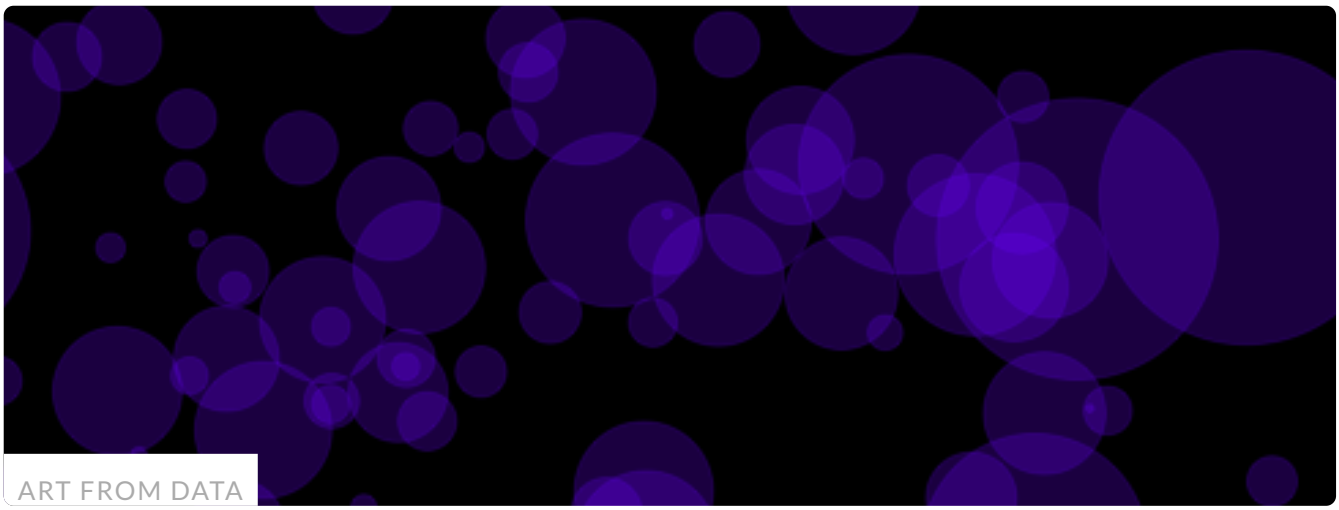
MENU

THE R GRAPH GALLERY



#267 REORDER A VARIABLE IN GGPLOT2

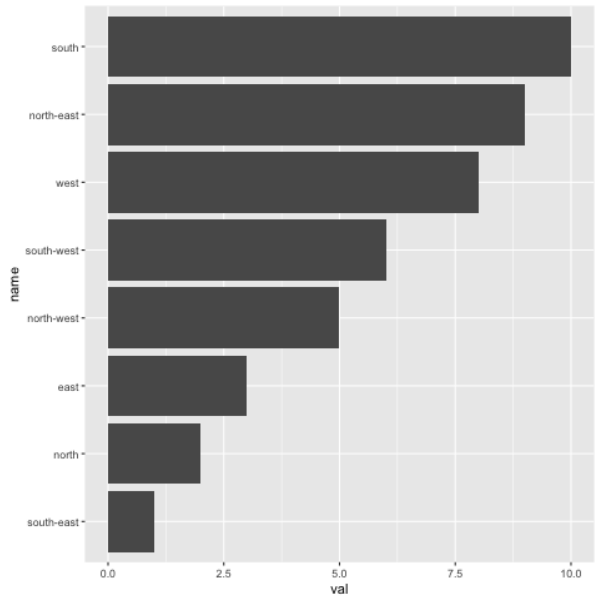
Reordering groups in a ggplot2 graphic is sometimes seen as a struggle. This is due to the fact that ggplot2 takes into account the order of the factor levels, not the order you observe in your data frame. You can sort your input data frame with sort() or arrange(), it will never have any impact on your ggplot2 graphic.

