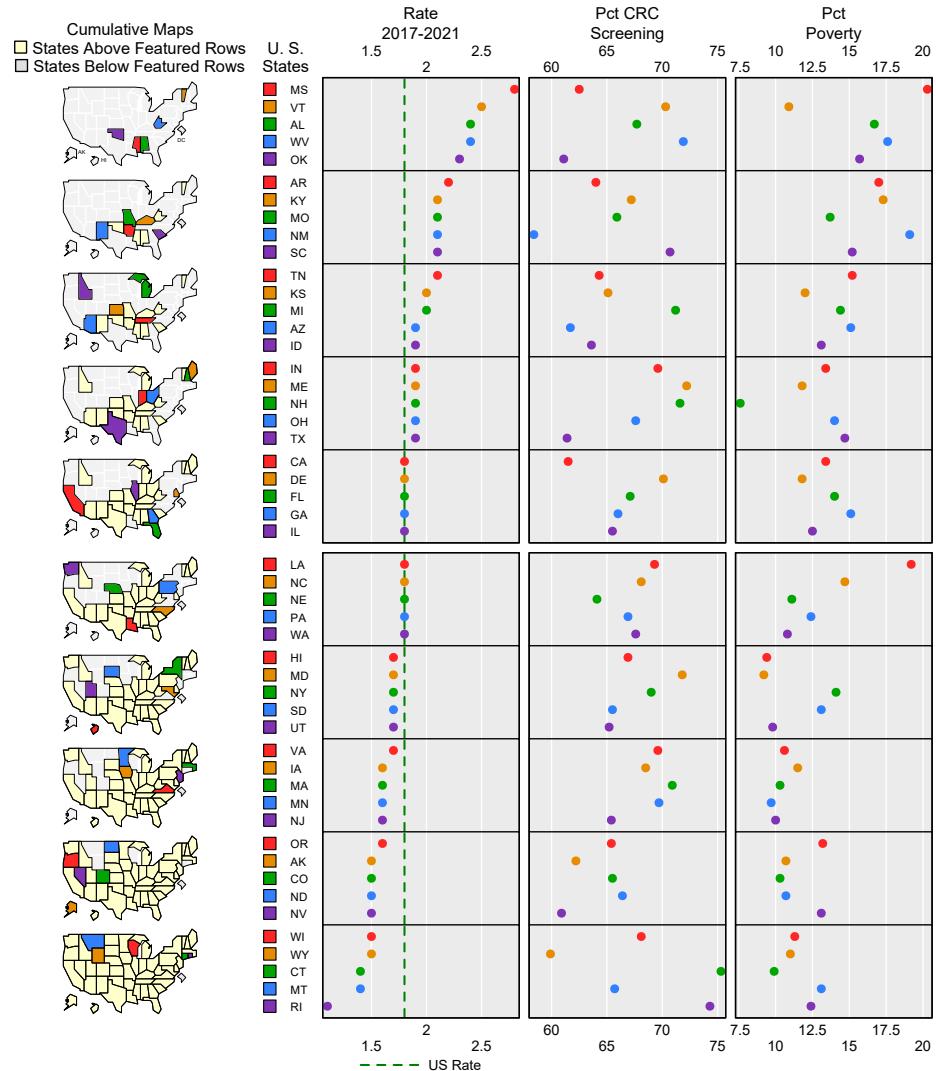


FIGURE 14.6 Mortality rates and screening for colorectal cancer, among whites ages < 50

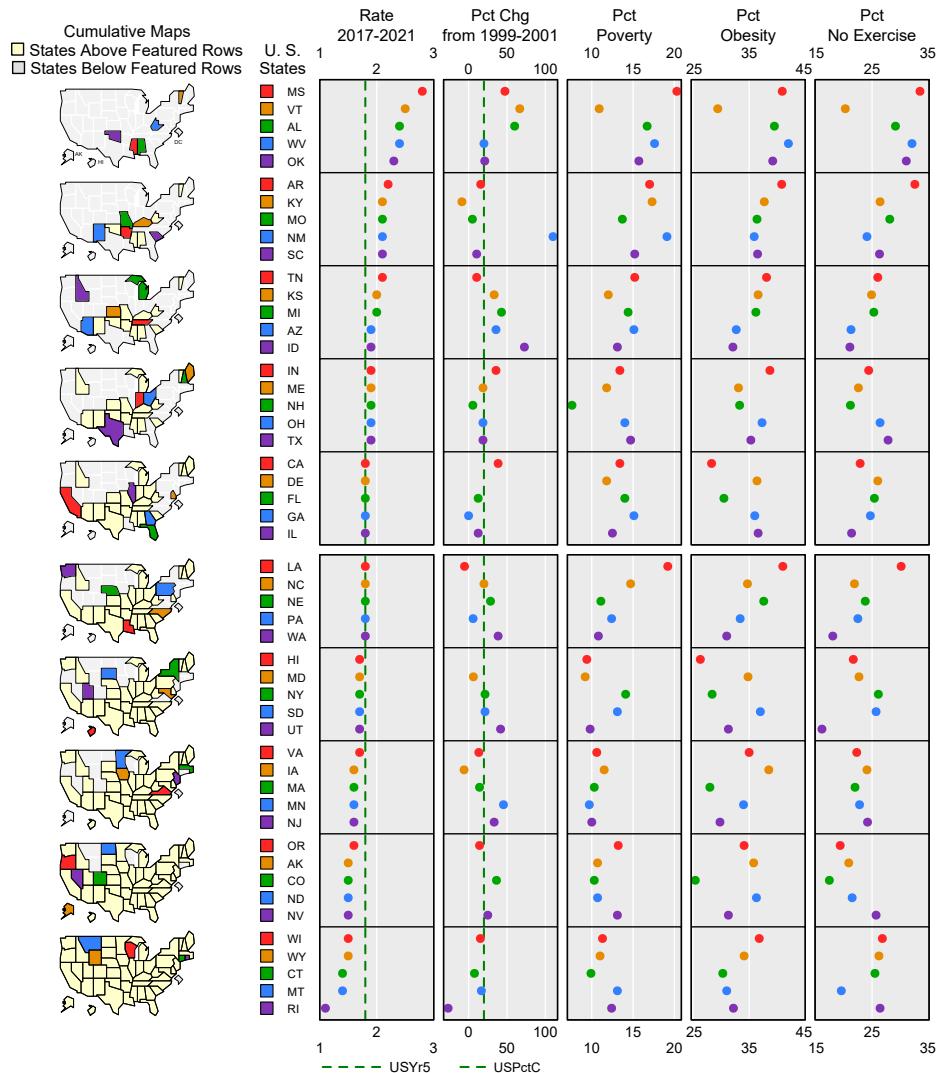


FIGURE 14.7 Colorectal cancer mortality among whites ages < 50 and Pct Poverty, Pct Obese, Pct No Exercise

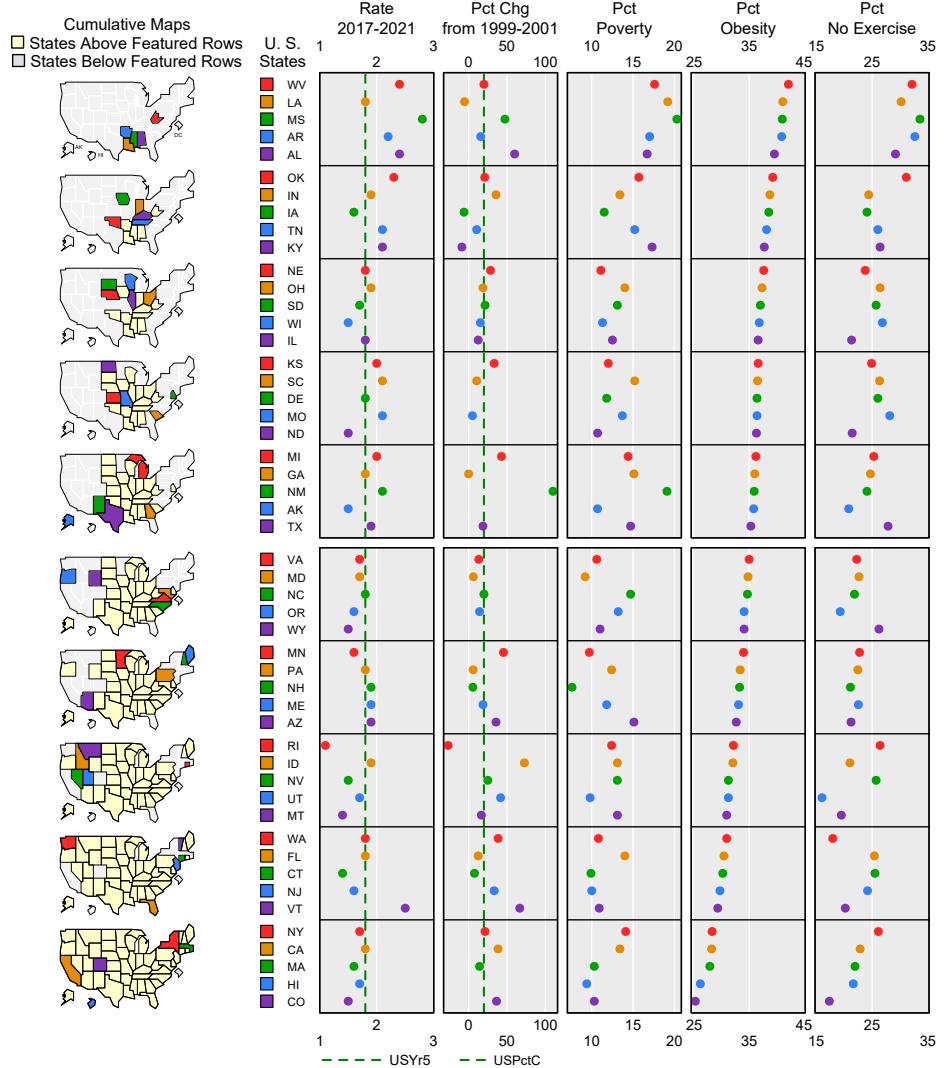


FIGURE 14.8 Colorectal cancer mortality among whites ages < 50 and Pct Poverty, Pct Obese, Pct No Exercrise

```

parmx <- list(list(NA), list(NA), list(NA), list(line="LOWESS",line.col="RED",line.lwd=1.25), list(line="LOWESS",
      line.col="RED",line.lwd=1.25))

panelHC10[,"parm"] <- list(parmx)

micromapST(AllData2,panelHC10,sortVar="SexBYr5AgeY",ascend=FALSE,
           rowNames="ab", rowNamesCol="MAbbr")

```

The scatter plots clearly show that Vermont (VT) is a high-rate outlier even though it is one of the lower states for percent poverty, obesity and sedentary lifestyle. Mississippi (MS) has the highest rate, slightly higher than Vermont, but also has high levels of all three risk factors. The highest obesity rates are in states along the Mississippi River (Figure 14.8). Rates vary across states almost three-fold, from Rhode Islane (RI) to Mississippi. The LOWESS scatter plot smoothing line has a positive slope for all three plotted factors, indicating a positive association between them and the mortality rates.

14.4 Applying Linked Micromap Plots to Sub-state Data

As we saw in Chapter 5, the **micromapST** package is no longer limited to displaying U.S. states. To illustrate, we will explore patterns in colorectal cancer mortality rates among those under age 50 in Kentucky. Low rates in this age group preclude examination by county, so we will display rates by Area Development District (ADD), aggregations of the 120 Kentucky counties (Kentucky Council of Area Development Districts, 2025). First, we need to create a working file of mortality and risk factor data. The border group (boundary file) used for these displays of Kentucky ADD data was created in Chapter 5.

```

library(micromapST)
library(stringr)
# BordDir     <- "data/Ch4b/"
# bdGroup     <- "KYADD2BG"
BordDir     <- "data/Ch12-data/"
bdGroup     <- "KYADD2BG"
BordGrpKYADDFile <- paste0(BordDir,bdGroup,".rda")

BGData      <- load(file=BordGrpKYADDFile)

ShtNT       <- (areaNamesAbbrsIDs[,c("Name","Abbr","ID")])
ShtNT$Name <- str_to_upper(ShtNT$Name)

# BuildBorderGroup saves a copy of this table for publication to
# <bgname>_rpt.txt in the same directory the <bgname>.rda file is written.
#
# Load the data for the linked micromap
MortName   <- "data/Ch12-data/KYCRCData.csv"
RiskName   <- "data/Ch12-data/KYADDRiskFactors.csv"

```

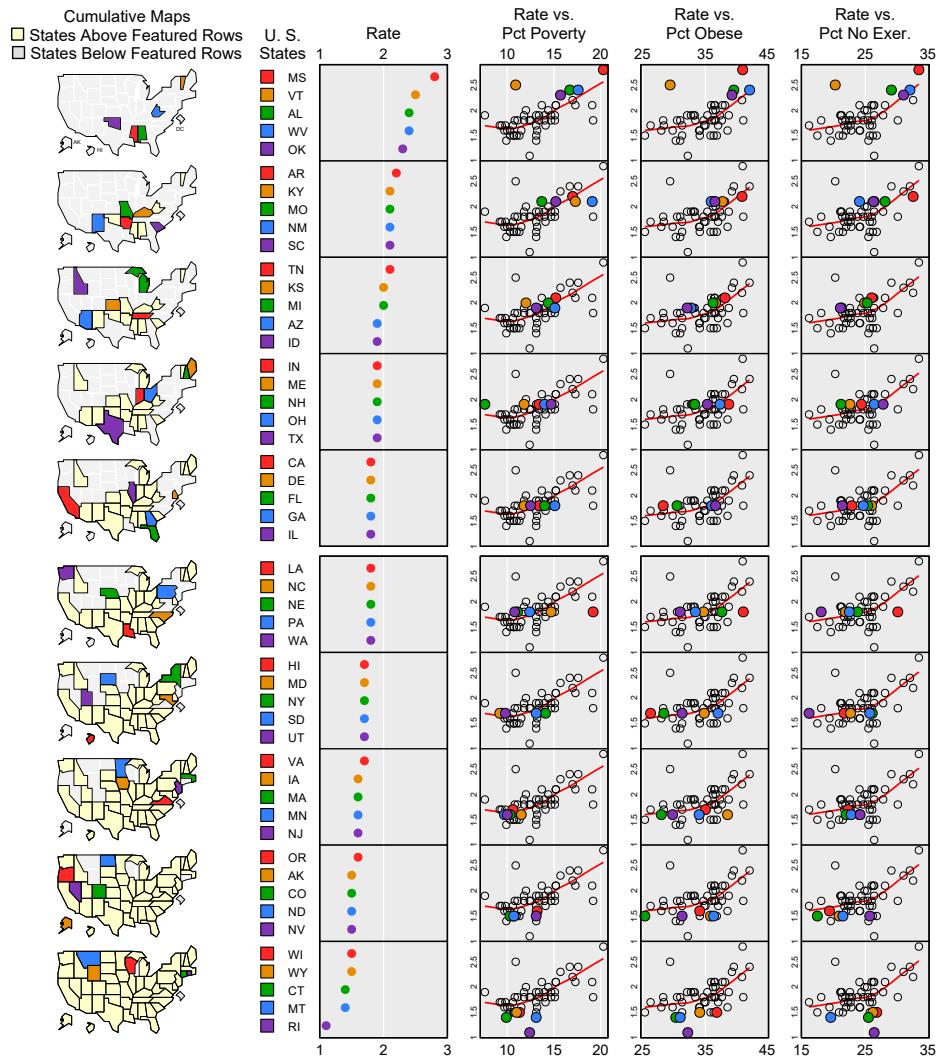


FIGURE 14.9 Colorectal cancer mortality among whites ages < 50, scatter plots of rates and main risk factors

```

KYADDMort <- read.csv(MortName,header=TRUE,stringsAsFactors=FALSE)
KYADDRisk <- read.csv(RiskName,header=TRUE,stringsAsFactors=FALSE)

KYADDRisk[, "X"] <- NULL
names(KYADDRisk)[1] <- "KYADD"
KYADDMort$KYADD <- str_to_upper(KYADDMort$KYADD)
KYADDRisk$KYADD <- str_to_upper(KYADDRisk$KYADD)

xm <- match(KYADDMort$KYADD,KYADDRisk$KYADD)

KYADDAllData <- cbind(KYADDMort,KYADDRisk[xm,-1])

xm <- match(KYADDAllData$KYADD,ShtNT$Name)

KYADDAllData$Abbr <- ShtNT$Abbr[xm]

KYADDAllData          # print dataset

