MRP Package Vignette

Michael Malecki

2010-10-20

1 Running Example

The running example is the same combined poll data and model in Kastellec's MRP Primer, dealing with support for same-sex marriage in 2004–2005. A simple model using only a few group-level intercepts, is run and subject to `R CMD check`; in the "not run" section of the examples, code for the complete model with census poststratification, individual- and state-level predictors, and categorical interactions is shown. This vignette walks through both of those examples.

1.1 Data

The data for the example are the combined survey results in the "marriage.data" data.frame, and state-level predictors in "Statelevel" data.frame. Both of these are loaded by the command `data(samesexmarriage)`. The package includes several other pieces of data that are useful to US practitioners of MRP:

mrp.census Census data with main data columns 'weighted2000', 'weighted2004', and 'weighted2008'. See documentation for `mrp.census` for details on the demographic features in the file.

mrp.regions A data.frame with state two-letter abbreviations and five census region codes, with DC as its own region.

spmap.states A projected map object with state names, FIPS codes, and two-letter state abbreviations.

```
R> data(samesexmarriage)
R> data(mrp.census)
R> data(mrp.regions)
R> data(spmap.states)
```

2 Fitting a basic model

2.1 Preparing the data

Almost all data will involve recoding and careful checking of categories. In particular, we rely heavily on R's "factor" data type. Factors have associated "levels" (names for categories) and may be ordered. For MRP, categorical variables need to be factors and the levels need to match between the survey data and the poststratification (census) data.

```
R> marriage.data <- within(marriage.data, {
       state <- factor(state,exclude=NA)
       poll <- factor(poll,exclude=NA)
       age <- factor(age.cat,exclude=NA,
                     labels=c("18-29","30-44","45-64","65+"))
       edu <- factor(edu.cat,exclude=NA,labels=c("< High School",
                                         "High School",
                                         "Some College",
                                         "Graduated College"))
       ## Code interaction here, first fixing levels
       female <- factor(female,levels=c(0,1),
                        labels=c("Male","Female"))
       race.wbh <- factor(race.wbh)
       levels(race.wbh) <- c("White","Black","Hispanic")
       f.race <- interaction(female,race.wbh)
     })
R>   ## Remove empty "" state and drop it from levels.
R>   marriage.data <- subset(marriage.data,!is.na(state) & state!="" )
R>   marriage.data$state <- factor(marriage.data$state)
```

In this case, the poll data we have uses four instead of five categories for education. Below, we combine the top two levels of census education into a survey-matching single "Graduated College" level. We also drop any states from the census that are not in our survey dataset.

```
R>   mrp.census <- na.omit(mrp.census[mrp.census$state %in% marriage.data$state,])
R>   mrp.census <- within(mrp.census,{
       age <- factor(age,exclude=NA,labels=c("18-29","30-44","45-64","65+"))
       education[education=="postgraduate"] <- "college graduate"
       edu <- factor(education,exclude=NA,labels=c("< High School",
                                          "High School",
                                          "Some College",
                                          "Graduated College"))
       state <- factor(state,exclude=NA)
```

```
      race[race==”Other”] <- NA
      race <- factor(race,exclude=NA)
      f.race <- interaction(sex,race)
    })
R>  mrp.census <- na.omit(mrp.census)
```

2.2 Calling `mrp()` for the first time

In the most basic setup MRP will fit an intercept to each of the groups provided in the formula. In the simple example, these categories are **state**, **age**, and **education**; all of these exist in the census data as well. The outcome variable `yes.of.all` is already coded as 0/1; within `mrp()` it is transformed into the two-column form used for estimation.

```
R> mrp.simple <- mrp(yes.of.all ~ state+age+edu,
                     poll=marriage.data,
                     population=mrp.census,
                     use=”weighted2004”)
```

The `mrp` function then executes the following steps:

1. Stratify the poll data according to the formula. This creates an N-way array of data where the dimensions are given by the categories of the stratification variables. On this array, it computes the $\bar{Y}$, and the effective N taking into account any survey weights provided, and the “design effect” in that cell of the array.