This post explains how to reorder the level of your factor through several examples. To go further, I strongly advise to read the dedicated chapter of the book R for data science. Examples are based on 2 dummy datasets:

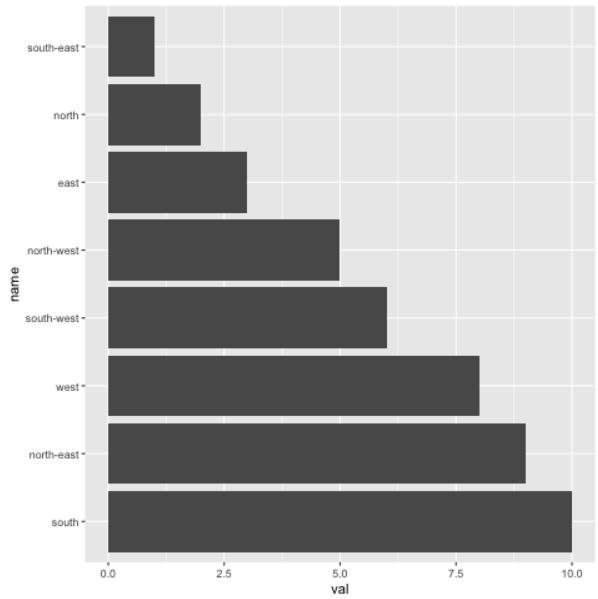
```
# data: one value per group
data <- data.frame(
  name=c("north","south","south-east","north-west","south-west","north-east","west","east
  val=sample(seq(1,10), 8 )
)

# mpg: several values per group
mpg
```

The Forecats library

The Forecast library is a library from the tidyverse specially made to handle factors in R. It provides a suite of useful tools that solve common problems with factors. The fact_reorder function allows to reorder the factor (data$name for example) following the value of another column (data$val here).



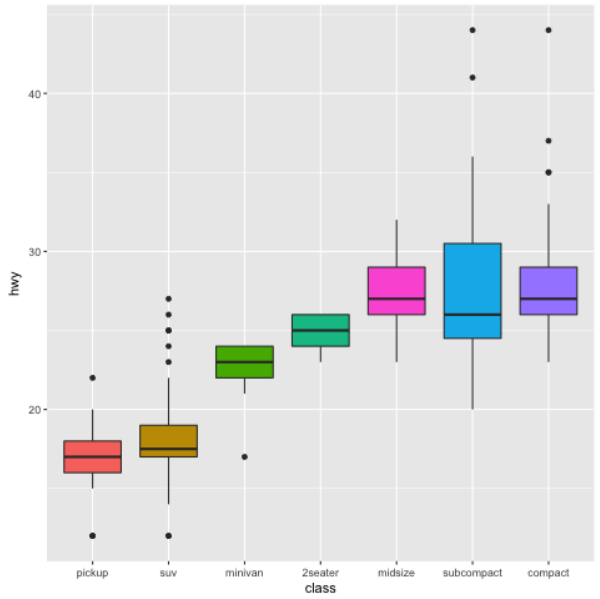
#267 Reorder barplot



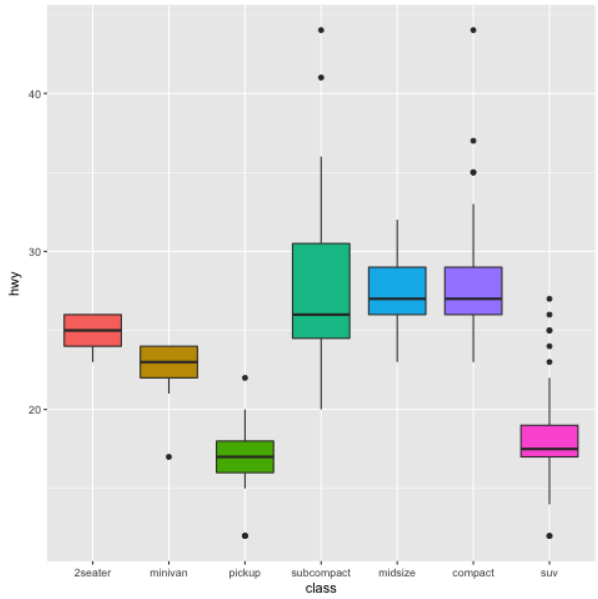
#267 Reorder barplot

```r
# Reorder following the value of another column:
data %>%
  mutate(name = fct_reorder(name, val)) %>%
  ggplot( aes(x=name, y=val)) +
    geom_bar(stat="identity") +
    coord_flip()

# Reverse side
data %>%
  mutate(name = fct_reorder(name, desc(val))) %>%
  ggplot( aes(x=name, y=val)) +
  geom_bar(stat="identity") +
  coord_flip()
```

If you have several values per level of your factor, you can specify which function to apply to determine the order. The default is to use the median, but you can use the number of data points per group to make the classification:



#267 Reorder boxplot

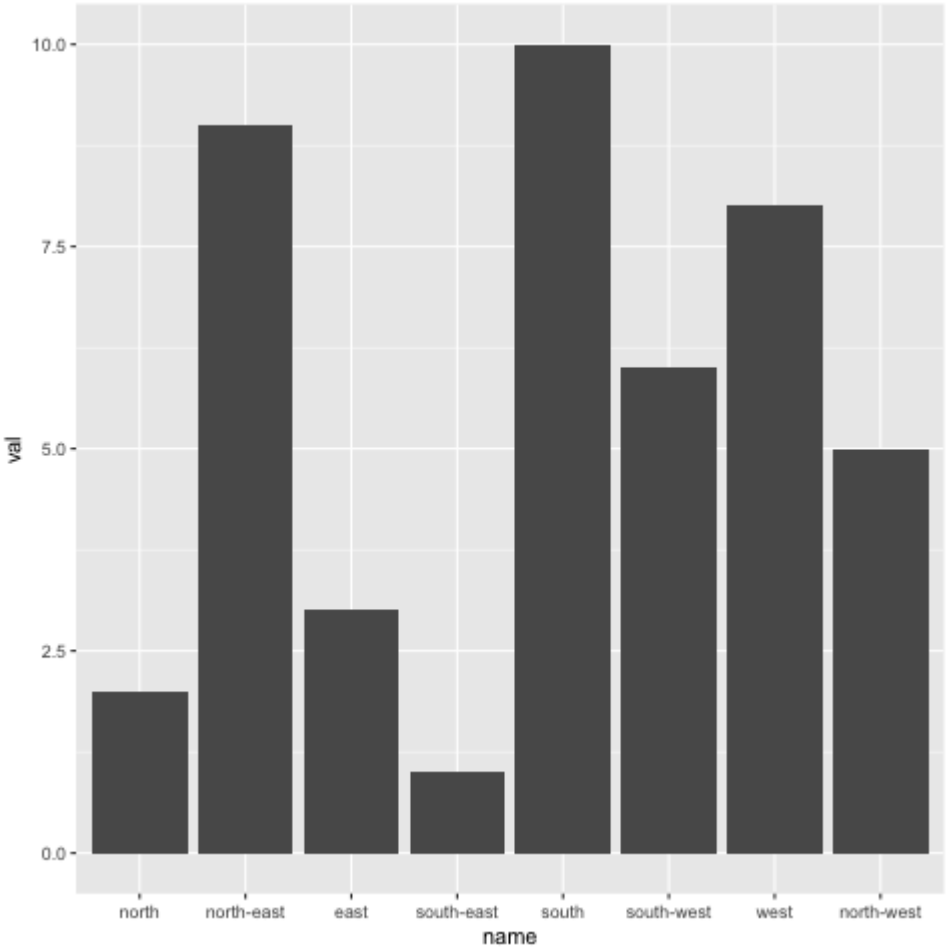


#267 Reorder boxplot

```
# Using median
mpg %>%
  mutate(class = fct_reorder(class, hwy, fun=median)) %>%
  ggplot( aes(x=reorder(class, hwy), y=hwy, fill=class)) +
    geom_boxplot() +
    xlab("class") +
    theme(legend.position="none")

# Using number of observation per group
mpg %>%
  mutate(class = fct_reorder(class, hwy, fun=length )) %>%
  ggplot( aes(x=class, y=hwy, fill=class)) +
  geom_boxplot() +
  xlab("class") +
  theme(legend.position="none")
```

The last common operation is to provide a specific order to your levels, you can do so using the fct_relevel function:

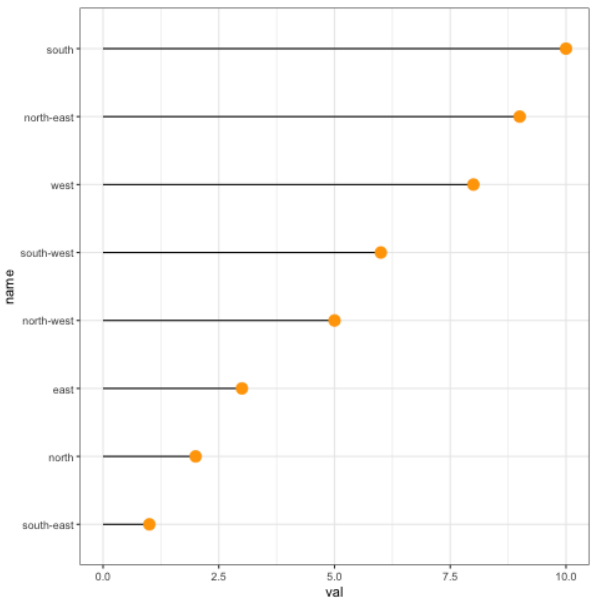
```
# Reorder following a precise order
data %>%
  mutate(name = fct_relevel(name, "north", "north-east", "east", "south-east", "south", "
  ggplot( aes(x=name, y=val)) +
    geom_bar(stat="identity")
```