```

		KYADD	nYr4	RateYr4	nYr5	RateYr5	KYcursmk	KYobesity	KYpoverty
## 1	BARREN RIVER	21	2.1	23	2.9	16.43	44.17	12.37	
## 2	BIG SANDY	17	3.0	14	3.3	19.77	42.20	20.32	
## 3	BLUEGRASS	62	2.4	39	1.9	16.97	31.40	10.02	
## 4	BUFFALO TRACE	NA	NA	NA	NA	16.77	36.64	16.14	
## 5	CUMBERLAND VALLEY	28	3.2	12	1.8	16.52	42.43	21.40	
## 6	FIVCO	10	1.9	NA	NA	24.62	52.19	13.90	
## 7	GATEWAY	NA	NA	NA	NA	18.53	34.63	14.77	
## 8	GREEN RIVER	13	1.8	18	3.2	20.32	39.20	11.63	
## 10	KENTUCKY RIVER	16	3.7	NA	NA	24.92	46.95	22.41	
## 9	KIPDA	45	1.6	40	1.7	14.36	36.98	9.30	
## 11	LAKE CUMBERLAND	18	2.4	12	2.2	21.22	35.16	16.88	
## 12	LINCOLN TRAIL	15	1.6	17	2.4	17.51	36.87	10.05	
## 13	NORTHERN KENTUCKY	28	1.7	26	2.0	16.27	35.68	7.03	
## 14	PENNYRILE	NA	NA	11	2.1	21.89	39.69	13.03	
## 15	PURCHASE	NA	NA	10	2.0	16.55	42.90	11.79	
	KYCRCScreen Abbr								
## 1	61.00	BR							
## 2	56.45	BS							
## 3	66.57	BG							
## 4	63.99	BT							
## 5	61.22	CV							
## 6	65.09	FI							
## 7	63.06	GA							
## 8	68.21	GR							
## 10	66.24	KR							
## 9	63.14	KI							
## 11	57.75	LC							
## 12	62.83	LT							
## 13	64.15	NK							
## 14	68.15	PE							
## 15	70.59	PU							

Next, we will display a linked micromap plot for the 2017-2021 rates and the change in rates since 2012-2016.

```
# NOTE: NA data is present in the rate data for Buffalo Trace, FIVCO, Gateway,
# Kentucky River, Pennyville and Purchase.

# Assumptions are the rows will have the ADD abbreviation as the row.name and
# the columns provided are: RateYr5, and RateYr4 for the rates and
# KYCRCScreen, KYobese, KYpoverty, and KYcursmk as the percentage values.
# 

panelHC11 <- data.frame(type=c("mapcum", "id", "dot",           "arrow"),
                          lab1=c(NA,          NA,    "Rate",      "Trend from"),
                          lab2=c(NA,          NA,    "2017-2021", "2012-2016"),
                          col1=c(NA,          NA,    "RateYr5",   "RateYr4"),
                          col2=c(NA,          NA,    NA,         "RateYr5")
                         )

micromapST(KYADDAllData, panelHC11, sortVar="RateYr5", ascend=FALSE,
            rowNames="ab", rowNamesCol="Abbr", bordDir=BordDir, bordGrp=bdGroup)
```

Even after aggregating the data to 15 ADDs, there are four ADDs with suppressed mortality rates, i.e., with fewer than 10 deaths. These ADDS are in eastern Kentucky, an area found to have many poor health outcomes and poor health behaviors in a recent assessment by the Kentucky Department for Public Health (Kentucky Department for Public Health, Cabinet for Health and Family Services, [August 31, 2023](#)). These four ADDs also have the smallest populations among the 15 ADDs and so it is not surprising that only a few deaths occurred in each area due to a rare cause.

There are clear geospatial clusters of ADDs with similar rates. The arrow plots indicate that the highest rate places have seen the greatest increase since the earlier period. Next, we plot the risk factor data at the ADD level.

```
panelHC12 <-
  data.frame(type=c("mapcum", "id", "dot",   "dot",           "dot", "dot"),
             lab1=c(NA,          NA,    "Rate", "Pct Current", "Pct Obese", "Pct Poverty", "Pct Screen"),
             lab2=c(NA,          NA,    "2017-2021", "Smoking",     NA,           NA,           NA),
             col1=c(NA,          NA,    "RateYr5",   "KYcursmk",   "KYobesity", "KYpoverty", "KYpoverty"),
             )

micromapST(KYADDAllData, panelHC12, sortVar="RateYr5", ascend=FALSE,
            rowNames="ab", rowNamesCol="Abbr", bordDir=BordDir, bordGrp=bdGroup)
```

The measures of the behavioral factors displayed in Figure 14.11 are based on survey samples of these small populations and so would be expected to have a high degree of uncertainty. We hesitate to draw any conclusions from the Kentucky linked micromap plots because of this small numbers problem, although we saw nothing at the ADD level in Kentucky that contradicts the associations noted at the U.S. state level. If we had the underlying data, we could attempt to stabilize these rates and factors by aggregating ADDs to even larger areas using the methods shown in Chapter 5.

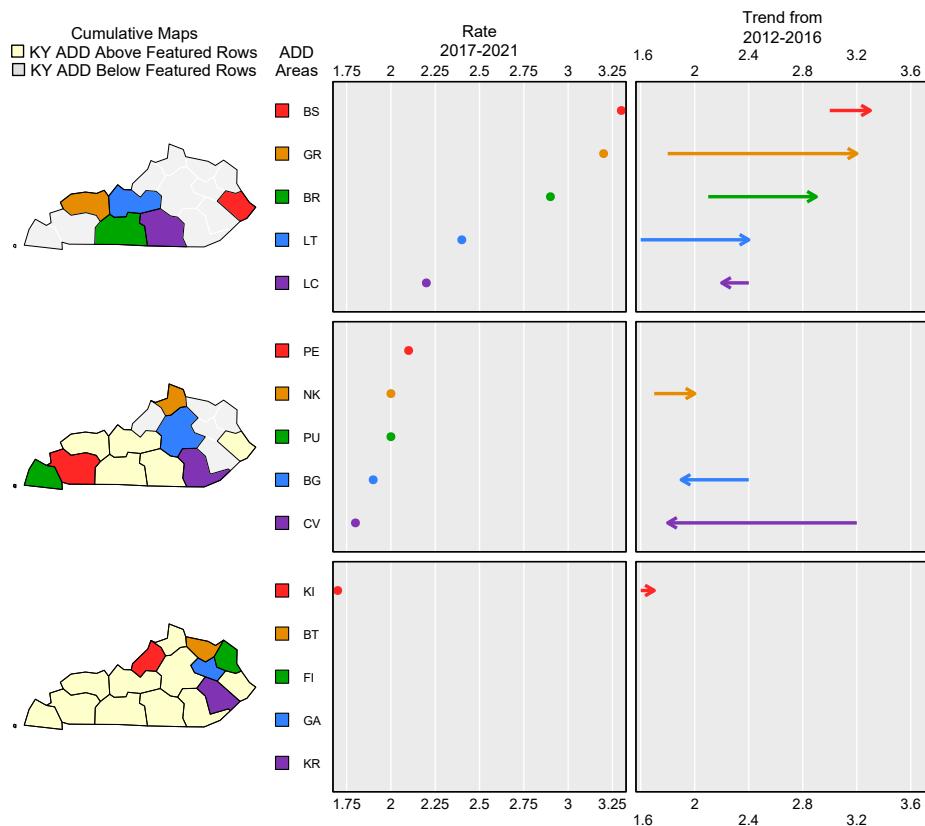


FIGURE 14.10 Rates and time trends of colorectal cancer mortality among whites ages <50 by Area Development Districts in Kentucky

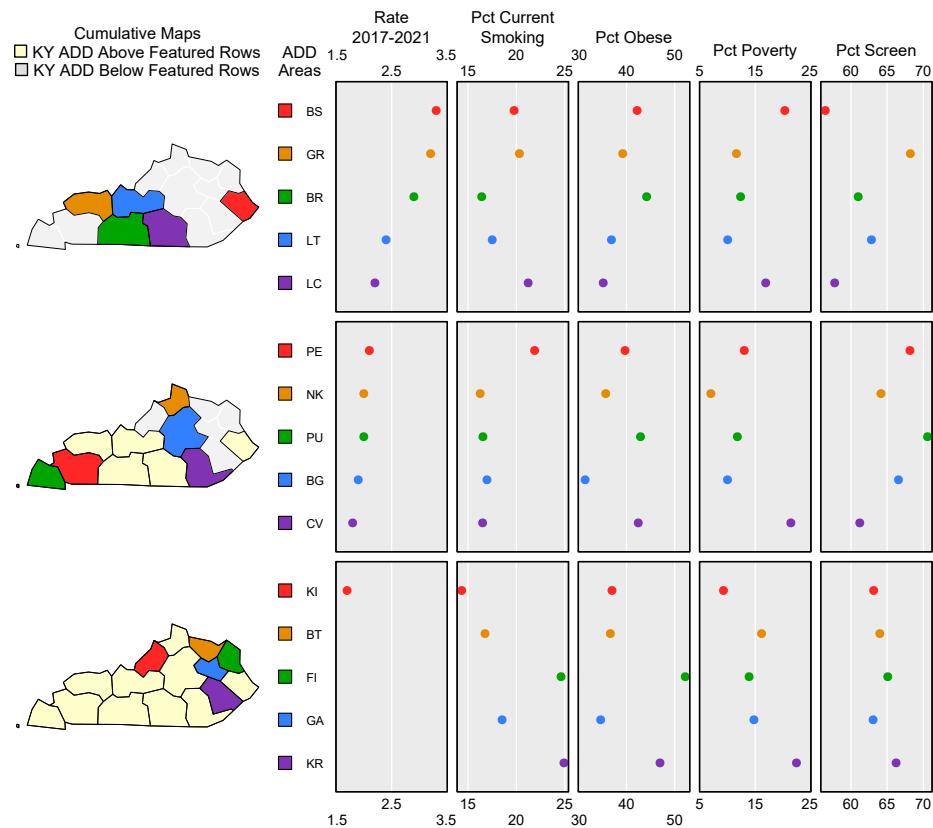


FIGURE 14.11 Rates and risk factors of colorectal cancer mortality among whites ages <50 by Area Development Districts in Kentucky

An aggregation of Kentucky counties into an Appalachian versus non-Appalachian region is often used for studies in that area (Appalachian Regional Commission, 2025).

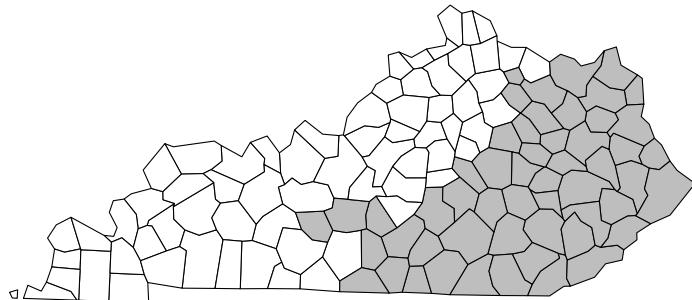


FIGURE 14.12 Appalachian counties in Kentucky

Previous research has shown significant differences between all-age colorectal cancer rates in Appalachian versus non-Appalachian counties in Kentucky (Hudson et al., 2024), with Appalachian incidence and mortality rates being 20% and 13% higher, respectively (Hudson et al., 2024). Thus, while our linked micromap plots were hampered by suppressed data due to small numbers of deaths in eastern Kentucky ADDs, results of the visual exploratory analysis are consistent with analyses at a broader geographic level. In addition, residents of Appalachian counties were more likely to be obese and to smoke and were less likely to have been screened for CRC (Hudson et al., 2024).

14.5 Summary and Further Reading

We have shown in this chapter how linked micromap plots might be used to explore and communicate health data. Specifically, we examined deaths due to colorectal cancer among persons less than age 50, rates that have been rising in recent years, unlike rapidly declining rates among persons 50 and older who are recommended for regular CRC screening. Despite the lack of risk factor data collected well before the cancer occurred, our exploratory analysis at the U.S. state level did find associations between the younger CRC rates and factors found to be associated with older CRC. Drilling down to the sub-state level, however, was less productive, due to suppressed data, but our results were consistent with analyses of similar data for broader geographic areas. The R code included in this chapter demonstrates how to apply typical steps to explore health rates not only for U.S. states but for sub-state areas defined by the user.

Readers interested in delving further into the study of public health patterns are referred to an epidemiology textbook, such as the classic Gordis Epidemiology (Celentano et al., 2024) or the more practical Essential Epidemiology in Public Health (Aschengrau, 2025). Carr's first application of linked micromaps was to environmental data (Carr, Olsen, Pierson, et al., 2000), while specific health applications such as those described in this chapter are shown in the Carr and Pickle book (Carr & Pickle, 2010) and in a summary article Pickle and Carr (2010).

References

- Appalachian Regional Commission. (2025). Local Development Districts [Web Page, <https://www.arc.gov/map/local-development-districts/>] (accessed June 20, 2025)].
- Aschengrau, A. (2025). *Aschengrau & Seage's Essentials of Epidemiology in Public Health (5th Edition)*. Jones & Bartlett Learning, Burlington, MA.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (2000). Using Linked Micromap Plots to Characterize Omernik Ecoregions [<https://doi.org/10.1023/A:1009828700017>]. *Data Mining and Knowledge Discovery*, 4(1), 43–67.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Celentano, D., Szklo, M., & Farag, Y. (2024). *Gordis Epidemiology (7th Edition)*. Elsevier, Philadelphia, PA.
- Centers for Disease Control and Prevention. (2025). CDC Wonder [Web Page, <https://wonder.cdc.gov/>] (accessed February 16, 2025)].
- Cleveland, W. S. (1981). LOWESS: A Program for Smoothing Scatterplots by Robust Locally Weighted Regression [<https://doi.org/10.2307/2683591>]. *The American Statistician*, 35(1), 54.
- Hudson, L., Burus, T., Park, L., Huang, B., Hull, P. C., & Vanderford, N. L. (2024). Cancer Disparities in Appalachian Kentucky [<https://doi.org/10.1111/jrh.12763>]. *The Journal of Rural Health*, 40(1), 87–95.
- Kentucky Council of Area Development Districts. (2025). KCADD [Web Page, <https://www.kcadd.org/>] (accessed May 27, 2025)].
- Kentucky Department for Public Health, Cabinet for Health and Family Services. (August 31, 2023). State Health Assessment Report, 2023 [Frankfort, Kentucky, <https://www.chfs.ky.gov/agencies/dph/Documents/StateHealthAssessment2023.pdf>].
- National Cancer Institute. (2025a). Colorectal Cancer Prevention (PDQ®)–Health Professional Version [Web Page, <https://www.cancer.gov/types/colorectal/hp/colorectal-prevention-pdq>] (accessed May 27, 2025)].
- National Cancer Institute. (2025b). State Cancer Profiles [Web Page, <https://statecancerprofiles.cancer.gov/>] (accessed March 4, 2025, and April 17, 2025)].
- National Cancer Institute (C. Phillips). (2025). As Rates of Some Cancers Increase in Younger People, Researchers Search for Answers [Web Page, <https://www.cancer.gov/news-events/cancer-currents-blog/2025/early-onset-cancer-research-environment-genetics-support>] (accessed June 9, 2025)].
- National Cancer Institute (NCI Staff). (2020). Why Is Colorectal Cancer Rising Rapidly among Young Adults? [Web Page, <https://www.cancer.gov/news-events/cancer-currents-blog/2020/colorectal-cancer-rising-younger-adults>] (accessed March 28, 2025)].
- Pickle, L. W., & Carr, D. B. (2010). Visualizing Health Data with Micromaps [<https://doi.org/10.1016/j.ssste.2010.03.007>]. *Spatial and Spatio-Temporal Epidemiology*, 1(2–3), 143–150.
- Siegel, R. L., Fedewa, S. A., Anderson, W. F., Miller, K. D., Ma, J., Rosenberg, P. S., & Jemal, A. (2017). Colorectal Cancer Incidence Patterns in the United States, 1974–2013 [<https://doi.org/10.1093/jnci/djw322>]. *JNCI: Journal of the National Cancer Institute*, 109(8), djw322.

- United States Census Bureau. (2025). American Community Survey (ACS) [Web Page, <https://www.census.gov/programs-surveys/acs.html> (accessed May 27, 2025)].

A

More to Say

Yeah! I have finished my book, but I have more to say about some topics. Let me explain them in this appendix.

To know more about **bookdown**, see <https://bookdown.org>.