2. Collapse the N-way survey array into a rectangular matrix, using $\bar{Y}$, effective N, and the design effects to form for each combination of strata (‘cell’ of the N-way data) a sum of yes and no responses.

3. Perform any transformations on the prepared data; the ‘add’ argument is discussed in detail below.

4. Stratify the population data, creating an N-way array of matching dimension to that of the survey.

5. Estimate a multilevel model. By default this is a call to `glmer` with `family= quasibinomial( link=”logit”)`. By default, as in the call above, the formula for this model fits just an intercept for each stratum in the specification.

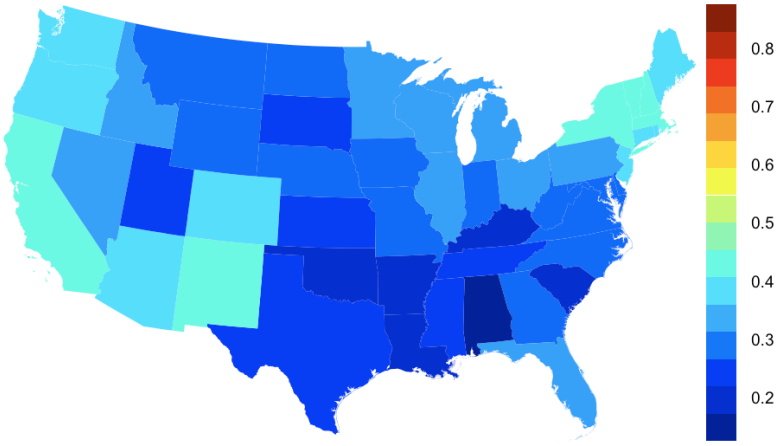
Poststratification is straightforward: multiply the fitted values from the multilevel model by the population frequencies, and collapse across any remaining dimensions. Below, we collapse across states to show the poststratified estimates of support for same-sex marriage by education and age using the formula interface of the ‘poststratify’ method.

Table 1. Support for same-sex marriage (in percent) by level of education and age cohort. Simple model poststratified results.

	18-29	30-44	45-64	65+
< High School	37	27	22	14
High School	40	30	24	16
Some College	48	37	31	21
Graduated College	57	46	39	28

Figure 1. Support for same-sex marraige from simple MRP model.

```
R> print(spplot(mrp.simple, state~state))
```




```
R> xtable(100*poststratify(mrp.simple, ~ edu+age), digits=0)
```

2.3 Basic Map Plotting

The package makes it easy to plot results onto maps.[1] For a wide variety of conditioning plots the *MRP* `spplot` method preserves the same formula interface and to build complex maps. The method for `mrp` objects extends the `spplot` method, which uses Lattice as the "high level" graphics language. In future versions of *MRP* we hope to make *ggplot2* maps with poststratified data similarly easy.

A map of results poststratified by state with no further conditioning (collapsed across the other strata) using the included map object and `spplot` defaults is fantastically easy to produce.

[1]Maps very quickly blow up file sizes, and bitmap renditions may be acceptable for many documents, at least for drafts. For this reason we have submitted a patch to R to add a 'png' option for Sweave.

3 The Full Model

In this section we demonstrate the use of the high-level arguments to `mrp()` for fitting more complex models with more data including intercepts strata not present in the population, additional group-level predictors, and transformations in the data.

The full model includes intercepts for poll (the data combine several polls), for region (states are grouped into regions), state-level predictors such as religious attendance and democratic vote share in previous elections, and interactions between the cross-classifying strata. The interface to supply or create this data and modify the regression formula are intuitive to use.

