Bar Plot Views: Stacked Versus Separate Columns

Brandon Tao

Oct 3, 2018

Sections

0. Setup

```
library(tidyverse)
```

```
## -- Attaching packages -------- tidyverse 1.2.1 --
```

```
## v ggplot2 2.2.1.9000     v purrr   0.2.4
## v tibble  1.4.1          v dplyr   0.7.4
## v tidyr   0.7.2          v stringr 1.2.0
## v readr   1.1.1          v forcats 0.2.0
```

```
## -- Conflicts ----------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(micromapST)
source('hw.R')
```

1. Access and prepare the data

The data from micromapST package. We can make it available in the workspace with the data() function.

```
data(Educ8thData)
head(Educ8thData)
```

```
##    StAbbrev      State avgscore PctBelowBasic PctAtBasic PctProficie
nt
## AK       AK     Alaska      283            26         39
28
## AL       AL    Alabama      269            40         40
17
## AR       AR   Arkansas      279            30         41
24
## AZ       AZ    Arizona      279            32         37
24
## CA       CA California      273            39         36
19
## CO       CO   Colorado      292            20         37
31
##    PctAdvanced
## AK           7
## AL           3
## AR           5
## AZ           7
## CA           6
## CO          12
```

1.1 Select variables

```
edDf <- select(Educ8thData,
  State, PctBelowBasic:PctAdvanced)
# Shorter: edDf <- Educ8thData[,-c(1,3)]
```

1.2 Rename variables

```
colnames(edDf)
```

```
## [1] "State"         "PctBelowBasic" "PctAtBasic"    "PctProficient"
## [5] "PctAdvanced"
```

```
colnames(edDf) = c("State","Below_Basic",
  "At_Basic","Proficient","Advanced")
colnames(edDf)
```

```
## [1] "State"       "Below_Basic" "At_Basic"    "Proficient"  "Advance
d"
```

1.3 Arrange rows

Arranging data.frame or data.frame rows is often a key step in simplifying the appearance of tables and row-labeled plots. Typically we want similar rows (based on our chosen critera) to appear close together.

Arranging rows will suffice for some tasks such table production and connecting groups of points using geom_path. It is helpful in the production of row labeled plots.

The default arrange function sort order is ascending. When the first sorting variable yields tied cases the function uses additionally specified variables to break ties.

Side note: The arrange function provide poor support for a descending row arrangement. It requires use of of desc() that takes only one variable a argument, so it does not directly support tie breaking. Using the R order function to produce row subscripts is an approach that handles both ascending and descending arrangements with tie breaking.

```
edOrd <- arrange(edDf,
   Below_Basic, At_Basic, Proficient, Advanced)

head(edOrd)
```

```
##           State Below_Basic At_Basic Proficient Advanced
## 1 Massachusetts          14       34         36       15
## 2  North Dakota          15       42         34        8
## 3     Minnesota          17       36         34       13
## 4       Montana          17       37         35       11
## 5    New Jersey          18       35         33       14
## 6       Vermont          18       36         33       13
```

```
# In the table above, The Below_Basic column shows Minnesota
# and Montana tied at 17.  The Minnesota and Montana At_Basic
# values are 36 and 37, respectively, so Minnesota was put first.
```

There are table formatting functions.

```
# Tables
knitr::kable(head( edOrd ),caption = "A knitr kable.")
```

A knitr kable.

State	Below_Basic	At_Basic	Proficient	Advanced
Massachusetts	14	34	36	15
North Dakota	15	42	34	8
Minnesota	17	36	34	13
Montana	17	37	35	11
New Jersey	18	35	33	14
Vermont	18	36	33	13

1.4 Create and add a state name factor with reversed levels

For some graphics, such as bar charts and row labeled plots, we associate unique character strings with axis plotting coordinates. For example we can think plotting

“Small”, “Medium”, and “Large” along the x-axis and centered at implicit locations 1, 2, and 3.

In R we can convert a character string vector into factor. A factor include a levels vector that consists of the unique character strings. The 1st element of this vector is associated with an axis coordinate 1, the 2nd with axis coordinate 2 and so on. We control the plotting order when create the levels vector as shown below.