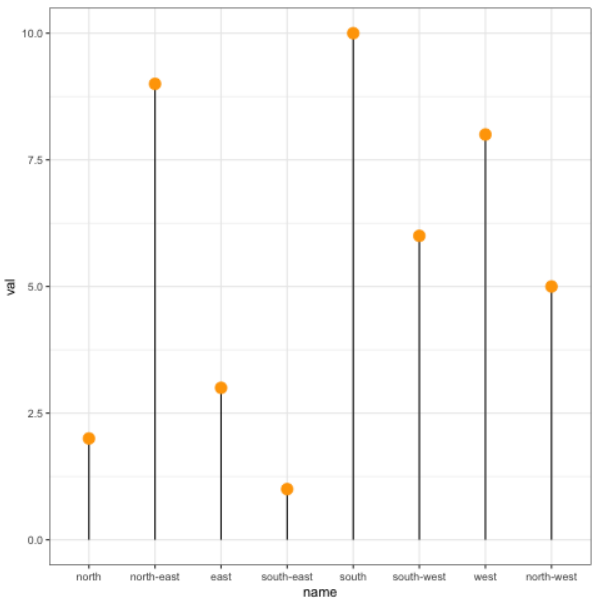
Using dplyr only

The mutate function of dplyr allows to create a new variable or modify an existing one. It is possible to use it to recreate a factor with a specific order. Here are 2 examples:

- The first use arrange() to sort your data frame, and reorder the factor following this desired order.
- The second specify a custom order for the factor giving the levels one by one.



#267 Reorder factor in lollipop

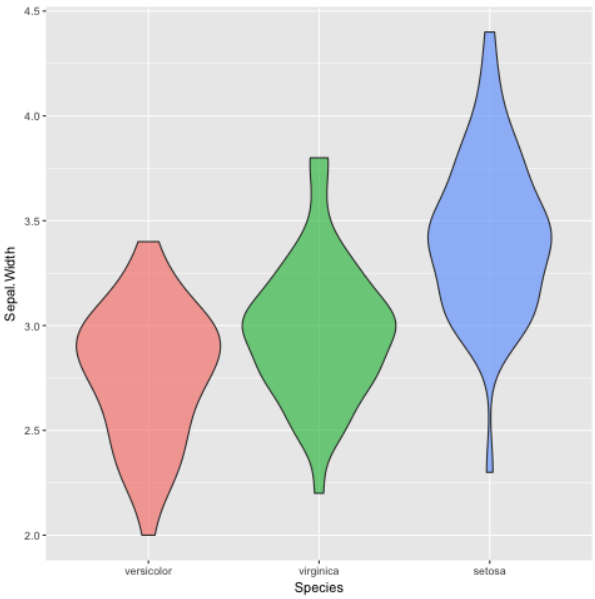


#267 Reorder factor in lollipop

```
data %>%
  arrange(val) %>%
  mutate(name=factor(name, levels=name)) %>%
  ggplot( aes(x=name, y=val)) +
```

```
    geom_segment( aes(xend=name, yend=0)) +
    geom_point( size=4, color="orange") +
    coord_flip() +
    theme_bw()

data %>%
  arrange(val) %>%
  mutate(name = factor(name, levels=c("north", "north-east", "east", "south-east", "sout
  ggplot( aes(x=name, y=val)) +
  geom_segment( aes(xend=name, yend=0)) +
  geom_point( size=4, color="orange") +
  theme_bw()
```

Using base R: the reorder function



In case your an unconditional user of the good old R, here is how to control the order using the reorder function:

```
# iris dataset
iris

# reorder is close to order, but is made to change the order of the factor levels.
iris$Species = with(iris, reorder(Species, Sepal.Width, mean))

# Now you can plot
ggplot(iris, aes(x=Species, y=Sepal.Width, fill=Species)) +
  geom_violin(alpha=0.6) +
  theme(legend.position="none")
```

Related