3.1 Modifying Population Strata

We would like to separate the estimation data by poll, but obviously the population data are the same across polls. To remove a stratification dimension, subtract it from the initial formula by adding this argument:

```
R> population.formula= . ~ . - poll
```

3.2 Joining Predictors and Transforming Data

the `add` argument is a powerful way to add predictors by left-joining ("merging") other data.frames onto the prepared flattened cross-classified data. In this case we have state-level predictors – a few that are used in the analysis and several others; and a set that are used only in the display – but they are simple to join on matching keys in same-named columns. The `Statelevel` data.frame has a column 'state' with factor levels matching those in the prepared data.

Two common transformations are used in the data for this example: making a continuous group-level predictor out of the categorical one by rescaling it, and making an additional categorical variable by the interaction of other ones that are already included as cross-classifying strata.

Both types of additional data are provided in a `list`:

```
R> add=list(
     Statelevel,
     Regions,
     expression(z.age <- rescale(age)),
     expression(age.edu <- interaction(age,edu)))
```

3.3 Specifying the Multilevel Regression Formula

Finally, we need to adjust the multilevel regression formula. By default, *MRP* will build a simple model for only intercepts by each stratum indicated in the main formula. We want to include all of those, but also the state-level predictors and possibly varying slopes in some of them as well. Again we use the dot to indicate what has already been included.

```
R> mr.formula= .~.+ (1|region) + (1|age.edu) + z.age + p.relig.full + p.kerry.full
```

3.4 Re-fitting with existing data

The same updating of the formula is used in the event that you want to re-run or extend an existing analysis and modify its formula using the `mr()` method.

3.5 Full Model `mrp()` Call

```
R> mrp.statelevel <- mrp(yes.of.all~
                         state+f.race+age+edu+poll,
                         poll=marriage.data,
                         population=mrp.census,use="weighted2008",
                         population.formula=.~.-poll,
                         add=list(Statelevel,
                           mrp.regions,
                           expression(age.edu <- interaction(age,edu)),
                           expression(z.age <- rescale(age))),
                         mr.formula=.~.+(1|region)+ (1|age.edu)+
                          z.age+p.relig.full+p.kerry.full
                         )
```

4 Advanced Plotting with `spplot`

The `spplot,mrp-method` uses all parts of the fitted `mrp` object to produce beautiful plots with meaningful labels with the familiar fomula interface, intelligent subsetting, easy customization, and the addition of other information to the plot by selectively modifying the stroke on a list of features.

The `spplot` method takes an `mrp` object and a stratification specification, and merges it onto a map from the SpatialPolygonsDataFrame class, provided by the *sp* package. Using *maptools*, users can read in their own ESRI shapefiles. Shapefiles are somewhat of a standard for GIS applications. As mentioned in the data section above, we have included a map read into R from a publicly available shapefile. According to

Wickham, *sp* maps can be used with *ggplot2* via `fortify` methods; we plan to make poststratified results more accessible to *ggplot2* methods in the future.

4.1 Conditioning Panels

The `formula` argument to our `spplot` method takes a similar form to the overall `mrp` formula specifying the different cross-classifying strata desired on the plot. The left-hand side indicates the variable name in the data that corresponds to a variable name in the map. In the example, the data has two-letter US state abbreviations in a variable called 'state' and the included map has the same data in the variable 'STATE' (the function default). Right-hand-side conditioning variables in the formula form the rows and columns of a faceted "trellis" or "small multiples" display.

The 'exclude' argument suppresses plotting of certain geographic units by removing them from the map data; the same effect can be achieved with 'subset', except 'subset' applies to all variables not just the geographic one. In this way we could focus on just a subset of respondents, for example:

```
R> exclude=c("AK","HI")
R> subset=(age=="18-29")
```

It is occasionally useful to indicate the presence or absence of a condition for certain geographic units. In our example, we change the stroke of the states that allow same-sex couples legal marriage. The 'stroke' argument provides a flexible method of doing this. Indeed, several different conditions can be drawn with different stroke attributes ('lty', 'lwd', and 'col'). The list can contain subscript vectors or expressions. Each element of the list is evaluated in the context of the full two-dimensional cross-classified data (that is, the 'data' slot of the `mrp` object. This context makes it easy to use state-level data already included via the 'add' argument.