In row-labeled plots the row-labels are unique. In this example they are state names. Since we have arranged the data.frame rows we can used the states names for the levels vector.

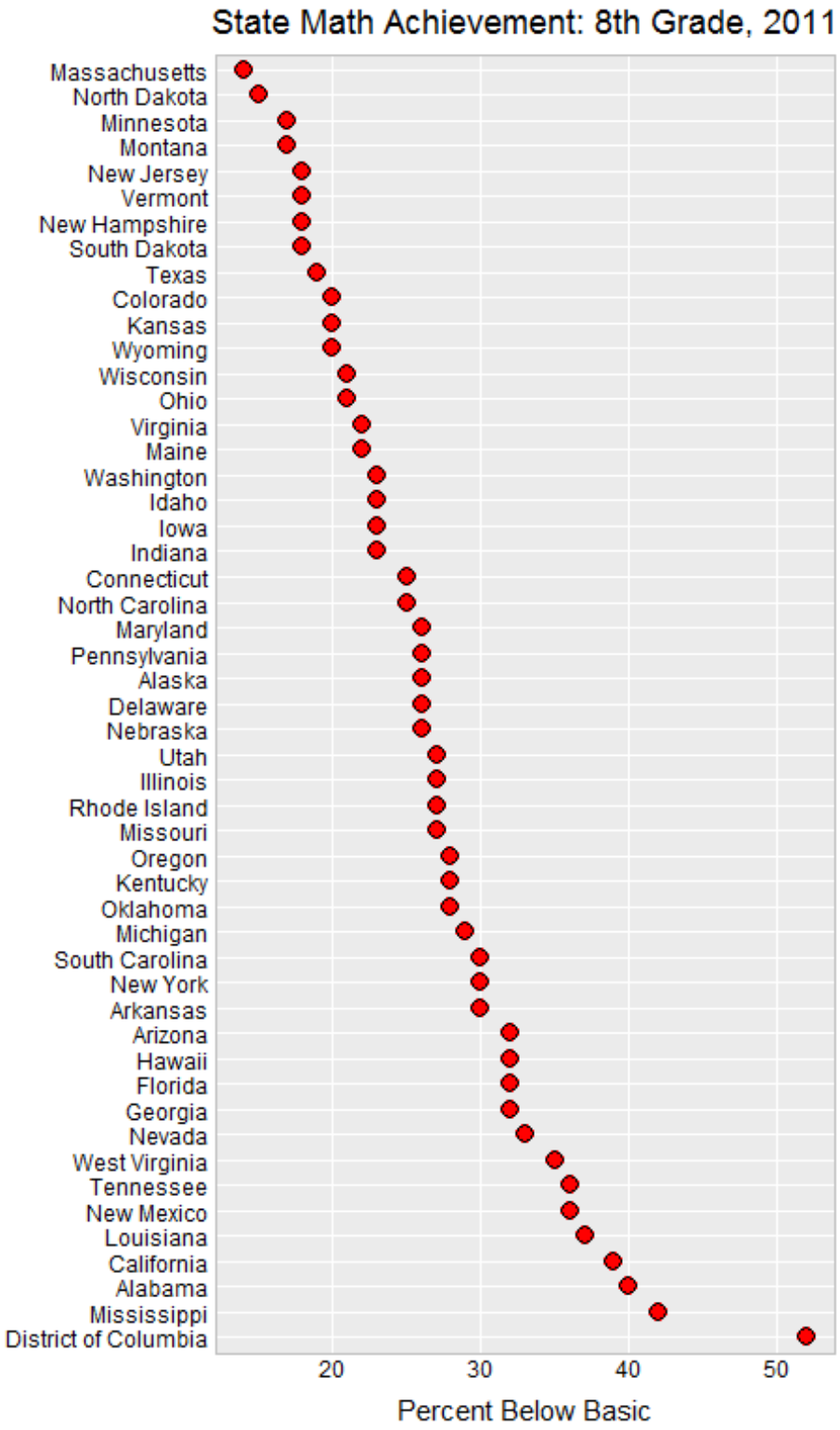
There is one detail to address. As a table Massachusetts will appears in the row. If Massachusetts is the first element in the levels vector, it will be associated with the coordinate 1 on the Y axis and hence plot at the bottom rather than the top. We need to reverse the state name order in the levels vector if want Massachusetts to plot the top.