A.1 Micromap Reference Overview

Here is an extended list of micromap references. A few more to come over the next few weeks. Most of the underlying articles, book chapters, posters, etc. have been made available via a Box folder now. You should have received an e-mail invitation to this Box folder on 3/25/2022. If you did not receive such an invitation or cannot access the files, please let me know.

This list should serve as a basic overview of what could be cited in various chapters of our book. You have to decide yourselves what is most suitable for your chapter. Of course, bring in additional references that are not listed yet.

If possible work with the bibtex file `referencesMicromaps.bib` as the basis for your citations. Revisit Section 1.11 how to do so (and see the examples below).

If you want to cite additional references, provide me with an initial bibtex entry, provide me with the DOI of the article or book chapter, point me to the web page, or provide me with any other way to construct a related bibtex entry.

- Ahn, Park, et al.: Ahn (2013), Ahn (2015), Ahn (2016), Ahn and Park (2016), Choi et al. (2014), Kim (2015), Park and Ahn (2013), Park and Ahn (2014), Park and Ahn (2015a), Park and Ahn (2015b), Park (2016) (App at <https://s2c-maps.shinyapps.io/home>),
- Carr, Pickle, Bell, et al.: Bell et al. (2006), Olsen et al. (1996), Carr and Pierson (1996), Carr, Olsen, Courbois, et al. (1998), Carr, Olsen, Pierson, et al. (1998), Carr, Olsen, Pierson, et al. (2000), Carr, Wallin, et al. (2000), Carr (2001), Carr (2002), Carr and Zhang (2002), Carr, Zhang, and Li (2002), Carr, Chen, et al. (2002), Carr et al. (2003), Carr (2005), Carr et al. (2005), Carr and Pickle (2010), Carr and Pearson Jr. (2015), Carr (2015), Chen et al. (2006), Fingerman (2010) (Book Review), Matthews (2013) (Book Review), Heim (2014), Kolvoord (2010) (Book Review), Lewis et al. (2017), Pearson Jr. and Carr (2016), Pearson Jr. and Carr (2022), Pickle and Carr (2010), Pickle et al. (2015), Ugarte (2012) (Book Review), Unwin (2011) (Book Review), Wang et al. (2002), C. Zhang (2012), Y. Zhang and Carr (2014), Zhu et al. (2016),
- Mast et al.: Mast (2013b), Mast (2013a), Mast (2014c), Mast (2014a), Mast (2014b), Mast (2015), Mast (2018), Mast (2020),
- McManus, Payton, Weber, et al.: Beck (2022) (Code at https://github.com/fawda123/micromap_app/tree/v2.0.0, App at https://beckmw.shinyapps.io/micromap_app/), Griffith (2014), McManus et al. (2016), Payton et al. (2012), Payton et al. (2013), Payton et al. (2015), Payton and Olsen (2015), Payton and Olsen (2021),
- Symanzik et al.: Chapala (2005), Gebreab et al. (2008), Gebreab (2010), Gebreab et al. (2015), Han et al. (2014), Han et al. (2016), Hurst et al. (2003), Jones and Symanzik (2001), Li and Symanzik (2016), Li (2017), Li and Symanzik (2017), Medri et al. (2019), Medri Cobos (2021), Probst and Symanzik (2019), Probst (2020), Symanzik et al. (2000), Symanzik, Axelrad, et al. (1999), Symanzik, Carr, Axelrad, et al. (1999), Symanzik and Jones (2001), Symanzik et al. (2002), Symanzik et al. (2003), Symanzik (2004), Symanzik and Carr (2008), Symanzik (2012), Symanzik and Carr (2013), Symanzik (2014), Symanzik et al. (2014), Symanzik et al. (2016), Symanzik, Carr, McManus, et al. (2017), Symanzik, Li, et al. (2017), Symanzik (2021), Thapliyal (2009), Voge and Symanzik (2011), Voge (2012), Yarra (2010),

- Others: asado23 (February 8, 2014), Baulier (August 18, 2011), Blunt (2006), Bonnal et al. (2011), Cairo (July 5, 2013), Das Gupta and Wong (2021), Dray and Jombart (2011), Ellis (April 30, 2017), Everitt (2021), Fonseca and Wong (2000), Friendly (2007), Kubota et al. (2015), Kolb (2015), Luo et al. (July 17 – August 7, 2017) (App at https://mandiluo.shinyapps.io/The_Indian_Story/), Nyadanu et al. (2019), Senkayi and Sattler (2013), Sahar et al. (2019), Silvanima et al. (2018), Sips et al. (2007), Slocum et al. (2023), Tatalovich and Stinchcomb (2019), Utzaut and Morgan (2010), Virginia Department of Environmental Quality (2020) (Code at <https://github.com/EmmaVJones/ProbDash/tree/master/app>, App at <https://evjones.shinyapps.io/FreshwaterProbMonEDA/>), Wartenberg (2009), Yu (2021),

R Core Team (2022)

Note: 126 micromap references found on Semantic Scholar on 3/23/2022 - see <https://www.semanticscholar.org/search?q=micromap&sort=relevance>. However, some of these references are related to MicroMAPS (Microprocessor-based Measurement of Air Pollution from Satellite).

References

- Ahn, J. Y. (2013). Visualizing Statistical Information Using Korean Linked Micromap Plots. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 219–221). Asian Regional Section of the IASC.
- Ahn, J. Y. (2015). Spatial Analysis of Air Pollution Data Based on Linked Micromap Plots in Korea [<https://doi.org/10.2991/cas-15.2015.42>]. *Proceedings of the 2015 AASRI International Conference on Circuits and Systems (CAS 2015)* (pp. 173–177). Atlantis Press, Dordrecht, The Netherlands.
- Ahn, J. Y. (2016). Micromap Plots to Visualize Air Pollution at National and Local Level in Korea [<https://doi.org/10.1080/00207233.2016.1148449>]. *International Journal of Environmental Studies*, 73(2), 277–285.
- Ahn, J. Y., & Park, S. J. (2016). Exploring Ground Level Ozone Distributions with Micromap Plots in Seoul of Korea [<https://doi.org/10.7763/IJCTE.2016.V8.1084>]. *International Journal of Computer Theory and Engineering*, 8(5), 429–433.
- asado23. (February 8, 2014). Shapefile to Produce a Linked Micromap in R [Web Page, <https://stackoverflow.com/questions/21651985/shapefile-to-produce-a-linked-micromap-in-r>].
- Baulier, J. (August 18, 2011). Creating Micromaps in JMP [Web Page, <https://community.jmp.com/t5/JMP-Blog/Creating-micromaps-in-JMP/ba-p/30019>].
- Beck, M. W. (2022). fawda123/micromap_app: V2.0.0 (Micromap Shiny Application Using State Data) [Zenodo Web Page, <https://doi.org/10.5281/zenodo.6532271>].
- Bell, S. B., Hoskins, R. E., Pickle, L. W., & Wartenberg, D. (2006). Current Practices in Spatial Analysis of Cancer Data: Mapping Health Statistics to Inform Policymakers and the Public [<https://doi.org/10.1186/1476-072X-5-49>]. *International Journal of Health Geographics*, 5, 49.
- Blunt, G. (2006). Using Grid Graphics to Produce Linked Micromap Plots of Large Financial Datasets [<https://www.r-project.org/conferences/useR-2006/Slides/Blunt.pdf>]. *Presentation, user! 2006, The R User Conference 2006, Vienna, Austria*. Austrian Association for Statistical Computing.

- Bonnal, L., Favard, P., Laurent, T., & Ruiz-Gazen, A. (2011). Pourquoi le coût de l'éducation est-il plus élevé en zone rurale? Le cas de la région Midi-Pyrénées [(In French), <https://doi.org/10.3917/reru.115.0887>]. *Revue d'Économie Régionale & Urbaine*, 2011/5, 887–910.
- Cairo, A. (July 5, 2013). Falling in Love with Micromaps [Web Page, <http://www.thefunctionalart.com/2013/07/falling-in-love-with-micromaps.html>].
- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B. (2002). Graphical Displays. In A. H. El-Shaarawi & W. W. Piegorsch (Eds.), *Encyclopedia of Environmetrics, Volume 2* (pp. 933–960). Wiley, Chichester, U.K.
- Carr, D. B. (2005). Some Recent Graphics Templates and Software for Showing Statistical Summaries [[https://doi.org/10.1016/S0169-7161\(04\)24015-9](https://doi.org/10.1016/S0169-7161(04)24015-9)]. In C. R. Rao, E. J. Wegman, & J. L. Solka (Eds.), *Handbook of Statistics, Vol. 24: Data Mining and Data Visualization* (pp. 415–436). North Holland, New York, NY.
- Carr, D. B. (2015). Exploratory Data Analysis [https://press.uchicago.edu/books/HO_C/HOC_V6/HOC_VOLUME6_E.pdf]. In M. Monmonier (Ed.), *The History of Cartography, Volume 6: Cartography in the Twentieth Century* (pp. 419–423). University of Chicago Press, Chicago, IL.
- Carr, D. B., Bell, B. S., Pickle, L. W., Zhang, Y., & Li, Y. (2003). The State Cancer Profiles Web Site and Extensions of Linked Micromap Plots and Conditioned Choropleth Map Plots [<https://dl.acm.org/doi/10.5555/1123196.1123307>]. *Proceedings of the Third National Conference on Digital Government Research* (pp. 269–273). Digital Government Research Center (DGRC).
- Carr, D. B., Chen, J., Bell, B. S., Pickle, L. W., & Zhang, Y. (2002). Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps [<https://dl.acm.org/doi/10.5555/1123098.1123147>]. *Proceedings of the Second National Conference on Digital Government Research* (pp. 61–67). Digital Government Research Center (DGRC).
- Carr, D. B., Olsen, A. R., Courbois, J.-Y. P., Pierson, S. M., & Carr, D. A. (1998). Linked Micromap Plots: Named and Described. *Statistical Computing and Statistical Graphics Newsletter*, 9(1), 24–32.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (1998). Boxplot Variations in a Spatial Context: An Omernik Ecoregion and Weather Example. *Statistical Computing and Statistical Graphics Newsletter*, 9(2), 4–13.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (2000). Using Linked Micromap Plots to Characterize Omernik Ecoregions [<https://doi.org/10.1023/A:1009828700017>]. *Data Mining and Knowledge Discovery*, 4(1), 43–67.
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Carr, D. B., & Pierson, S. M. (1996). Emphasizing Statistical Summaries and Showing Spatial Context with Micromaps. *Statistical Computing and Statistical Graphics Newsletter*, 7(3), 16–23.
- Carr, D. B., Wallin, J. F., & Carr, D. A. (2000). Two New Templates for Epidemiology Applications: Linked Micromap Plots and Conditioned Choropleth Maps [[https://doi.org/10.1002/1097-0258\(20000915/30\)19:17/18<2521::AID-SIM585>3.0.CO;2-K](https://doi.org/10.1002/1097-0258(20000915/30)19:17/18<2521::AID-SIM585>3.0.CO;2-K)]. *Statistics in Medicine*, 19(17–18), 2521–2538.

- Carr, D. B., White, D., & MacEachren, A. M. (2005). Conditioned Choropleth Maps and Hypothesis Generation [<https://doi.org/10.1111/j.1467-8306.2005.00449.x>]. *Annals of the Association of American Geographers*, 95(1), 32–53.
- Carr, D. B., & Zhang, Y. (2002). An Introduction to Dynamically Conditioned Choropleth Maps. *Survey Research Methods Section Newsletter*, 15(July), 2–5.
- Carr, D. B., Zhang, Y., & Li, Y. (2002). Dynamically Conditioned Choropleth Maps: Shareware for Hypothesis Generation and Education. *Statistical Computing and Statistical Graphics Newsletter*, 13(2), 2–7.
- Chapala, G. K. (2005). Development of Rich Features for Web-Based Interactive Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Chen, J. X., Carr, D. B., Wechsler, H., & Pan, Z. (2006). Interactive Visualization of Multivariate Statistical Data [<https://doi.org/10.20870/IJVR.2006.5.3.2701>]. *The International Journal of Virtual Reality*, 5(3), 67–73.
- Choi, S. H., Han, K. S., Park, S. J., & Ahn, J. Y. (2014). Visualizing Data with Linked and Comparative Micromap Plots. *International Conference on Artificial Intelligence and Industrial Application* (pp. 94–99). WIT Press, Southampton, U.K.
- Das Gupta, D., & Wong, D. (2021). How “Dependent” Are We? A Spatiotemporal Analysis of the Young and the Older Adult Populations in the US [<https://doi.org/10.1007/s11113-020-09590-y>]. *Population Research and Policy Review*, 40(6), 1221–1252.
- Dray, S., & Jombart, T. (2011). Revisiting Guerry’s Data: Introducing Spatial Constraints in Multivariate Analysis [<https://doi.org/10.1214/10-AOAS356>]. *The Annals of Applied Statistics*, 5(4), 2278–2299.
- Ellis, P. (April 30, 2017). Luke-Warm About Micromaps [Web Page, <http://freerangestats.info/blog/2017/04/30/micromaps>].
- Everitt, B. S. (2021). *Medical Statistics from A to Z: A Guide for Clinicians and Medical Students (Third Edition)* [<https://doi.org/10.1017/9781108919739>]. Cambridge University Press, Cambridge, U.K.
- Fingerman, S. (2010). Sci-Tech Book News Reviews: Visualizing Data Patterns with Micromaps [<https://jdc.jefferson.edu/scitechnews/vol64/iss4/14>]. *Sci-Tech News*, 64(4), 43–44.
- Fonseca, J. W., & Wong, D. W. (2000). Changing Patterns of Population Density in the United States [<https://doi.org/10.1111/0033-0124.00242>]. *The Professional Geographer*, 52(3), 504–517.
- Friendly, M. (2007). A.-M. Guerry’s Moral Statistics of France: Challenges for Multivariable Spatial Analysis [<https://doi.org/10.1214/07-STS241>]. *Statistical Science*, 22(3), 368–399.
- Gebreab, S. Y. (2010). *Spatial Epidemiology of Birth Defects in the United States and the State of Utah Using Geographic Information Systems and Spatial Statistics* (Doctoral dissertation) [<https://doi.org/10.26076/c49a-4f7f>]. Utah State University, Department of Watershed Sciences, Logan, Utah.
- Gebreab, S. Y., Davis, S. K., Symanzik, J., Mensah, G. A., Gibbons, G. H., & Diez-Roux, A. V. (2015). Geographic Variations in Cardiovascular Health in the United States: Contributions of State- and Individual-Level Factors [<https://doi.org/10.1161/JAHA.114.001673>]. *Journal of the American Heart Association*, 4(6), e001673.
- Gebreab, S. Y., Gillies, R. R., Munger, R. G., & Symanzik, J. (2008). Visualization and Interpretation of Birth Defects Data Using Linked Micromap Plots [<https://doi.org/10.1002/bdra.20419>]. *Birth Defects Research (Part A): Clinical and Molecular Teratology*, 82(2), 110–119.
- Griffith, M. B. (2014). Natural Variation and Current Reference for Specific Conductivity and Major Ions in Wadeable Streams of the Conterminous USA [<https://doi.org/10.1086/674704>]. *Freshwater Science*, 33(1), 1–17.