We use this to specify a thin black line around states with legal marriage, and a slightly thicker but semi-transparent stroke around California, where legal marriage was on, off, stayed, unstayed, appealed, and stayed again. The stroke argument takes this form:

```
R> stroke=list(
     expression(hasmarriage2010==TRUE),
     "CA")
```

4.2 Centering or otherwise Shifting the Scale

The `spplot` methods eventually call `levelplot` which maps values into colors along a range. These colors are called 'regions' in *lattice* parlance and the number and location of cut points between them can be

specified explicitly or calculated from the range of the data. It may be desirable to show values as offsets, for example, from a national poststratified average. Recentering the scale provides an easy way to do that with MRP results:

```
R> center=poststratify(mrp.statelevel)
```

The default number of 'cuts' in the data is 63, and the default range of colors is a 64-valued heatmap. There are a lot of color palettes out there, and they may emphasize different aspects of the data. A popular set are designed by Cynthia Brewer in the *RColorBrewer* package. However, the Brewer Palettes offer at most 11 categories.[2] The in-between ranges can be interpolated, and a continuous version of Brewer palettes is available in the *fBasics* package. We use an *fBasics* version of a Brewer "diverging" palette below.

4.3 The `add.settings` list

Much of the flexibility of *lattice* comes from themes or lists of `trellis.par.settings`. *MRP* includes a theme with some of the values already discussed (the 64-valued range of colors for the regions; the thin-black stroke as the first value of 'superpose.line'). In addition, several other settings are worth mentioning.

add.line The lines drawn around all the other features that are not selected in the 'stroke' list. The default is a 0-width 20% gray line.

reference.line In general reference lines are not drawn on maps (latitude and longitude lines can be drawn but should follow the projection specifications given in the documentation for `spmap.states`. We have appropriated the `reference.line$col` for the color used for NA-valued results. For example, state-by-age-by-income groups for which there is simply not enough data to make a poststratified prediction. The default is a neutral 68% opaque black.

add.text Text settings for the labels for conditioning rows and columns. These are taken from the names of factor levels in the data. The defaults are smaller (`cex=0.7`) and italic. Depending on graphics devices and registered fonts, the 'font' and 'fontface' can also be specified here.

layout.{width|height} By default `spplot,mrp-method` will place left and top strip titles for the entire row and entire column, rather than on each map panel. This is achieved by providing 0s here for every panel in the order drawn except for those lying at the top or left.

[2] When specifying a vector of colors for 'regions' especially when it is a small number, note that the default `mrp` number of cuts is 63; that is, you will always have to specify one fewer cuts than the number of colors provided.

4.4 The Color Key

The 'colorkey' is a list passed down to `levelplot`. The defaults are not bad, but in the case where values are recentered (and computed symmetric about the center) there are often easy changes that need to be made to the colorkey. Our `spplot` leaves all of these up to the user.

5 The Final Map

```
R> print(spplot(mrp.statelevel, state ~ edu+age,
                      subset=TRUE,
                      spmap.states, "STATE", exclude=c("AK","DC","HI"),
                      stroke=list(expression(hasmarriage2010==TRUE),
                        "CA"),
                      center=poststratify(mrp.statelevel), cuts=50,
                      sub=paste("National average:",
                        format(poststratify(mrp.statelevel),digits=2)),
                      add.settings=list(
                        regions=list(col=fBasics:::divPalette(51,"BrBG")),
                        superpose.line=list(col=c("black","#00000066"),lwd=c(.3,1.3))
                        ),
                      colorkey=list(
                        space="bottom",height=.5,width=.5,
                        labels=list(at=c(.04,.34,.64),
                          labels=c("-30%","|","+30%"), cex=.7)
                        )
                      ))
```

Figure 2. The final map shows support for same-sex marriage by age cohort and level of education, centered around the national poststratified average of 34%.