The table convention with row 1 at top conflicts the graphics convention with “row” 1 at the bottom.

```
nams <- as.character(edOrd$State)
edOrd$State <- factor(nams,levels = rev(nams))
```

1.5 A Dot Plot for Below Basic Achievement Percents

```
# A reusable title
titleAch = 'State Math Achievement: 8th Grade, 2011'

ggplot(edOrd,aes(x = Below_Basic,y = State)) +
geom_point(shape = 21,fill = "red",col = "black",
  size = rel(3.3)) +
labs(x = "Percent Below Basic",y = "",
 title = titleAch) + hw +
 theme(axis.text.y = element_text(size = rel(1.05)))
```



With 51 lines of text the plot needs to use most of a standard page height for easy text reading. The plot height was set to 8 inches. The plot width can be specified.

The use of State Postal Codes instead of names supports further width reduction but a very limited audience knows all the postal codes.

With linked micromaps, those that know the state locations on map and can learn the postal codes.

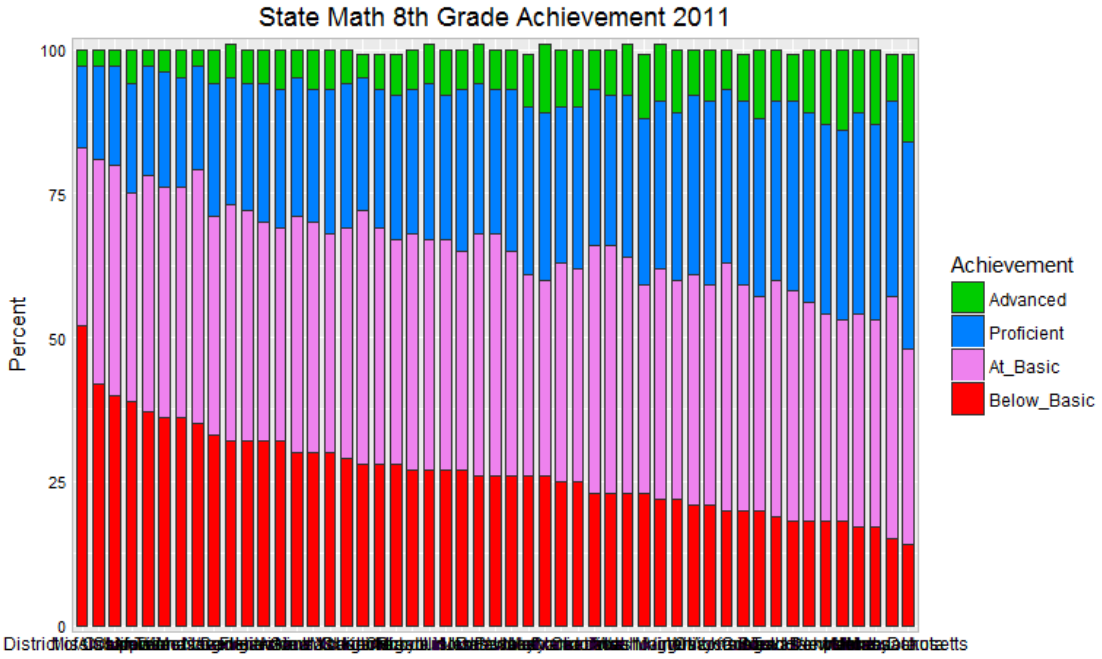
2. Make a vertically stacked bar plot with slanted labels

2.1 We use gather to stack the percents columns into one column called Percent. Gather also handles the bookkeeping, the Achievement column kept track of the previous column membership.

```
edIndexed <- gather(edOrd,
  key = Achievement,
  value = "Percent",
  Advanced:Below_Basic, factor_key = TRUE)

# The factor_key= TRUE
# makes the factor level be in order
# same order as the columns

levels(edIndexed$Achievement)

## [1] "Advanced"    "Proficient"  "At_Basic"    "Below_Basic"
```