- Han, K. S., Park, S. J., Mun, G. S., Choi, S. H., Symanzik, J., Gebreab, S., & Ahn, J. Y. (2014). Linked Micromaps for the Visualization of Geographically Referenced Data. *ICIC Express Letters*, 8(2), 443–448.
- Han, K. S., Park, S. J., Symanzik, J., Choi, S. H., & Ahn, J. Y. (2016). Trends in Obesity at the National and Local Level among South Korean Adolescents [<https://doi.org/10.4081/gh.2016.381>]. *Geospatial Health*, 11(381), 130–136.
- Heim, K. (2014). *Visualization and Modeling for Crime Data Indexed by Road Segments* (Doctoral dissertation) [<https://hdl.handle.net/1920/8991>]. George Mason University, Statistical Science, Fairfax, VA.
- Hurst, J., Symanzik, J., & Gunter, L. (2003). Interactive Federal Statistical Data on the Web Using “nViZn” [(CD & <http://interfacesymposia.org/I03/I2003Proceedings/HurstJon/HurstJon.paper.pdf>)]. *Computing Science and Statistics*, 35.
- Jones, L., & Symanzik, J. (2001). Statistical Visualization of Environmental Data on the Web Using nViZn [(CD & <http://interfacesymposia.org/I01/I2001Proceedings/LJones/LJones.pdf>)]. *Computing Science and Statistics*, 33.
- Kim, M. (2015). Linked Micromap: Exploratory Data Analysis and Geographic Visualization of Spatial Statistics Data [(In Korean), <https://doi.org/10.16879/jkca.2015.15.2.039>]. *Journal of the Korean Cartographic Association*, 15(2), 39–50.
- Kolb, J.-P. (2015). Visualisation of Macroeconomic Indicators in Maps with R [https://eropa.eu/eurostat/cros/system/files/NTTS15_Kolb.pdf]. *Presentation, New Techniques and Technologies for Statistics (NTTS) 2015*. European Commission, Collaboration in Research; Methodology for Official Statistics (CROS).
- Kolvoord, R. A. (2010). Book Review: Visualizing Data Patterns with Micromaps [<http://choicereviews.org/review/10.5860/CHOICE.48-0920>]. *Choice Reviews*, 48(2), 48–0920.
- Kubota, T., Iizuka, M., & Tsubaki, H. (2015). Visualization of Spatial and Paneled Data for Reason-Specified Suicide Data by Prefecture in Japan [<https://2015.isiproceedings.org/Files/CPS415-P1-S.pdf>]. *Proceedings of the 60th World Statistics Congress of the International Statistical Institute, Rio de Janeiro, Brazil, 26-31 July 2015*. International Statistical Institute, The Hague, The Netherlands.
- Lewis, D. R., Pickle, L. W., & Zhu, L. (2017). Recent Spatiotemporal Patterns of US Lung Cancer by Histologic Type [<https://doi.org/10.3389/fpubh.2017.00082>]. *Frontiers in Public Health*, 5, 82.
- Li, C. (2017). *Extracting and Visualizing Data from Mobile and Static Eye Trackers in R and Matlab* (Doctoral dissertation) [<https://doi.org/10.26076/5c8c-d8a5>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Li, C., & Symanzik, J. (2016). The Linked Microposter Plot as a New Means for the Visualization of Eye Tracking Data [(CD)]. *2016 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Li, C., & Symanzik, J. (2017). EyeTrackR: An R Package for Extracting and Visualizing Data from Mobile and Static Eye Trackers [(CD)]. *2018 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Luo, M., Majumdar, P., & Vasudeva Rao, S. (July 17 – August 7, 2017). The Indian Story [Web Page, https://wiki.smu.edu.sg/1617t3isss608g1/The_Indian_Story].
- Mast, B. D. (2013a). Exploring Housing Cost Data with Conditioned Choropleth Maps [<https://www.huduser.gov/portal/periodicals/cityscape/vol15num3/ch18.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 15(3), 251–255.
- Mast, B. D. (2013b). Visualizing Same-Sex Couple Household Data with Linked Micromaps [<https://www.huduser.gov/portal/periodicals/cityscape/vol15num2/ch23.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 15(2), 267–271.

- Mast, B. D. (2014a). Comparative Micromaps and Changing State Homeownership Rates [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num2/ch12.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(2), 163–167.
- Mast, B. D. (2014b). Mapping White–Black and Temporal Differences in State Homeownership Rates with Two-Way Comparative Micromaps [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num3/ch7.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(3), 149–152.
- Mast, B. D. (2014c). Measuring Spatial Mismatch between Homelessness and Homeless Resources with a Theil Index and Statistical Inference [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num1/ch21.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(1), 339–350.
- Mast, B. D. (2015). Measuring Neighborhood Opportunity with AFFH Data [<https://www.huduser.gov/portal/periodicals/cityscape/vol17num3/ch12.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 17(3), 221–230.
- Mast, B. D. (2018). School Performance of Schools Assigned to HUD-Assisted Households [<https://www.huduser.gov/portal/periodicals/cityscape/vol20num3/ch10.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 20(3), 189–222.
- Mast, B. D. (2020). Measuring Homelessness and Resources to Combat Homelessness with PIT and HIC Data [<https://www.huduser.gov/portal/periodicals/cityscape/vol22num1/ch7.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 22(1), 215–225.
- Matthews, S. A. (2013). Book Review: Visualizing Data Patterns with Micromaps [<https://doi.org/10.1007/BF03354893>]. *Spatial Demography*, 1(1), 141–143.
- McManus, M. G., Pond, G. J., Reynolds, L., & Griffith, M. B. (2016). Multivariate Condition Assessment of Watersheds with Linked Micromaps [<https://doi.org/10.1111/1752-1688.12399>]. *Journal of the American Water Resources Association*, 52(2), 494–507.
- Medri, J., Probst, B. D., & Symanzik, J. (2019). Housing Affordability and Immigration: An Exploratory Analysis in New York City [(CD)]. *2019 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Medri Cobos, J. (2021). *Housing Variables and Immigration: An Exploratory and Predictive Data Analysis in New York City* (Master's thesis) [<https://doi.org/10.26076/dd0e-699c>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Nyadanu, S. D., Pereira, G., Nawumbeni, D. N., & Adampah, T. (2019). Geo–Visual Integration of Health Outcomes and Risk Factors Using Excess Risk and Conditioned Choropleth Maps: A Case Study of Malaria Incidence and Sociodemographic Determinants in Ghana [<https://doi.org/10.1186/s12889-019-6816-z>]. *BMC Public Health*, 19, 514.
- Olsen, A. R., Carr, D. B., Courbois, J.-Y. P., & Pierson, S. M. (1996). Presentation of Data in Linked Attribute and Geographic Space. *1996 Abstracts, Joint Statistical Meetings, Chicago, Illinois* (p. 271). American Statistical Association, Alexandria, VA.
- Park, S. J. (2016). *Visualizing Geospatial Data with Statistical Graphs on Google Maps and Micromaps* (Doctoral dissertation). Chonbuk National University, Jeonju, South Korea.
- Park, S. J., & Ahn, J. Y. (2013). Visualizing Statistical Data Using Linked Micromap Plots [(In Korean)]. *Journal of The Korean Official Statistics*, 18(2), 111–127.
- Park, S. J., & Ahn, J. Y. (2014). Visualization and Interpretation of Cancer Data Using Linked Micromap Plots [<https://doi.org/10.7465/jkdi.2014.25.6.1531>]. *Journal of the Korean Data and Information Science Society*, 25(6), 1531–1538.

- Park, S. J., & Ahn, J. Y. (2015a). Analyzing the Relationship between Standardized Mortality Ratio and Health Indicators Using Linked Micromap Plots. *ICIC Express Letters, 9*(1), 1–7.
- Park, S. J., & Ahn, J. Y. (2015b). Spatial Visualization of Climatic Data Using Micromap Plots in Korea. *ICIC Express Letters, Part B: Applications, 6*(8), 2059–2064.
- Payton, Q. C., McManus, M. G., Weber, M. H., Olsen, A. R., & Kincaid, T. M. (2015). micromap: A Package for Linked Micromaps [<https://doi.org/10.18637/jss.v063.i02>]. *Journal of Statistical Software, 63*(2), 1–16.
- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2021). *micromap: Linked Micromap Plots* [R package version 1.9.5 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., Weber, M. H., McManus, M. G., Kincaid, T. M., & Olsen, A. R. (2013). Linked Micromaps: Statistical Summaries in a Spatial Context Using the micromap R Package [<https://www.usiale.org/austin2013/presentation-details/4693>]. *Landscape Dynamics Along Environmental Gradients, 2013 Annual Symposium, April 14–18, 2013, Austin, TX*. US Regional Association of the International Association for Landscape Ecology (US–IALE).
- Payton, Q. C., Weber, M. H., McManus, M. G., & Olsen, A. R. (2012). Linked Micromaps: Statistical Summaries in a Spatial Context [https://acwi.gov/monitoring/conference/2012/posters/posters_2012_conference.pdf]. *Water: One Resource — Shared Effort — Common Future, 8th National Monitoring Conference, April 30–May 4, 2012, Portland, Oregon*. National Water Quality Monitoring Council.
- Pearson Jr., J. B., & Carr, D. B. (2016). *micromapST: Linked Micromap Plots for General U. S. and Other Geographic Areas* [R package version 1.1.1 (<http://CRAN.R-project.org/package=micromapST>)].
- Pearson Jr., J. B., & Carr, D. B. (2022). *micromapST: Linked Micromap Plots for U. S. and Other Geographic Areas* [R package version 1.1.3 (<http://CRAN.R-project.org/package=micromapST>)].
- Pickle, L. W., & Carr, D. B. (2010). Visualizing Health Data with Micromaps [<https://doi.org/10.1016/j.sste.2010.03.007>]. *Spatial and Spatio-Temporal Epidemiology, 1*(2–3), 143–150.
- Pickle, L. W., Pearson Jr., J. B., & Carr, D. B. (2015). micromapST: Exploring and Communicating Geospatial Patterns in US State Data [<https://doi.org/10.18637/jss.v063.i03>]. *Journal of Statistical Software, 63*(3), 1–25.
- Probst, B. D. (2020). ‘LMshapemaker’: Utilizing the ‘Rmapshaper’ R Package to Modify Shapefiles for Use in Linked Micromap Plots (Master’s thesis) [<https://doi.org/10.26076/j9yj-mm66>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Probst, B. D., & Symanzik, J. (2019). Using ‘rmapshaper’ to Modify Boundary Files for Use in Linked Micromap Plots [[https://2019.isiproceedings.org/Files/10.Contributed-Paper-Session\(CPS\)-Volume-4.pdf](https://2019.isiproceedings.org/Files/10.Contributed-Paper-Session(CPS)-Volume-4.pdf)]. *Proceeding, Contributed Paper Session Volume 4, 62nd ISI World Statistics Congress 2019, 18–23 August 2019, Kuala Lumpur, Malaysia* (pp. 281–289). Department of Statistics Malaysia, Putrajaya, Malaysia.
- R Core Team. (2022). *R: A Language and Environment for Statistical Computing* [<http://www.R-project.org/>]. R Foundation for Statistical Computing. Vienna, Austria.
- Sahar, L., Foster, S. L., Sherman, R. L., Henry, K. A., Goldberg, D. W., Stinchcomb, D. G., & Bauer, J. E. (2019). GIScience and Cancer: State of the Art and Trends for Cancer Surveillance and Epidemiology [<https://doi.org/10.1002/cncr.32052>]. *Cancer, 125*(15), 2544–2560.

- Senkayi, S. N., & Sattler, M. L. (2013). Varying Spatial Levels in GIS Analysis Environmental Epidemiological Data in Texas [https://www.cc.gatech.edu/gvu/ii/PublicHealthVis/Papers/VaryingSpatialLevelsGIS-EnvirnomentalEPI_F.pdf]. *Public Health's Wicked Problems: Can Info Vis Save Lives?, Workshop, October 13, 2013, Atlanta, GA, USA*.
- Silvanima, J., Woeber, A., Sunderman-Barnes, S., Copeland, R., Sedlacek, C., & Seal, T. (2018). A Synoptic Survey of Select Wastewater-Tracer Compounds and the Pesticide Imidacloprid in Florida's Ambient Freshwaters [<https://doi.org/10.1007/s10661-018-6782-4>]. *Environmental Monitoring and Assessment*, 190(7), 435.
- Sips, M., Schneidewind, J., & Keim, D. A. (2007). Highlighting Space–Time Patterns: Effective Visual Encodings for Interactive Decision–Making [<https://doi.org/10.1080/13658810701362147>]. *International Journal of Geographical Information Science*, 21(8), 879–893.
- Slocum, T. A., McMaster, R. B., Kessler, F. C., & Howard, H. H. (2023). Data Exploration [<https://doi.org/10.1201/9781003150527-28>]. In T. A. Slocum, R. B. McMaster, F. C. Kessler, & H. H. Howard (Eds.), *Thematic Cartography and Geovisualization (Fourth Edition)* (Online). CRC Press, Boca Raton, FL.
- Symanzik, J. (2004). Interactive and Dynamic Graphics. In J. E. Gentle, W. Härdle, & Y. Mori (Eds.), *Handbook of Computational Statistics — Concepts and Methods* (pp. 293–336). Springer, Berlin, Heidelberg.
- Symanzik, J. (2012). Interactive and Dynamic Graphics [https://doi.org/10.1007/978-3-642-21551-3_12]. In J. E. Gentle, W. K. Härdle, & Y. Mori (Eds.), *Handbook of Computational Statistics, Volume 1 — Concepts and Methods (Second Edition)* (pp. 335–373). Springer, Berlin, Heidelberg.
- Symanzik, J. (2014). Exploratory Spatial Data Analysis [https://doi.org/10.1007/978-3-642-23430-9_76]. In M. M. Fischer & P. Nijkamp (Eds.), *Handbook of Regional Science* (pp. 1295–1310). Springer, Berlin, Heidelberg.
- Symanzik, J. (2021). Exploratory Spatial Data Analysis [https://doi.org/10.1007/978-3-662-60723-7_76]. In M. M. Fischer & P. Nijkamp (Eds.), *Handbook of Regional Science (Second and Extended Edition)* (pp. 1845–1861). Springer, Berlin, Heidelberg.
- Symanzik, J., Axelrad, D. A., Carr, D. B., Wang, J., Wong, D., & Woodruff, T. J. (1999). HAPs, Micromaps and GPL — Visualization of Geographically Referenced Statistical Summaries on the World Wide Web. *Annual Proceedings (ACSM–WFPS–PLSO–LSAW 1999 Conference CD)*. American Congress on Surveying & Mapping.
- Symanzik, J., Bao, S., Dai, X., Shui, M., & She, B. (2016). Recent Advancements in Geovisualization, with a Case Study on Chinese Religions [https://doi.org/10.1007/978-3-319-42571-9_8]. In Z. Jin, M. Liu, & X. Luo (Eds.), *New Developments in Statistical Modeling, Inference and Application: Selected Papers from the 2014 ICSA/KISS Joint Applied Statistics Symposium in Portland, OR* (pp. 151–166). Springer, Cham, Switzerland.
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.
- Symanzik, J., & Carr, D. B. (2013). Linked Micromap Plots in R. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 213–218). Asian Regional Section of the IASC.
- Symanzik, J., Carr, D. B., Axelrad, D. A., Wang, J., Wong, D., & Woodruff, T. J. (1999). Interactive Tables and Maps — A Glance at EPA's Cumulative Exposure Project Web Page. *1999 Proceedings of the Section on Statistical Graphics* (pp. 94–99). American Statistical Association, Alexandria, VA.

- Symanzik, J., Carr, D. B., McManus, M. G., & Weber, M. H. (2017). Micromaps [<https://doi.org/10.1002/9781118445112.stat07938>]. *Wiley StatsRef: Statistics Reference Online*. Wiley Online Library.
- Symanzik, J., Dai, X., Weber, M. H., Payton, Q., & McManus, M. G. (2014). Linked Micromap Plots for South America — General Design Considerations and Specific Adjustments [<https://doi.org/10.15446/rce.v37n2spe.47949>]. *Revista Colombiana de Estadística*, 37(2), 451–469.
- Symanzik, J., Gebreab, S., Gillies, R., & Wilson, J. (2003). Visualizing the Spread of West Nile Virus [(CD)]. *2003 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Hurst, J., & Gunter, L. (2002). Recent Developments for Interactive Statistical Graphics on the Web Using “nViZn” [(CD)]. *2002 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., & Jones, L. (2001). “nViZn” Federal Statistical Data on the Web [(CD)]. *2001 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Li, C., Zhang, B., Studenka, B., & McKinnney, E. (2017). Eye—Tracking in Practice: A First Analysis of a Study on Human Postures [(CD)]. *2017 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Wong, D., Wang, J., Carr, D. B., Woodruff, T. J., & Axelrad, D. A. (2000). Web-based Access and Visualization of Hazardous Air Pollutants [<https://webharvest.gov/peth04/20041025104701/http://www.atsdr.cdc.gov/gis/conference98/proceedings/pdf/symanzik.pdf>]. *Geographic Information Systems in Public Health: Proceedings of the Third National Conference, August 18–20, 1998, San Diego, California* (pp. 235–248). Agency for Toxic Substances & Disease Registry.
- Tatalovich, Z., & Stinchcomb, D. G. (2019). Creating Maps and Mapping Systems for Cancer Control and Prevention [https://doi.org/10.1007/978-3-030-18408-7_3]. In D. Berrigan & N. A. Berger (Eds.), *Geospatial Approaches to Energy Balance and Breast Cancer* (pp. 59–79). Springer Nature, Cham, Switzerland.
- Thapliyal, A. (2009). Enhancement of Web-based Interactive Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Ugarte, M. D. (2012). Book Review: Visualizing Data Patterns with Micromaps [https://doi.org/10.1111/j.1467-985X.2012.01069_3.x]. *Journal of the Royal Statistical Society, Series A: Statistics in Society*, 175(4), 1072–1073.
- Unwin, A. (2011). Short Book Review: Visualizing Data Patterns with Micromaps [https://doi.org/10.1111/j.1751-5823.2011.00134_14.x]. *International Statistical Review*, 79(1), 127–128.
- Utlaut, T., & Morgan, G. (2010). On Enhancing JMP’s Visual Analytics Using JMP Scripting Language (JSL) [<https://community.jmp.com/t5/Discovery-Summit-2010/On-Enhancing-JMP-s-Visual-Analytics-Using-JMP-Scripting-Language/ta-p/40562>]. *JMP User Community Discovery Summit 2010*. JMP Statistical Discovery LLC.
- Virginia Department of Environmental Quality. (2020). Chapter 4.4 Freshwater Probabilistic Monitoring Results [<https://www.deq.virginia.gov/home/showpublisheddocument/2221/637436316328230000>]. *2020 305(b)/303(d) Water Quality Assessment Integrated Report, EPA Approved, December 9, 2020* (pp. 112–152). Virginia DEQ, Richmond, VA.
- Voge, N. D. (2012). Ignoring the Spatial Context in Intro Statistics Classes — And Some Simple Graphical Remedies [MS Report, Utah State University, Department of Mathematics and Statistics, Logan, UT, <https://doi.org/10.26076/663e-4af7>].
- Voge, N. D., & Symanzik, J. (2011). Ignoring the Spatial Context in Intro Stats Classes — And Some Simple Graphical Remedies [(CD)]. *2011 JSM Proceedings*. American Statistical Association, Alexandria, VA.