2.2 Make a vertically stacked bar plot

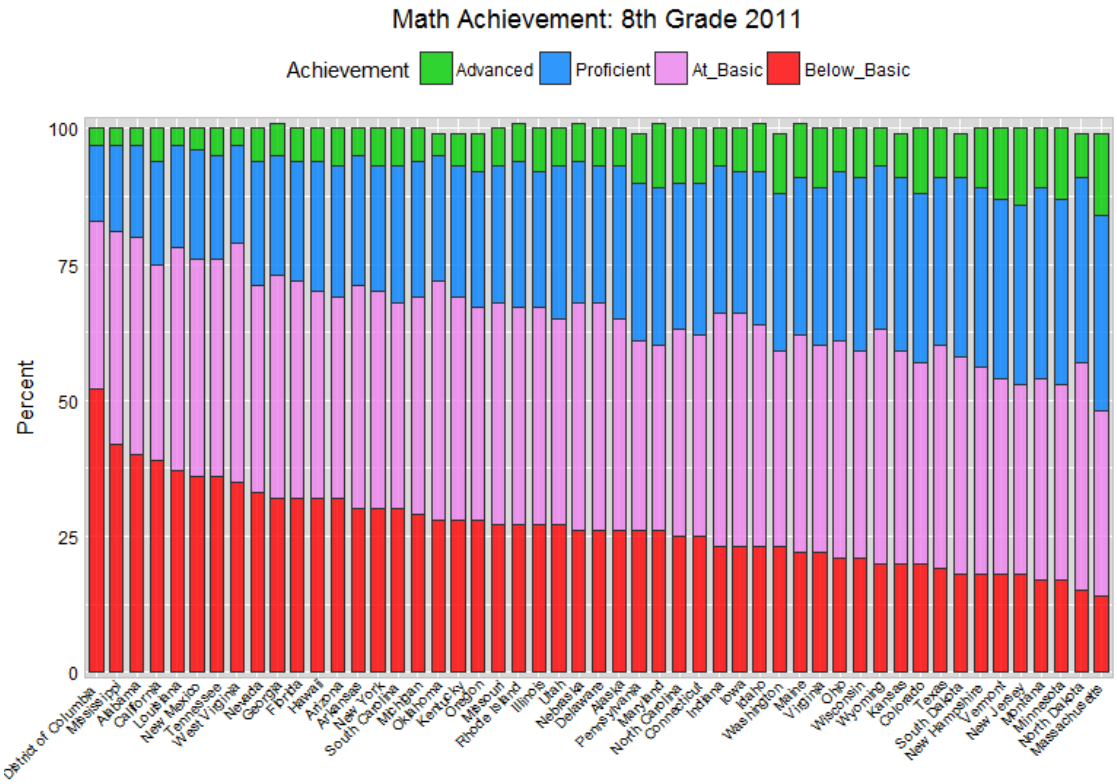
```
ggplot(edIndexed,
  aes(x = State, y = Percent, fill = Achievement)) +
geom_bar(stat = "identity",alpha = 1,
  color = gray(.2),width = .67,size = .3) +
 theme(legend.position = "right",
 axis.text.y = element_text(size = rel(1.05))) + hw +
scale_y_continuous(expand = c(.01,0)) +
scale_fill_manual(
     values = c(rgb(0,.8,0),rgb(0,.5,1),'violet','red')) +
 labs(x = '', y = 'Percent',
 title = 'State Math 8th Grade Achievement 2011')
```



At the top we see that the rounded percents don't total 100. At the bottom we see the state name overplotting mess. We partially address the mess by change the text angle and position used the theme element for axis.text.x. This takes up extra horizontal space so we move legend to top to more horizontal resolution.