- Wang, X., Chen, J. X., Carr, D. B., Bell, B. S., & Pickle, L. W. (2002). Geographic Statistics Visualization: Web-based Linked Micromap Plots [<https://doi.org/10.1109/5992.998645>]. *Computing in Science & Engineering*, 4(3), 90–94.
- Wartenberg, D. (2009). Some Considerations for the Communication of Results of Air Pollution Health Effects Tracking [<https://doi.org/10.1007/s11869-009-0046-y>]. *Air Quality, Atmosphere & Health*, 2(4), 207–221.
- Yarra, P. K. (2010). Refactoring Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Yu, B. (2021). Real-World Evidence from Population-Based Cancer Registry Data [<http://doi.org/10.1201/9780429398674-3>]. In H. Yang & B. Yu (Eds.), *Real-World Evidence in Drug Development and Evaluation* (pp. 47–70). Chapman & Hall / CRC, Boca Raton, FL.
- Zhang, C. (2012). *Interfaces and Visual Analytics for Visualizing Spatio-Temporal Data with Micromaps* (Doctoral dissertation) [<https://hdl.handle.net/1920/7888>]. George Mason University, Computational Sciences and Informatics, Fairfax, VA.
- Zhang, Y., & Carr, D. B. (2014). Providing Overviews of China Datasets Using Linked Micromaps [<https://doi.org/10.1109/COM.Geo.2014.20>]. *2014 Fifth International Conference on Computing for Geospatial Research and Application, Washington, DC, USA* (p. 123). IEEE.
- Zhu, L., Pickle, L. W., & Pearson Jr., J. B. (2016). Confidence Intervals for Rate Ratios between Geographic Units [<https://doi.org/10.1186/s12942-016-0073-5>]. *International Journal of Health Geographics*, 15, 44.

Bibliography

- Ahn, J. Y. (2013). Visualizing Statistical Information Using Korean Linked Micromap Plots. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 219–221). Asian Regional Section of the IASC.
- Ahn, J. Y. (2015). Spatial Analysis of Air Pollution Data Based on Linked Micromap Plots in Korea [<https://doi.org/10.2991/cas-15.2015.42>]. *Proceedings of the 2015 AASRI International Conference on Circuits and Systems (CAS 2015)* (pp. 173–177). Atlantis Press, Dordrecht, The Netherlands.
- Ahn, J. Y. (2016). Micromap Plots to Visualize Air Pollution at National and Local Level in Korea [<https://doi.org/10.1080/00207233.2016.1148449>]. *International Journal of Environmental Studies*, 73(2), 277–285.
- Ahn, J. Y., & Park, S. J. (2016). Exploring Ground Level Ozone Distributions with Micromap Plots in Seoul of Korea [<https://doi.org/10.7763/IJCTE.2016.V8.1084>]. *International Journal of Computer Theory and Engineering*, 8(5), 429–433.
- Appalachian Regional Commission. (2025). Local Development Districts [Web Page, <https://www.arc.gov/map/local-development-districts/> (accessed June 20, 2025)].
- asado23. (February 8, 2014). Shapefile to Produce a Linked Micromap in R [Web Page, <https://stackoverflow.com/questions/21651985/shapefile-to-produce-a-linked-micromap-in-r>].
- Aschengrau, A. (2025). *Aschengrau & Seage's Essentials of Epidemiology in Public Health (5th Edition)*. Jones & Bartlett Learning, Burlington, MA.
- Atkins, A., McPherson, J., & Allaire, J. J. (2021). *rsconnect: Deployment Interface for R Markdown Documents and Shiny Applications* [R package version 0.8.25 (<https://CRAN.R-project.org/package=rsconnect>)].
- Barthelme, S. (2022). *imager: Image Processing Library Based on 'CImg'* [R package version 0.42.13 (<http://CRAN.R-project.org/package=imager>)].
- Baulier, J. (August 18, 2011). Creating Micromaps in JMP [Web Page, <https://community.jmp.com/t5/JMP-Blog/Creating-micromaps-in-JMP/ba-p/30019>].
- Beaulieu, J. J., McManus, M. G., & Nietch, C. T. (2016). Estimates of Reservoir Methane Emissions Based on a Spatially Balanced Probabilistic-Survey [<https://doi.org/10.1002/lno.10284>]. *Limnology and Oceanography*, 61(S1), S27–S40.
- Beck, M. W. (2022). *fawda123/micromap_app*: V2.0.0 (Micromap Shiny Application Using State Data) [Zenodo Web Page, <https://doi.org/10.5281/zenodo.6532271>].
- Beecham, R., Dykes, J., Meulemans, W., Slingsby, A., Turkay, C., & Wood, J. (2017). Map LineUps: Effects of Spatial Structure on Graphical Inference [<https://doi.org/10.1109/TVCG.2016.2598862>]. *IEEE Transactions on Visualization and Computer Graphics*, 23(1), 391–400.
- Bell, S. B., Hoskins, R. E., Pickle, L. W., & Wartenberg, D. (2006). Current Practices in Spatial Analysis of Cancer Data: Mapping Health Statistics to Inform Policymakers and the Public [<https://doi.org/10.1186/1476-072X-5-49>]. *International Journal of Health Geographics*, 5, 49.

- Bivand, R. S. (2022). R Packages for Analyzing Spatial Data: A Comparative Case Study with Areal Data [<https://doi.org/10.1111/gean.12319>]. *Geographical Analysis*, 54(3), 488–518.
- Bivand, R. S., Pebesma, E., & Gómez-Rubio, V. (2013). *Applied Spatial Data Analysis with R, Second Edition* [<https://doi.org/10.1007/978-1-4614-7618-4> and <https://asdar-book.org/>]. Springer, New York, NY.
- Blunt, G. (2006). Using Grid Graphics to Produce Linked Micromap Plots of Large Financial Datasets [<https://www.r-project.org/conferences/useR-2006/Slides/Blunt.pdf>]. *Presentation, userR! 2006, The R User Conference 2006, Vienna, Austria*. Austrian Association for Statistical Computing.
- Bonnal, L., Favard, P., Laurent, T., & Ruiz-Gazen, A. (2011). Pourquoi le coût de l'éducation est-il plus élevé en zone rurale? Le cas de la région Midi-Pyrénées [(In French), <https://doi.org/10.3917/reu.115.0887>]. *Revue d'Économie Régionale & Urbaine*, 2011/5, 887–910.
- Brewer, C. A., Hatchard, G. W., & Harrower, M. A. (2003). ColorBrewer in Print: A Catalog of Color Schemes for Maps [<https://doi.org/10.1559/152304003100010929>]. *Cartography and Geographic Information Science*, 30(1), 5–32.
- Brewer, C. A., & Pickle, L. W. (2002). Comparison of Methods for Classifying Epidemiological Data on Choropleth Maps in Series. *Annals of the Association of American Geographers*, 92(4), 662–681.
- Cairo, A. (July 5, 2013). Falling in Love with Micromaps [Web Page, <http://www.thefunctionalart.com/2013/07/falling-in-love-with-micromaps.html>].
- Carr, D. B. (1994). *Converting Tables to Plots* (tech. rep. No. 101). Center for Computational Statistics, George Mason University, Fairfax, VA.
- Carr, D. B. (2001). Designing Linked Micromap Plots for States with Many Counties [<https://doi.org/10.1002/sim.670>]. *Statistics in Medicine*, 20(9–10), 1331–1339.
- Carr, D. B. (2002). Graphical Displays. In A. H. El-Shaarawi & W. W. Piegorsch (Eds.), *Encyclopedia of Environmetrics, Volume 2* (pp. 933–960). Wiley, Chichester, U.K.
- Carr, D. B. (2005). Some Recent Graphics Templates and Software for Showing Statistical Summaries [[https://doi.org/10.1016/S0169-7161\(04\)24015-9](https://doi.org/10.1016/S0169-7161(04)24015-9)]. In C. R. Rao, E. J. Wegman, & J. L. Solka (Eds.), *Handbook of Statistics, Vol. 24: Data Mining and Data Visualization* (pp. 415–436). North Holland, New York, NY.
- Carr, D. B. (2015). Exploratory Data Analysis [https://press.uchicago.edu/books/HO_C/HOC_V6/HOC_VOLUME6_E.pdf]. In M. Monmonier (Ed.), *The History of Cartography, Volume 6: Cartography in the Twentieth Century* (pp. 419–423). University of Chicago Press, Chicago, IL.
- Carr, D. B., Bell, B. S., Pickle, L. W., Zhang, Y., & Li, Y. (2003). The State Cancer Profiles Web Site and Extensions of Linked Micromap Plots and Conditioned Choropleth Map Plots [<https://dl.acm.org/doi/10.5555/1123196.1123307>]. *Proceedings of the Third National Conference on Digital Government Research* (pp. 269–273). Digital Government Research Center (DGRC).
- Carr, D. B., Chen, J., Bell, B. S., Pickle, L. W., & Zhang, Y. (2002). Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps [<https://dl.acm.org/doi/10.5555/1123098.1123147>]. *Proceedings of the Second National Conference on Digital Government Research* (pp. 61–67). Digital Government Research Center (DGRC).
- Carr, D. B., & Nusser, S. M. (1995). Converting Tables to Plots: A Challenge from Iowa State. *Statistical Computing and Statistical Graphics Newsletter*, 6(3), 11–18.
- Carr, D. B., Olsen, A. R., Courbois, J.-Y. P., Pierson, S. M., & Carr, D. A. (1998). Linked Micromap Plots: Named and Described. *Statistical Computing and Statistical Graphics Newsletter*, 9(1), 24–32.

- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (1998). Boxplot Variations in a Spatial Context: An Omernik Ecoregion and Weather Example. *Statistical Computing and Statistical Graphics Newsletter*, 9(2), 4–13.
- Carr, D. B., Olsen, A. R., Pierson, S. M., & Courbois, J.-Y. P. (2000). Using Linked Micromap Plots to Characterize Omernik Ecoregions [<https://doi.org/10.1023/A:1009828700017>]. *Data Mining and Knowledge Discovery*, 4(1), 43–67.
- Carr, D. B., & Pearson Jr., J. B. (2013). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pearson Jr., J. B. (2015). *micromapST: Linked Micromap Plots for U. S. States* [R package version 1.0.5 (<http://CRAN.R-project.org/package=micromapST>)].
- Carr, D. B., & Pickle, L. W. (2010). *Visualizing Data Patterns with Micromaps*. Chapman & Hall / CRC, Boca Raton, FL.
- Carr, D. B., & Pierson, S. M. (1996). Emphasizing Statistical Summaries and Showing Spatial Context with Micromaps. *Statistical Computing and Statistical Graphics Newsletter*, 7(3), 16–23.
- Carr, D. B., Valliant, R., & Rope, D. J. (1996). Plot Interpretation and Information Webs: A Time-Series Example from the Bureau of Labor Statistics. *Statistical Computing and Statistical Graphics Newsletter*, 7(2), 19–26.
- Carr, D. B., Wallin, J. F., & Carr, D. A. (2000). Two New Templates for Epidemiology Applications: Linked Micromap Plots and Conditioned Choropleth Maps [[https://doi.org/10.1002/1097-0258\(20000915/30\)19:17/18<2521::AID-SIM585>3.0.CO;2-K](https://doi.org/10.1002/1097-0258(20000915/30)19:17/18<2521::AID-SIM585>3.0.CO;2-K)]. *Statistics in Medicine*, 19(17–18), 2521–2538.
- Carr, D. B., White, D., & MacEachren, A. M. (2005). Conditioned Choropleth Maps and Hypothesis Generation [<https://doi.org/10.1111/j.1467-8306.2005.00449.x>]. *Annals of the Association of American Geographers*, 95(1), 32–53.
- Carr, D. B., & Zhang, Y. (2002). An Introduction to Dynamically Conditioned Choropleth Maps. *Survey Research Methods Section Newsletter*, 15(July), 2–5.
- Carr, D. B., Zhang, Y., & Li, Y. (2002). Dynamically Conditioned Choropleth Maps: Shareware for Hypothesis Generation and Education. *Statistical Computing and Statistical Graphics Newsletter*, 13(2), 2–7.
- Celentano, D., Szklo, M., & Farag, Y. (2024). *Gordis Epidemiology (7th Edition)*. Elsevier, Philadelphia, PA.
- Centers for Disease Control and Prevention. (2025). CDC Wonder [Web Page, <https://wonder.cdc.gov/> (accessed February 16, 2025)].
- Chang, W. (2021). *shinythemes: Themes for Shiny* [R package version 1.2.0 (<https://CRAN.R-project.org/package=shinythemes>)].
- Chang, W., & Borges Ribeiro, B. (2021). *shinydashboard: Create Dashboards with 'Shiny'* [R package version 0.7.2 (<https://CRAN.R-project.org/package=shinydashboard>)].
- Chang, W., Cheng, J., Allaire, J. J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., & Borges, B. (2021). *shiny: Web Application Framework for R* [R package version 1.7.1 (<https://CRAN.R-project.org/package=shiny>)].
- Chapala, G. K. (2005). Development of Rich Features for Web-Based Interactive Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Chen, J. X., Carr, D. B., Wechsler, H., & Pan, Z. (2006). Interactive Visualization of Multivariate Statistical Data [<https://doi.org/10.20870/IJVR.2006.5.3.2701>]. *The International Journal of Virtual Reality*, 5(3), 67–73.
- Choi, S. H., Han, K. S., Park, S. J., & Ahn, J. Y. (2014). Visualizing Data with Linked and Comparative Micromap Plots. *International Conference on Artificial Intelligence and Industrial Application* (pp. 94–99). WIT Press, Southampton, U.K.