2.3 Slanting the x-axis state names

```
ggplot(edIndexed,
  aes(x = State, y = Percent, fill = Achievement)) +
geom_bar(stat = "identity",alpha = .8,
  color = gray(.2),width = .67, size = .3) + hw +
  scale_y_continuous(expand = c(.01,0)) +
 theme(legend.position = "top",
   axis.text.y = element_text(size = rel(1.)),
   axis.text.x =
     element_text(angle = 45, vjust = 1.08, hjust = 1.1,size = rel(.87)
),
   panel.background = element_rect(gray(.85))) +
   scale_fill_manual(
     values = c(rgb(0,.8,0),rgb(0,.5,1),'violet','red')) +
   labs(x = "", y = "Percent",
   title = "Math Achievement: 8th Grade 2011")
```



In ggplot graphics, theme arguments control many appearance features. The hw script uses theme_gray and make modifies a few argument.

Above we make few additional argument modifications. The default legend.position is on the right. We put it on "top" so doesn't take up a lot horizontal space.

For the axis.text.x and axis.text.y arguments we the element_text function and its arguments. Here we control the angle and show the result of experimenting with with horizonal and vertical adjustments.
We modified the panel.background to have a darker gray so the white grid lines would have a higher contrast and be easier to spot when we want to.

In ggplot graphics, scale functions give us control of encoding variable values to rendered aesthetics. Here we change the colors used for the four achievement variables.

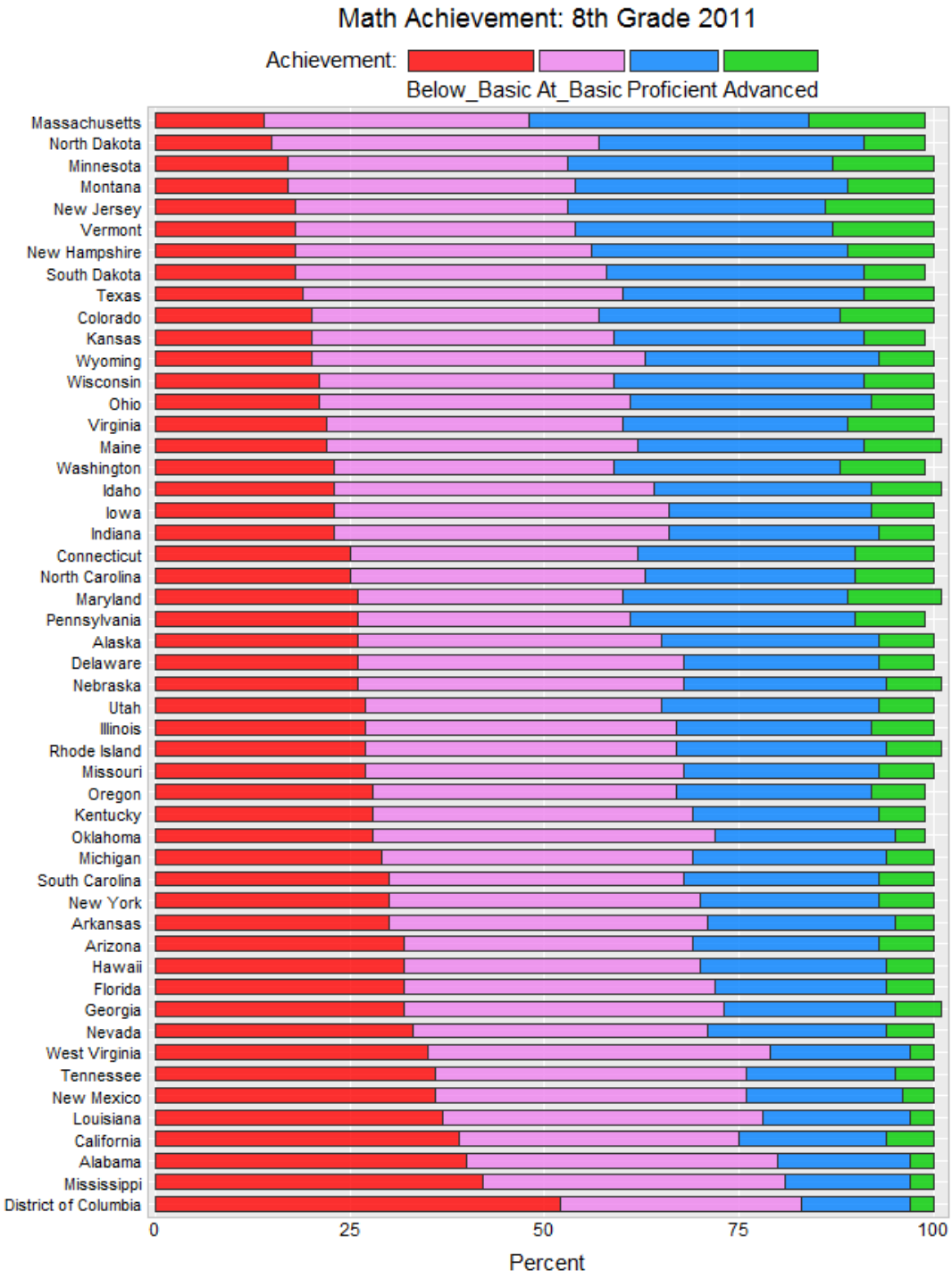
We could refine the plot a bit more but move on horizontal stacked bars that are easier to read.