- Cleveland, W. S. (1981). LOWESS: A Program for Smoothing Scatterplots by Robust Locally Weighted Regression [<https://doi.org/10.2307/2683591>]. *The American Statistician*, 35(1), 54.
- Cleveland, W. S., & McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods [<https://doi.org/10.1080/01621459.1984.10478080>]. *Journal of the American Statistical Association*, 79(387), 531–554.
- Das Gupta, D., & Wong, D. (2021). How “Dependent” Are We? A Spatiotemporal Analysis of the Young and the Older Adult Populations in the US [<https://doi.org/10.1007/s11113-020-09590-y>]. *Population Research and Policy Review*, 40(6), 1221–1252.
- Dray, S., & Jombart, T. (2011). Revisiting Guerry’s Data: Introducing Spatial Constraints in Multivariate Analysis [<https://doi.org/10.1214/10-AOAS356>]. *The Annals of Applied Statistics*, 5(4), 2278–2299.
- Dumelle, M., Kincaid, T. M., Olsen, A. R., & Weber, M. H. (2022). *spsurvey: Spatial Sampling Design and Analysis* [R package version 5.3.0 (<https://CRAN.R-project.org/package=spsurvey>)].
- Ellis, P. (April 30, 2017). Luke-Warm About Micromaps [Web Page, <http://freerangestats.info/blog/2017/04/30/micromaps>].
- Environmental Systems Research Institute, Inc. (1998). ESRI Shapefile Technical Description [White Paper, <https://www.esri.com/Library/Whitepapers/Pdfs/Shapefile.pdf>].
- Environmental Systems Research Institute, Inc. (2016). ArcGIS for Desktop: Shapefile File Extensions [Web Page, <https://desktop.arcgis.com/en/arcmap/10.3/manage-data/shapefiles/shapefile-file-extensions.htm> (accessed August 31, 2022)].
- Everitt, B. S. (2021). *Medical Statistics from A to Z: A Guide for Clinicians and Medical Students (Third Edition)* [<https://doi.org/10.1017/9781108919739>]. Cambridge University Press, Cambridge, U.K.
- Fay, C., Rochette, S., Guyader, V., & Girard, C. (2022). *Engineering Production-Grade Shiny Apps*. CRC Press, Boca Raton, FL.
- Fingerman, S. (2010). Sci-Tech Book News Reviews: Visualizing Data Patterns with Micromaps [<https://jdc.jefferson.edu/scitechnews/vol64/iss4/14>]. *Sci-Tech News*, 64(4), 43–44.
- Firestone, M. J., Wienkes, H., Garfin, J., Wang, X., Vilen, K., Smith, K. E., Holzbauer, S., Plumb, M., Pung, K., Medus, C., Yao, J. D., Binnicker, M. J., Nelson, A. C., Yohe, S., Como-Sabetti, K., Ehresmann, K., Lynfield, R., & Danila, R. (2020). COVID-19 Outbreak Associated with a 10-Day Motorcycle Rally in a Neighboring State - Minnesota, August-September 2020 [<http://doi.org/10.15585/mmwr.mm6947e1>]. *Morbidity and Mortality Weekly Report*, 69(47), 1771–1776.
- Fonseca, J. W., & Wong, D. W. (2000). Changing Patterns of Population Density in the United States [<https://doi.org/10.1111/0033-0124.00242>]. *The Professional Geographer*, 52(3), 504–517.
- Friendly, M. (2007). A.-M. Guerry’s Moral Statistics of France: Challenges for Multivariable Spatial Analysis [<https://doi.org/10.1214/07-STS241>]. *Statistical Science*, 22(3), 368–399.
- Gebreab, S. Y. (2010). *Spatial Epidemiology of Birth Defects in the United States and the State of Utah Using Geographic Information Systems and Spatial Statistics* (Doctoral dissertation) [<https://doi.org/10.26076/c49a-4f7f>]. Utah State University, Department of Watershed Sciences, Logan, Utah.
- Gebreab, S. Y., Davis, S. K., Symanzik, J., Mensah, G. A., Gibbons, G. H., & Diez-Roux, A. V. (2015). Geographic Variations in Cardiovascular Health in the United States:

- Contributions of State- and Individual-Level Factors [<https://doi.org/10.1161/JAHA.114.001673>]. *Journal of the American Heart Association*, 4(6), e001673.
- Gebreab, S. Y., Gillies, R. R., Munger, R. G., & Symanzik, J. (2008). Visualization and Interpretation of Birth Defects Data Using Linked Micromap Plots [<https://doi.org/10.1002/bdra.20419>]. *Birth Defects Research (Part A): Clinical and Molecular Teratology*, 82(2), 110–119.
- Griffith, M. B. (2014). Natural Variation and Current Reference for Specific Conductivity and Major Ions in Wadeable Streams of the Conterminous USA [<https://doi.org/10.1086/674704>]. *Freshwater Science*, 33(1), 1–17.
- Han, K. S., Park, S. J., Mun, G. S., Choi, S. H., Symanzik, J., Gebreab, S., & Ahn, J. Y. (2014). Linked Micromaps for the Visualization of Geographically Referenced Data. *ICIC Express Letters*, 8(2), 443–448.
- Han, K. S., Park, S. J., Symanzik, J., Choi, S. H., & Ahn, J. Y. (2016). Trends in Obesity at the National and Local Level among South Korean Adolescents [<https://doi.org/10.4081/gh.2016.381>]. *Geospatial Health*, 11(381), 130–136.
- Harrower, M. A., & Brewer, C. A. (2003). ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps [<https://doi.org/10.1179/000870403235002042>]. *The Cartographic Journal*, 40(1), 27–37.
- Heiberger, R. M., & Robbins, N. R. (2014). Design of Diverging Stacked Bar Charts for Likert Scales and Other Applications [<https://doi.org/10.18637/jss.v057.i05>]. *Journal of Statistical Software*, 57(5), 1–32.
- Heim, K. (2014). *Visualization and Modeling for Crime Data Indexed by Road Segments* (Doctoral dissertation) [<https://hdl.handle.net/1920/8991>]. George Mason University, Statistical Science, Fairfax, VA.
- Hijmans, R. J. (2022a). *raster: Geographic Data Analysis and Modeling* [R package version 3.6–3 (<http://CRAN.R-project.org/package=raster>)].
- Hijmans, R. J. (2022b). *terra: Spatial Data Analysis* [R package version 1.6–17 (<http://CRAN.R-project.org/package=terra>)].
- Huang, B., Pollock, E., Zhu, L., Athens, J. P., Gangnon, R., Feuer, E. J., & Tucker, T. C. (2018). Ranking Composite Cancer Burden Indices for Geographic Regions: Point and Interval Estimates [<https://doi.org/10.1007/s10552-018-1000-9>]. *Cancer Causes & Control*, 29(2), 279–287.
- Hudson, L., Burus, T., Park, L., Huang, B., Hull, P. C., & Vanderford, N. L. (2024). Cancer Disparities in Appalachian Kentucky [<https://doi.org/10.1111/jrh.12763>]. *The Journal of Rural Health*, 40(1), 87–95.
- Hurst, J., Symanzik, J., & Gunter, L. (2003). Interactive Federal Statistical Data on the Web Using “nViZn” [(CD & <http://interfacesymposia.org/I03/I2003Proceedings/HurstJon/HurstJon.paper.pdf>)]. *Computing Science and Statistics*, 35.
- Iannone, R. (2022). *DiagrammeR: Graph/Network Visualization* [R package version 1.0.9 (<http://CRAN.R-project.org/package=DiagrammeR>)].
- Iannone, R., Allaire, J. J., & Borges, B. (2020). *flexdashboard: R Markdown Format for Flexible Dashboards* [R package version 0.5.2 (<https://CRAN.R-project.org/package=flexdashboard>)].
- Jones, L., & Symanzik, J. (2001). Statistical Visualization of Environmental Data on the Web Using nViZn [(CD & <http://interfacesymposia.org/I01/I2001Proceedings/LJones/LJones.pdf>)]. *Computing Science and Statistics*, 33.
- Kentucky Council of Area Development Districts. (2025). KCADD [Web Page, <https://www.kcadd.org/> (accessed May 27, 2025)].
- Kentucky Department for Public Health, Cabinet for Health and Family Services. (August 31, 2023). State Health Assessment Report, 2023 [Frankfort, Kentucky, <https://www.chfs.ky.gov/agencies/dph/Documents/StateHealthAssessment2023.pdf>].

- Kentucky Legislative Research Commission. (1972). 147A.050 Area Development Districts Created [Web Page, <https://apps.legislature.ky.gov/law/statutes/statute.aspx?id=1652> (accessed September 27, 2022)].
- Kim, M. (2015). Linked Micromap: Exploratory Data Analysis and Geographic Visualization of Spatial Statistics Data [(In Korean), <https://doi.org/10.16879/jkca.2015.15.2.039>]. *Journal of the Korean Cartographic Association*, 15(2), 39–50.
- Kolb, J.-P. (2015). Visualisation of Macroeconomic Indicators in Maps with R [https://eropa.eu/eurostat/cros/system/files/NTTS15_Kolb.pdf]. *Presentation, New Techniques and Technologies for Statistics (NTTS) 2015*. European Commission, Collaboration in Research; Methodology for Official Statistics (CROS).
- Kolvoord, R. A. (2010). Book Review: Visualizing Data Patterns with Micromaps [<http://choicereviews.org/review/10.5860/CHOICE.48-0920>]. *Choice Reviews*, 48(2), 48–0920.
- Kubota, T., Iizuka, M., & Tsubaki, H. (2015). Visualization of Spatial and Paneled Data for Reason-Specified Suicide Data by Prefecture in Japan [<https://2015.isiproceedings.org/Files/CPS415-P1-S.pdf>]. *Proceedings of the 60th World Statistics Congress of the International Statistical Institute, Rio de Janeiro, Brazil, 26-31 July 2015*. International Statistical Institute, The Hague, The Netherlands.
- Lewis, D. R., Pickle, L. W., & Zhu, L. (2017). Recent Spatiotemporal Patterns of US Lung Cancer by Histologic Type [<https://doi.org/10.3389/fpubh.2017.00082>]. *Frontiers in Public Health*, 5, 82.
- Li, C. (2017). *Extracting and Visualizing Data from Mobile and Static Eye Trackers in R and Matlab* (Doctoral dissertation) [<https://doi.org/10.26076/5c8c-d8a5>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Li, C., & Symanzik, J. (2016). The Linked Microposter Plot as a New Means for the Visualization of Eye Tracking Data [(CD)]. *2016 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Li, C., & Symanzik, J. (2017). EyeTrackR: An R Package for Extracting and Visualizing Data from Mobile and Static Eye Trackers [(CD)]. *2018 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Likert, R. (1932). A Technique for the Measurement of Attitudes [https://legacy.voteview.com/pdf/Likert_1932.pdf]. *Archives of Psychology*, 140, 5–55.
- Luo, M., Majumdar, P., & Vasudeva Rao, S. (July 17 – August 7, 2017). The Indian Story [Web Page, https://wiki.smu.edu.sg/1617t3isss608g1/The_Indian_Story].
- Mason, T. J., McKay, F. W., Hoover, R., Blot, W. J., & Fraumeni Jr., J. F. (1975). *Atlas of Cancer Mortality for U.S. Counties: 1950-1969* [DHEW Publication No. (NIH) 75-780, https://archive.org/details/atlasofcancermor00nati_0/mode/2up]. U.S. Department of Health, Education, and Welfare, Public Health Service, National Institutes of Health, Bethesda, MD.
- Massicotte, P., & South, A. (2023). *rnaturrearth: World Map Data from Natural Earth* [R package version 1.0.1 (<https://CRAN.R-project.org/package=rnaturrearth>)].
- Mast, B. D. (2013a). Exploring Housing Cost Data with Conditioned Choropleth Maps [<https://www.huduser.gov/portal/periodicals/cityscape/vol15num3/ch18.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 15(3), 251–255.
- Mast, B. D. (2013b). Visualizing Same-Sex Couple Household Data with Linked Micromaps [<https://www.huduser.gov/portal/periodicals/cityscape/vol15num2/ch23.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 15(2), 267–271.
- Mast, B. D. (2014a). Comparative Micromaps and Changing State Homeownership Rates [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num2/ch12.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(2), 163–167.

- Mast, B. D. (2014b). Mapping White–Black and Temporal Differences in State Homeownership Rates with Two-Way Comparative Micromaps [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num3/ch7.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(3), 149–152.
- Mast, B. D. (2014c). Measuring Spatial Mismatch between Homelessness and Homeless Resources with a Theil Index and Statistical Inference [<https://www.huduser.gov/portal/periodicals/cityscape/vol16num1/ch21.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 16(1), 339–350.
- Mast, B. D. (2015). Measuring Neighborhood Opportunity with AFFH Data [<https://www.huduser.gov/portal/periodicals/cityscape/vol17num3/ch12.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 17(3), 221–230.
- Mast, B. D. (2018). School Performance of Schools Assigned to HUD-Assisted Households [<https://www.huduser.gov/portal/periodicals/cityscape/vol20num3/ch10.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 20(3), 189–222.
- Mast, B. D. (2020). Measuring Homelessness and Resources to Combat Homelessness with PIT and HIC Data [<https://www.huduser.gov/portal/periodicals/cityscape/vol22num1/ch7.pdf>]. *Cityscape: A Journal of Policy Development and Research*, 22(1), 215–225.
- Matthews, S. A. (2013). Book Review: Visualizing Data Patterns with Micromaps [<https://doi.org/10.1007/BF03354893>]. *Spatial Demography*, 1(1), 141–143.
- McManus, M. G., Pond, G. J., Reynolds, L., & Griffith, M. B. (2016). Multivariate Condition Assessment of Watersheds with Linked Micromaps [<https://doi.org/10.1111/1752-1688.12399>]. *Journal of the American Water Resources Association*, 52(2), 494–507.
- Medri, J., Probst, B. D., & Symanzik, J. (2019). Housing Affordability and Immigration: An Exploratory Analysis in New York City [(CD)]. *2019 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Medri Cobos, J. (2021). *Housing Variables and Immigration: An Exploratory and Predictive Data Analysis in New York City* (Master's thesis) [<https://doi.org/10.26076/dd0e-699c>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Monmonier, M. (1993). *Mapping It Out: Expository Cartography for the Humanities and Social Sciences*. University of Chicago Press, Chicago, IL.
- Mons, B. (2018). *Data Stewardship for Open Science: Implementing FAIR Principles*. CRC Press, Boca Raton, FL.
- National Cancer Institute. (2021). Cervical Cancer Prevention (PDQ®)–Patient Version [Web Page, <https://www.cancer.gov/types/cervical/patient/cervical-prevention-pdq> (accessed October 3, 2022)].
- National Cancer Institute. (2025a). Colorectal Cancer Prevention (PDQ®)–Health Professional Version [Web Page, <https://www.cancer.gov/types/colorectal/hp/colorectal-prevention-pdq> (accessed May 27, 2025)].
- National Cancer Institute. (2025b). State Cancer Profiles [Web Page, <https://statecancerprofiles.cancer.gov/> (accessed March 4, 2025, and April 17, 2025)].
- National Cancer Institute (C. Phillips). (2025). As Rates of Some Cancers Increase in Younger People, Researchers Search for Answers [Web Page, <https://www.cancer.gov/news-events/cancer-currents-blog/2025/early-onset-cancer-research-environment-genetics-support> (accessed June 9, 2025)].
- National Cancer Institute (NCI Staff). (2020). Why Is Colorectal Cancer Rising Rapidly among Young Adults? [Web Page, <https://www.cancer.gov/news-events/cancer-currents-blog/2020/colorectal-cancer-rising-younger-adults> (accessed March 28, 2025)].

- National Cancer Institute, Surveillance, Epidemiology, and End Results Program. (2022a). About the SEER Program [Web Page, <https://seer.cancer.gov/about/> (accessed September 8, 2022)].
- National Cancer Institute, Surveillance, Epidemiology, and End Results Program. (2022b). List of SEER Registries [Web Page, <https://seer.cancer.gov/registries/list.html> (accessed April 19, 2022)].
- National Center for Education Statistics. (2011). *The Nation's Report Card: Mathematics 2011 (NCES 2012-458)* [<https://nces.ed.gov/nationsreportcard/pdf/main2011/2012458.pdf>]. Institute of Education Sciences, U.S. Department of Education, Washington, D.C.
- Neuwirth, E. (2022). *RColorBrewer: ColorBrewer Palettes* [R package version 1.1-3 (<https://CRAN.R-project.org/package=RColorBrewer>)].
- Northern Kentucky Area Development District. (2024). About Us [Web Page, <https://www.nkadd.org/about-us/> (accessed January 2, 2024)].
- Nyadanu, S. D., Pereira, G., Nawumbeni, D. N., & Adampah, T. (2019). Geo-Visual Integration of Health Outcomes and Risk Factors Using Excess Risk and Conditioned Choropleth Maps: A Case Study of Malaria Incidence and Sociodemographic Determinants in Ghana [<https://doi.org/10.1186/s12889-019-6816-z>]. *BMC Public Health*, 19, 514.
- Olsen, A. R., Carr, D. B., Courbois, J.-Y. P., & Pierson, S. M. (1996). Presentation of Data in Linked Attribute and Geographic Space. *1996 Abstracts, Joint Statistical Meetings, Chicago, Illinois* (p. 271). American Statistical Association, Alexandria, VA.
- Olsen, A. R., Kincaid, T. M., & Payton, Q. (2012). Spatially Balanced Survey Designs for Natural Resources [<https://doi.org/10.1017/CBO9781139022422.010>]. In R. A. Gitzen, J. J. Millspaugh, A. B. Cooper, & D. S. Licht (Eds.), *Design and Analysis of Long-Term Ecological Monitoring Studies* (pp. 126–150). Cambridge University Press, Cambridge, U.K.
- Open Geospatial Consortium. (2022). Simple Feature Access - Part 1: Common Architecture [Web Page, <https://www.ogc.org/standards/sfa> (accessed August 31, 2022)].
- Park, S. J. (2016). *Visualizing Geospatial Data with Statistical Graphs on Google Maps and Micromaps* (Doctoral dissertation). Chonbuk National University, Jeonju, South Korea.
- Park, S. J., & Ahn, J. Y. (2013). Visualizing Statistical Data Using Linked Micromap Plots [(In Korean)]. *Journal of The Korean Official Statistics*, 18(2), 111–127.
- Park, S. J., & Ahn, J. Y. (2014). Visualization and Interpretation of Cancer Data Using Linked Micromap Plots [<https://doi.org/10.7465/jkdi.2014.25.6.1531>]. *Journal of the Korean Data and Information Science Society*, 25(6), 1531–1538.
- Park, S. J., & Ahn, J. Y. (2015a). Analyzing the Relationship between Standardized Mortality Ratio and Health Indicators Using Linked Micromap Plots. *ICIC Express Letters*, 9(1), 1–7.
- Park, S. J., & Ahn, J. Y. (2015b). Spatial Visualization of Climatic Data Using Micromap Plots in Korea. *ICIC Express Letters, Part B: Applications*, 6(8), 2059–2064.
- Payton, Q. C., Beck, M., Weber, M., McManus, M., Olsen, A. R., & Kincaid, T. (2024). *Vignette: Linked Micromaps* [R package version 1.9.10 (https://cran.r-project.org/web/packages/micromap/vignettes/Introduction_Guide.pdf)].
- Payton, Q. C., McManus, M. G., Weber, M. H., Olsen, A. R., & Kincaid, T. M. (2015). micromap: A Package for Linked Micromaps [<https://doi.org/10.18637/jss.v063.i02>]. *Journal of Statistical Software*, 63(2), 1–16.
- Payton, Q. C., & Olsen, A. R. (2012). *micromap: Linked Micromap Plots* [R package version 1.5 (<https://CRAN.R-project.org/package=micromap>)].

- Payton, Q. C., & Olsen, A. R. (2015). *micromap: Linked Micromap Plots* [R package version 1.9.1 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2018). *micromap: Linked Micromap Plots* [R package version 1.9.3 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2021). *micromap: Linked Micromap Plots* [R package version 1.9.5 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., & Olsen, A. R. (2024). *micromap: Linked Micromap Plots* [R package version 1.9.10 (<https://CRAN.R-project.org/package=micromap>)].
- Payton, Q. C., Weber, M. H., McManus, M. G., Kincaid, T. M., & Olsen, A. R. (2013). Linked Micromaps: Statistical Summaries in a Spatial Context Using the micromap R Package [<https://www.usiale.org/austin2013/presentation-details/4693>]. *Landscape Dynamics Along Environmental Gradients, 2013 Annual Symposium, April 14–18, 2013, Austin, TX*. US Regional Association of the International Association for Landscape Ecology (US–IALE).
- Payton, Q. C., Weber, M. H., McManus, M. G., & Olsen, A. R. (2012). Linked Micromaps: Statistical Summaries in a Spatial Context [https://acwi.gov/monitoring/conference/2012/posters/posters_2012_conference.pdf]. *Water: One Resource — Shared Effort — Common Future, 8th National Monitoring Conference, April 30–May 4, 2012, Portland, Oregon*. National Water Quality Monitoring Council.
- Pearson Jr., J. B., & Carr, D. B. (2016). *micromapST: Linked Micromap Plots for General U. S. and Other Geographic Areas* [R package version 1.1.1 (<http://CRAN.R-project.org/package=micromapST>)].
- Pearson Jr., J. B., & Carr, D. B. (2022). *micromapST: Linked Micromap Plots for U. S. and Other Geographic Areas* [R package version 1.1.3 (<http://CRAN.R-project.org/package=micromapST>)].
- Pearson Jr., J. B., & Carr, D. B. (2024). *micromapST: Linked Micromap Plots for U. S. and Other Geographic Areas* [R package version 3.0.3 (<http://CRAN.R-project.org/package=micromapST>)].
- Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data [<https://doi.org/10.32614/RJ-2018-009>]. *The R Journal*, 10(1), 439–446.
- Pebesma, E. (2022). *sf: Simple Features for R* [R package version 1.0–8 (<http://CRAN.R-project.org/package=sf>)].
- Pebesma, E., & Bivand, R. S. (2022). *sp: Classes and Methods for Spatial Data* [R package version 1.5–0 (<http://CRAN.R-project.org/package=sp>)].
- Pebesma, E., & Bivand, R. S. (2023). *Spatial Data Science: With Applications in R* [<https://doi.org/10.1201/9780429459016> and <https://r-spatial.org/book/>]. Chapman & Hall / CRC, Boca Raton, FL.
- Perrier, V., Meyer, F., & Granjon, D. (2022). *shinyWidgets: Custom Inputs Widgets for Shiny* [R package version 0.6.4 (<https://CRAN.R-project.org/package=shinyWidgets>)].
- Peterson, S. A., Urquhart, N. S., & Welch, E. B. (1999). Sample Representativeness: A Must for Reliable Regional Lake Condition Estimates [<https://doi.org/10.1021/es9807111>]. *Environmental Science & Technology*, 33(10), 1559–1565.
- Pickle, L. W., & Carr, D. B. (2010). Visualizing Health Data with Micromaps [<https://doi.org/10.1016/j.sste.2010.03.007>]. *Spatial and Spatio-Temporal Epidemiology*, 1(2–3), 143–150.
- Pickle, L. W., Mason, T. J., Howard, N., Hoover, R., & Fraumeni Jr., J. F. (1987). *Atlas of U.S. Cancer Mortality Among Whites, 1950–1980* [DHHS Publication No. (NIH) 87-2900, https://www.google.com/books/edition/_/1entwBW4rZoC]. U.S. Department of Health and Human Services, Public Health Service, National Institutes of Health, Washington, D.C.

- Pickle, L. W., Mason, T. J., Howard, N., Hoover, R., & Fraumeni Jr., J. F. (1990). *Atlas of U.S. Cancer Mortality Among Nonwhites, 1950-1980* [DHHS Publication No. (NIH) 90-1582, https://www.google.com/books/edition/Atlas_of_U_S_Cancer_Mortality_Among_Nonw/fGYMvrYfRKEC]. U.S. Department of Health and Human Services, Public Health Service, National Institutes of Health, Washington, D.C.
- Pickle, L. W., Mungiole, M., Jones, G. K., & White, A. A. (1996). *Atlas of United States Mortality* [DHHS Publication No. (PHS) 97-1015, <https://stacks.cdc.gov/view/cd/c/23206>]. U.S. Department of Health and Human Services, Public Health Service, Centers for Disease Control and Prevention, National Center for Health Statistics, Hyattsville, MD.
- Pickle, L. W., Pearson Jr., J. B., & Carr, D. B. (2015). micromapST: Exploring and Communicating Geospatial Patterns in US State Data [<https://doi.org/10.18637/jss.v063.i03>]. *Journal of Statistical Software*, 63(3), 1–25.
- Porta, M., & Last, J. M. (2018). Choropleth Map [<https://www.oxfordreference.com/view/10.1093/acref/9780191844386.001.0001/acref-9780191844386-e-725>]. *A Dictionary of Public Health (Second Edition)*. Oxford University Press.
- Probst, B. D. (2020). ‘LMshapemaker’: Utilizing the ‘Rmapshaper’ R Package to Modify Shapefiles for Use in Linked Micromap Plots (Master’s thesis) [<https://doi.org/10.26076/j9yj-mm66>]. Utah State University, Department of Mathematics and Statistics, Logan, Utah.
- Probst, B. D., & Symanzik, J. (2019). Using ‘rmapshaper’ to Modify Boundary Files for Use in Linked Micromap Plots [[https://2019.isiproceedings.org/Files/10.Contributed-Paper-Session\(CPS\)-Volume-4.pdf](https://2019.isiproceedings.org/Files/10.Contributed-Paper-Session(CPS)-Volume-4.pdf)]. *Proceeding, Contributed Paper Session Volume 4, 62nd ISI World Statistics Congress 2019, 18–23 August 2019, Kuala Lumpur, Malaysia* (pp. 281–289). Department of Statistics Malaysia, Putrajaya, Malaysia.
- R Core Team. (2017). *R: A Language and Environment for Statistical Computing* [<http://www.R-project.org/>]. R Foundation for Statistical Computing. Vienna, Austria.
- R Core Team. (2022). *R: A Language and Environment for Statistical Computing* [<http://www.R-project.org/>]. R Foundation for Statistical Computing. Vienna, Austria.
- Sahar, L., Foster, S. L., Sherman, R. L., Henry, K. A., Goldberg, D. W., Stinchcomb, D. G., & Bauer, J. E. (2019). GIScience and Cancer: State of the Art and Trends for Cancer Surveillance and Epidemiology [<https://doi.org/10.1002/cncr.32052>]. *Cancer*, 125(15), 2544–2560.
- Saia, S. M., Nelson, N. G., Young, S. N., Parham, S., & Vandegrift, M. (2022). Ten Simple Rules for Researchers Who Want to Develop Web Apps [<https://doi.org/10.1371/journal.pcbi.1009663>]. *PLoS Computational Biology*, 18(1), e1009663.
- Salomon, J. A., Reinhart, A., Bilinski, A., Chua, E. J., La Motte-Kerr, W., Rönn, M. M., Reitsma, M. B., Morris, K. A., LaRocca, S., Farag, T. H., Kreuter, F., Rosenfeld, R., & Tibshirani, R. J. (2021). The US COVID-19 Trends and Impact Survey: Continuous Real-Time Measurement of COVID-19 Symptoms, Risks, Protective Behaviors, Testing, and Vaccination [<https://doi.org/10.1073/pnas.2111454118>]. *Proceedings of the National Academy of Sciences*, 118(51), e2111454118.
- Senkayi, S. N., & Sattler, M. L. (2013). Varying Spatial Levels in GIS Analysis Environmental Epidemiological Data in Texas [https://www.cc.gatech.edu/gvu/ii/PublicHealthVis/Papers/VaryingSpatialLevelsGIS-EnvirnomentalEPI_F.pdf]. *Public Health’s Wicked Problems: Can InfoVis Save Lives?, Workshop, October 13, 2013, Atlanta, GA, USA*.
- Siegel, R. L., Fedewa, S. A., Anderson, W. F., Miller, K. D., Ma, J., Rosenberg, P. S., & Jemal, A. (2017). Colorectal Cancer Incidence Patterns in the United States, 1974–

- 2013 [<https://doi.org/10.1093/jnci/djw322>]. *JNCI: Journal of the National Cancer Institute*, 109(8), djw322.
- Sievert, C. (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. CRC Press, Boca Raton, FL.
- Silvanima, J., Woeber, A., Sunderman-Barnes, S., Copeland, R., Sedlacek, C., & Seal, T. (2018). A Synoptic Survey of Select Wastewater-Tracer Compounds and the Pesticide Imidacloprid in Florida's Ambient Freshwaters [<https://doi.org/10.1007/s10661-018-6782-4>]. *Environmental Monitoring and Assessment*, 190(7), 435.
- Sips, M., Schneidewind, J., & Keim, D. A. (2007). Highlighting Space–Time Patterns: Effective Visual Encodings for Interactive Decision–Making [<https://doi.org/10.1080/13658810701362147>]. *International Journal of Geographical Information Science*, 21(8), 879–893.
- Slocum, T. A., McMaster, R. B., Kessler, F. C., & Howard, H. H. (2023). Data Exploration [<https://doi.org/10.1201/9781003150527-28>]. In T. A. Slocum, R. B. McMaster, F. C. Kessler, & H. H. Howard (Eds.), *Thematic Cartography and Geovisualization (Fourth Edition)* (Online). CRC Press, Boca Raton, FL.
- Stabler, B. (2022). *shapefiles: Read and Write ESRI Shapefiles* [R package version 0.7.2 (<http://CRAN.R-project.org/package=shapefiles>)].
- Stevens Jr., D. L., & Olsen, A. R. (2003). Variance Estimation for Spatially Balanced Samples of Environmental Resources [<https://doi.org/10.1002/env.606>]. *Environmetrics*, 14(6), 593–610.
- Stevens Jr., D. L., & Olsen, A. R. (2004). Spatially Balanced Sampling of Natural Resources [<https://doi.org/10.1198/016214504000000250>]. *Journal of the American Statistical Association*, 99(465), 262–278.
- Symanzik, J. (2004). Interactive and Dynamic Graphics. In J. E. Gentle, W. Härdle, & Y. Mori (Eds.), *Handbook of Computational Statistics — Concepts and Methods* (pp. 293–336). Springer, Berlin, Heidelberg.
- Symanzik, J. (2012). Interactive and Dynamic Graphics [https://doi.org/10.1007/978-3-642-21551-3_12]. In J. E. Gentle, W. K. Härdle, & Y. Mori (Eds.), *Handbook of Computational Statistics, Volume 1 — Concepts and Methods (Second Edition)* (pp. 335–373). Springer, Berlin, Heidelberg.
- Symanzik, J. (2014). Exploratory Spatial Data Analysis [https://doi.org/10.1007/978-3-642-23430-9_76]. In M. M. Fischer & P. Nijkamp (Eds.), *Handbook of Regional Science* (pp. 1295–1310). Springer, Berlin, Heidelberg.
- Symanzik, J. (2021). Exploratory Spatial Data Analysis [https://doi.org/10.1007/978-3-662-60723-7_76]. In M. M. Fischer & P. Nijkamp (Eds.), *Handbook of Regional Science (Second and Extended Edition)* (pp. 1845–1861). Springer, Berlin, Heidelberg.
- Symanzik, J., Axelrad, D. A., Carr, D. B., Wang, J., Wong, D., & Woodruff, T. J. (1999). HAPs, Micromaps and GPL — Visualization of Geographically Referenced Statistical Summaries on the World Wide Web. *Annual Proceedings (ACSM-WFPS-PLSO-LSAW 1999 Conference CD)*. American Congress on Surveying & Mapping.
- Symanzik, J., Bao, S., Dai, X., Shui, M., & She, B. (2016). Recent Advancements in Geovisualization, with a Case Study on Chinese Religions [https://doi.org/10.1007/978-3-319-42571-9_8]. In Z. Jin, M. Liu, & X. Luo (Eds.), *New Developments in Statistical Modeling, Inference and Application: Selected Papers from the 2014 ICSA/KISS Joint Applied Statistics Symposium in Portland, OR* (pp. 151–166). Springer, Cham, Switzerland.
- Symanzik, J., & Carr, D. B. (2008). Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data. In C. Chen, W. Härdle, & A. Unwin (Eds.), *Handbook of Data Visualization* (267–294 & 2 Color Plates). Springer, Berlin, Heidelberg.

- Symanzik, J., & Carr, D. B. (2013). Linked Micromap Plots in R. In S.-H. Cho (Ed.), *Proceedings of IASC–Satellite Conference for the 59th ISI WSC & The 8th Conference of IASC–ARS* (pp. 213–218). Asian Regional Section of the IASC.
- Symanzik, J., Carr, D. B., Axelrad, D. A., Wang, J., Wong, D., & Woodruff, T. J. (1999). Interactive Tables and Maps — A Glance at EPA's Cumulative Exposure Project Web Page. *1999 Proceedings of the Section on Statistical Graphics* (pp. 94–99). American Statistical Association, Alexandria, VA.
- Symanzik, J., Carr, D. B., McManus, M. G., & Weber, M. H. (2017). Micromaps [<https://doi.org/10.1002/9781118445112.stat07938>]. *Wiley StatsRef: Statistics Reference Online*. Wiley Online Library.
- Symanzik, J., Dai, X., Weber, M. H., Payton, Q., & McManus, M. G. (2014). Linked Micromap Plots for South America — General Design Considerations and Specific Adjustments [<https://doi.org/10.15446/rce.v37n2spe.47949>]. *Revista Colombiana de Estadística*, 37(2), 451–469.
- Symanzik, J., Gebreab, S., Gillies, R., & Wilson, J. (2003). Visualizing the Spread of West Nile Virus [(CD)]. *2003 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Hurst, J., & Gunter, L. (2002). Recent Developments for Interactive Statistical Graphics on the Web Using “nViZn” [(CD)]. *2002 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., & Jones, L. (2001). “nViZn” Federal Statistical Data on the Web [(CD)]. *2001 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Li, C., Zhang, B., Studenka, B., & McKinnney, E. (2017). Eye–Tracking in Practice: A First Analysis of a Study on Human Postures [(CD)]. *2017 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Symanzik, J., Wong, D., Wang, J., Carr, D. B., Woodruff, T. J., & Axelrad, D. A. (2000). Web-based Access and Visualization of Hazardous Air Pollutants [<https://webharvest.gov/peth04/20041025104701/http://www.atsdr.cdc.gov/gis/conference98/proceedings/pdf/symanzik.pdf>]. *Geographic Information Systems in Public Health: Proceedings of the Third National Conference, August 18–20, 1998, San Diego, California* (pp. 235–248). Agency for Toxic Substances & Disease Registry.
- Talbot, J. (2020). *labeling: Axis Labeling* [R package version 0.4.2 (<http://CRAN.R-project.org/package=labeling>)].
- Tatalovich, Z., & Stinchcomb, D. G. (2019). Creating Maps and Mapping Systems for Cancer Control and Prevention [https://doi.org/10.1007/978-3-030-18408-7_3]. In D. Berrigan & N. A. Berger (Eds.), *Geospatial Approaches to Energy Balance and Breast Cancer* (pp. 59–79). Springer Nature, Cham, Switzerland.
- Teucher, A., & Russell, K. (2022). *rmapshaper: Client for ‘mapshaper’ for ‘Geospatial’ Operations* [R package version 0.4.6 (<http://CRAN.R-project.org/package=rmapshaper>)].
- Thapliyal, A. (2009). Enhancement of Web-based Interactive Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- The Nation’s Report Card. (2022). Data Tools: NAEP Data Explorer [Web Page, <https://www.nationsreportcard.gov/nedcore/xplore/NDE> (accessed May 4, 2022)].
- Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press.
- Tufte, E. R. (1990). *Envisioning Information*. Graphics Press.
- Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press.
- Ugarte, M. D. (2012). Book Review: Visualizing Data Patterns with Micromaps [https://doi.org/10.1111/j.1467-985X.2012.01069_3.x]. *Journal of the Royal Statistical Society, Series A: Statistics in Society*, 175(4), 1072–1073.