3. A horizontally stacked bar plot

The function coord_flip() switches the axes.

```
ggplot(edIndexed,
  aes(x = State, y = Percent, fill = Achievement)) +
geom_bar(stat = "identity",alpha = .8,
```

```
  color = gray(.2),width = .67, size = .3) + hw +
  scale_y_continuous(expand = c(.01,0)) +
  scale_fill_manual(
     values = c(rgb(0,.8,0),rgb(0,.5,1),'violet','red')) +
  theme(legend.position = 'top',
    legend.margin = margin(5,0,3,0),
    legend.box.margin = margin(0,-5,-10,-10)) +
  guides(fill = guide_legend(reverse = TRUE,
    title.position = 'left',
    title = "Achievement:",
    title.hjust = .5, title.vjust = 1,
    label.position = "bottom", label.hjust = .5,
    label.theme = element_text(color = 'black',
      angle = 0, size = 11),
    keywidth = 2, keyheight = 0.8)) +
   labs(x = "", y = "Percent",
   title = "Math Achievement: 8th Grade 2011") +

coord_flip()
```



The legend at the top (or bottom) parallels the bar order in the plot. the Achievement class order so it parallels the order in the plot.

4. Constructing a multi-column row labeled bar plot

Below we plan show Achievement variables in separate columns using facet_grid(.~Achievement). However, with the current data.frame the Advanced variable will appear in the left column. The choice here is to put Below_Basic in the left column. In R there may be several ways to reach our objective. Here we just reuse the data.frame building script switch variable order.