- United States Census Bureau. (2013). co34_d00_shp.zip [Shapefiles for New Jersey] [Web Page, <https://www2.census.gov/geo/tiger/PREVGENZ/co/co00shp/>] (accessed October 17, 2022)].
- United States Census Bureau. (2021). Cartographic Boundary Files - Shapefile [Web Page, <https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.2000.html>] (accessed September 8, 2022)].
- United States Census Bureau. (2022a). American Community Survey, S1501, Educational Attainment [Web Page, <https://data.census.gov/cedsci/table?text=s1501&t=Education&g=0400000US24,24%240500000>] (accessed April 15, 2022)].
- United States Census Bureau. (2022b). American Community Survey, S1701, Poverty Status in the Past 12 Months [Web Page, <https://data.census.gov/cedsci/table?text=s1701&g=0400000US24,24%240500000>] (accessed April 15, 2022)].
- United States Census Bureau. (2022c). American National Standards Institute (ANSI) and Federal Information Processing Series (FIPS) Codes [Web Page, <https://www.census.gov/library/reference/code-lists/ansi.html>] (accessed September 8, 2022)].
- United States Census Bureau. (2025). American Community Survey (ACS) [Web Page, <https://www.census.gov/programs-surveys/acs.html>] (accessed May 27, 2025)].
- University of Kentucky, Markey Cancer Center, Kentucky Cancer Consortium, Kentucky Cancer Program. (2022). Kentucky Cancer Registry: The Population-Based Central Cancer Registry for the Commonwealth of Kentucky [Web Page, <https://www.kcr.uky.edu/>] (accessed September 27, 2022)].
- Unwin, A. (2011). Short Book Review: Visualizing Data Patterns with Micromaps [https://doi.org/10.1111/j.1751-5823.2011.00134_14.x]. *International Statistical Review*, 79(1), 127–128.
- USA Facts. (2022). US COVID-19 Cases and Deaths by State [Web Page, <https://usafacts.org/visualizations/coronavirus-covid-19-spread-map/>] (accessed April 19, 2022)].
- Utlaut, T., & Morgan, G. (2010). On Enhancing JMP's Visual Analytics Using JMP Scripting Language (JSL) [<https://community.jmp.com/t5/Discovery-Summit-2010/On-Enhancing-JMP-s-Visual-Analytics-Using-JMP-Scripting-Language/ta-p/40562>]. *JMP User Community Discovery Summit 2010*. JMP Statistical Discovery LLC.
- Vaidyanathan, R., Xie, Y., Allaire, J. J., Cheng, J., Sievert, C., & Russell, K. (2021). *html-widgets: HTML Widgets for R* [R package version 1.5.4 (<https://CRAN.R-project.org/package=htmlwidgets>)].
- Virginia Department of Environmental Quality. (2020). Chapter 4.4 Freshwater Probabilistic Monitoring Results [<https://www.deq.virginia.gov/home/showpublisheddocument/2221/637436316328230000>]. *2020 305(b)/303(d) Water Quality Assessment Integrated Report, EPA Approved, December 9, 2020* (pp. 112–152). Virginia DEQ, Richmond, VA.
- Voge, N. D. (2012). Ignoring the Spatial Context in Intro Statistics Classes — And Some Simple Graphical Remedies [MS Report, Utah State University, Department of Mathematics and Statistics, Logan, UT, <https://doi.org/10.26076/663e-4a7>].
- Voge, N. D., & Symanzik, J. (2011). Ignoring the Spatial Context in Intro Stats Classes — And Some Simple Graphical Remedies [(CD)]. *2011 JSM Proceedings*. American Statistical Association, Alexandria, VA.
- Walker, K. (2024). *tigris: Load Census TIGER/Line Shapefiles* [R package version 2.1, (<https://CRAN.R-project.org/package=tigris>)].
- Wang, X., Chen, J. X., Carr, D. B., Bell, B. S., & Pickle, L. W. (2002). Geographic Statistics Visualization: Web-based Linked Micromap Plots [<https://doi.org/10.1109/5992.998645>]. *Computing in Science & Engineering*, 4(3), 90–94.

- Wartenberg, D. (2009). Some Considerations for the Communication of Results of Air Pollution Health Effects Tracking [<https://doi.org/10.1007/s11869-009-0046-y>]. *Air Quality, Atmosphere & Health*, 2(4), 207–221.
- Wickham, H. (2019). *Advanced R (Second Edition)* [<https://adv-r.hadley.nz/>]. CRC Press, Boca Raton, FL.
- Wickham, H. (2021). *Mastering Shiny: Build Interactive Apps, Reports, and Dashboards Powered by R* [<https://mastering-shiny.org/>]. O'Reilly Media, Inc., Sebastopol, CA.
- Wickham, H., Cook, D., Hofmann, H., & Buja, A. (2010). Graphical Inference for Infovis [<https://doi.org/10.1109/TVCG.2010.161>]. *IEEE Transactions on Visualization and Computer Graphics*, 16(6), 973–979.
- Wilkinson, L., Rope, D. J., Carr, D. B., & Rubin, M. A. (2000). The Language of Graphics [<https://doi.org/10.1080/10618600.2000.10474897>]. *Journal of Computational and Graphical Statistics*, 9(3), 530–543.
- Wilkinson, L., Rope, D. J., Rubin, M. A., & Norton, A. (2001). nViZn: An Algebra-Based Visualization System [<https://www.semanticscholar.org/paper/nViZn-%3A-An-Algebra-Based-Visualization-System-Wilkinson/59c5ed82acbfe5fe043be0bd003385e279e822f>]. *1st International Symposium on Smart Graphics, March 21-23, 2001, Hawthorne, NY, USA*.
- Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016). The FAIR Guiding Principles for Scientific Data Management and Stewardship [<https://doi.org/10.1038/sdata.2016.18>]. *Scientific Data*, 3, 160018.
- Xie, Y. (2015). *Dynamic Documents with R and knitr (second edition)* [<https://yihui.org/knit/>]. CRC Press, Boca Raton, FL.
- Xie, Y. (2022a). *bookdown: Authoring Books and Technical Documents with R Markdown* [R package version 0.29 (<http://CRAN.R-project.org/package=bookdown>)].
- Xie, Y. (2022b). *knitr: A General-Purpose Package for Dynamic Report Generation in R* [R package version 1.40 (<http://CRAN.R-project.org/package=knitr>)].
- Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R Markdown: The Definitive Guide* [<https://bookdown.org/yihui/rmarkdown/>]. CRC Press, Boca Raton, FL.
- Xie, Y., Dervieux, C., & Riederer, E. (2021). *R Markdown Cookbook* [<https://bookdown.org/yihui/rmarkdown-cookbook>]. CRC Press, Boca Raton, FL.
- Yarra, P. K. (2010). Refactoring Micromaps [MS Report, Utah State University, Department of Computer Science, Logan, UT].
- Yu, B. (2021). Real-World Evidence from Population-Based Cancer Registry Data [<https://doi.org/10.1201/9780429398674-3>]. In H. Yang & B. Yu (Eds.), *Real-World Evidence in Drug Development and Evaluation* (pp. 47–70). Chapman & Hall / CRC, Boca Raton, FL.
- Zhang, C. (2012). *Interfaces and Visual Analytics for Visualizing Spatio-Temporal Data with Micromaps* (Doctoral dissertation) [<https://hdl.handle.net/1920/7888>]. George Mason University, Computational Sciences and Informatics, Fairfax, VA.
- Zhang, Y., & Carr, D. B. (2014). Providing Overviews of China Datasets Using Linked Micromaps [<https://doi.org/10.1109/COM.Geo.2014.20>]. *2014 Fifth International Conference on Computing for Geospatial Research and Application, Washington, DC, USA* (p. 123). IEEE.
- Zhu, L., Pickle, L. W., & Pearson Jr., J. B. (2016). Confidence Intervals for Rate Ratios between Geographic Units [<https://doi.org/10.1186/s12942-016-0073-5>]. *International Journal of Health Geographics*, 15, 44.

Index

- Africa country boundary file, 104
Aggregating boundaries, 119
Alaska, 106, 108
AOI, *see* Area of interest
Area of interest, 303–306, 308, 312, 313, 317, 320, 321, 328, 330, 332
Arrow plots, 185
Arrows, 192
- Bar chart, 128
Bar chart with confidence bounds, 128
Bar charts, 185
Bars, centered, 194
Bars, normalized, 192
Bars, segmented, 192
Boxplot, 3, 51, 56, 60, 61, 65–67, 128, 189, 191, 192, 320, 321, 330, 331
Boxplots, 185
BuildBorderGroup, 105
Bureau of Labor Statistics, 69
- CCmap, *see* Conditioned choropleth map
Centers for Disease Control and Prevention, 346
China boundary file, 104
Choropleth map, 2
Color blindness, 13, 87, 127, 243, 299
Color scheme
 Divergent, 42, 43
 Rainbow colors, 42
 Sequential, 42
 Spectral, 42
Colorectal cancer
 Early onset, 345, 365
 Risk factors, 345
 Time trends, 345, 347
Colors
 Colorblind safe, 42
 Greyscale publication, 42
 micromapST options, 77, 78
 Print friendly, 42
Comparative micromaps, 1, 11
- Conditioned choropleth map, 13, 87, 127, 243, 299
Conditioned choropleth map., 243
Conditioned micromaps, 1, 10
Confidence intervals, 185
Cumulative map shading, 77
Cumulative shading, 196
- Data aggregation, 347
Data communication, 355, 365
Data suppression, 346, 362
Datasets
 AllData, 349
 AOIName, 305
 cars, 12, 13
 edPov, 14, 32, 34, 35, 37, 38, 40, 41, 43, 46, 49, 87, 127, 243, 299
 FL_pesticide, 337
 iris, 12, 13
 KYADDAllData, 362
 KYADDMort, 362
 KYADDRisk, 362
 MathProficiency8thGr2011, 194
 MDPovEd, 82, 83
 MDPovEducACSData20162020, 198
 mort_ratesUS, 346
 MortData, 349
 OR_lakes, 342
 poster_dataset_all, 305
 RiskData, 349
 st_covid_case_rates, 71, 72, 74, 78, 80, 187
 state.x77, 234, 236, 238
 stats, 335
 USstates, 14, 32, 33, 87, 127, 234, 238, 243, 299
 VA_map_table, 335
 WV_Watershed, 50, 51, 53, 55, 128
- District of Columbia, 108
Dot plots, 185
Dotplot, 3, 6–8, 25, 35, 42, 56, 67, 128, 320, 328, 329

- Dotplot with confidence bounds, 51, 56, 60, 61, 65, 67, 128
- Environmental Protection Agency, 69, 70
- ESRI, 112
- ESRI, Inc., 105
- Exploratory spatial data analysis, 69, 345, 365
- Geospatial clusters, 70, 77, 345
- Glyphs, 185, 199
- GPL, *see* Graphics Production Library
- Graphics Production Library, 9
- Hawaii, 106, 108
- Hydrologic unit code, 59
- JAVA, 70
- Kansas county boundary file, 104
- Kentucky
- Area Development Districts, 359
 - Cancer data, 346
- Kentucky Area Development Districts, 119, 122
- Kentucky cancer data, 116, 122
- Korea, Seoul city district boundary file, 104
- Likert scale, 192
- Linked micromap plot, 1–4, 6, 8–10, 13, 14, 29–43, 45, 46, 48–51, 53–61, 63, 65–71, 74, 82, 85, 87, 88, 103, 127, 128, 132, 185, 201, 243, 299, 303–306, 331, 332
- Linked micromap plot for point locations, 201
- Linked micromap plots, 317, 345, 349
- Linked microposter plot, 303–310, 312, 317, 330, 331
- Linked scanpath microposter plot, 332
- LMplot, *see* Linked micromap plot
- longitude/latitude coordinates, 112
- LOWESS smoother, 198
- Maryland county boundary file, 104
- Micromap visualizations, 1–3
- micromapST
- Arrow plot, 362
 - Dot plot, 352, 355, 362
 - LOWESS smoother, 355, 359
 - Scatter plot, 359
 - Time series plot, 350
- Multipolygon areas, 112
- National Cancer Institute, 69, 70, 82
- State Cancer Profiles, 346
- National Center for Health Statistics, 69
- New York county boundary file, 104
- nViZn, 9
- Panel, 1, 3, 7, 35, 303, 306
- Perceptual group, 3, 4, 10, 13, 77, 83, 87, 103, 113, 127, 243, 299
- polygon, 103
- projection of coordinates, 112
- projection, Albers equal area, 112
- Prostate cancer, 348
- Puerto Rico, 106
- Quantile-quantile plot, 13, 87, 127, 243, 299
- R Markdown, 67
- R Packages
- bookdown, xvii, 67
 - datasets, 234, 236, 238
 - DiagrammeR, 31
 - EyeTrackR, 303, 332
 - flexdashboard, 234
 - ggmap, 13
 - ggplot2, 13
 - htmlwidgets, 233
 - imager, 308
 - knitr, xvii
 - labeling, 45
 - micromap, xvii, 13, 14, 29–33, 45, 50, 51, 67, 68, 87, 127, 128, 132, 201, 202, 224, 227, 229, 234, 238, 240, 243, 299, 312, 332, 333
 - micromapExtra, 1, 50, 51, 305, 310
 - micromapST, xvii, 13, 30, 67, 69–72, 74, 78, 80, 82, 85, 87, 103, 127, 185, 229, 240, 243, 299, 332, 345, 359
 - raster, 50, 51
 - RColorBrewer, 32, 42, 317
 - rmarkdown, 234
 - rnaturalearth, 206
 - rsconnect, 238
 - sf, 32, 50, 51, 202, 204, 343
 - shapefiles, 50
 - shiny, 18, 227, 229, 236, 238, 240
 - shinydashboard, 234
 - shinythemes, 236
 - shinyWidgets, 233
 - sp, 32, 112

- spsurvey, 334, 341, 342
- terra, 50
- tigris, 211
- R packages
 - rmapshaper, 112
 - spdep, 112
- Reference line, 77, 185, 191
- Rhode Island, 109
- Row plot, 69
- S-Plus, 8, 29
- S-plus, 70
- Scaled response data, 192
- Scatter plots, 185, 198
- Scatterplot, 3, 8
- Scatterplot matrix, 8
- Segmented bars, 185
- Shapefile, 30, 31, 111, 303–306, 308, 331
- shapefile, 103, 105
- Shapefiles, 30, 50, 51, 67, 68
- Simple features, 31, 50, 51, 53, 67
- SpatialPolygonsDataFrame, 30–33, 50, 51, 54
- State Cancer Profiles, 70
- Surveillance, Epidemiology and End Results (SEER) boundary file, 104
- Time Series, 3
- Time series plot, 187, 189
- Time series plots, 185
- UK, Ireland boundary file, 104
- Utah county boundary file, 104
- visibility map, 104