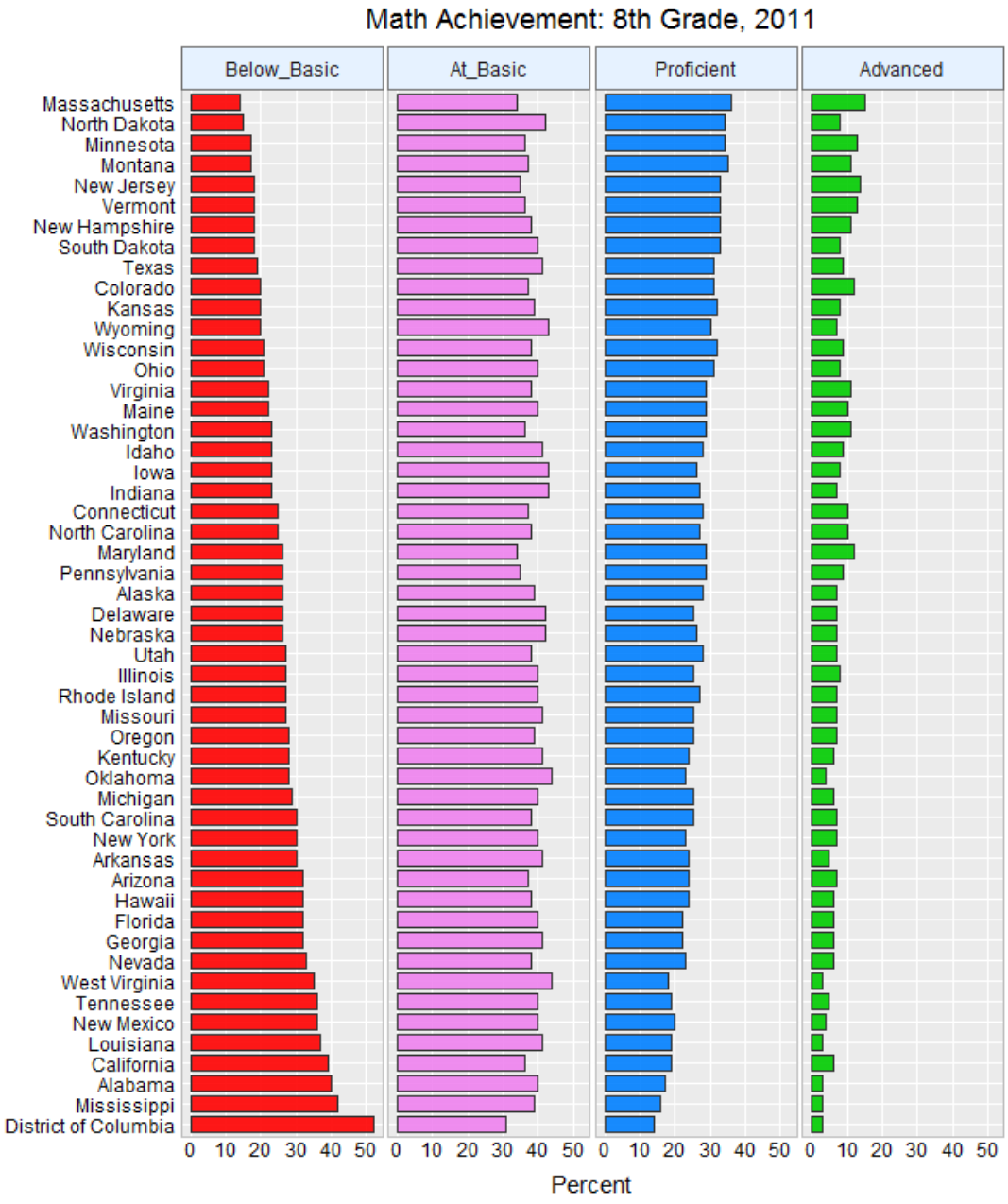
4.1 Rebuild the stacked data.frame the achievment class columns

```
edDfIndRev <- gather(edOrd,
  key = Achievement,
  value = "Percent",
  Below_Basic:Advanced, factor_key=TRUE)
```

Below we also switch the colors to be consistent with previous examples.

4.2 Produce the plot

```
ggplot(edDfIndRev,
  aes(x = State, y = Percent,fill = Achievement)) +
geom_bar(stat = "identity",alpha = .9,
  color = gray(.2),width = .75,size = .4) +
  scale_fill_manual(
     values = c('red','violet',rgb(0,.5,1), rgb(0,.8,0))) +
  facet_grid(.~ Achievement) + hw +
 coord_flip() +
 theme(axis.text.y = element_text(size = rel(1.05)),
       legend.position = 'none') +
 labs(x = "", y = "Percent",
 title = "Math Achievement: 8th Grade, 2011")
```


Math Achievement: 8th Grade, 2011
Below_Basic
At_Basic
Proficient
Advanced
Massachusetts
North Dakota
Minnesota
Montana
New Jersey
Vermont
New Hampshire
South Dakota
Texas
Colorado
Kansas
Wyoming
Wisconsin
Ohio
Virginia
Maine
Washington
Idaho
Iowa
Indiana
Connecticut
North Carolina
Maryland
Pennsylvania
Alaska
Delaware
Nebraska
Utah
Illinois
Rhode Island
Missouri
Oregon
Kentucky
Oklahoma
Michigan
South Carolina
New York
Arkansas
Arizona
Hawaii
Florida
Georgia
Nevada
West Virginia
Tennessee
New Mexico
Louisiana
California
Alabama
Mississippi
District of Columbia
0 10 20 30 40 50
Percent