Parallel Coordinate Plots in ggplot2

Prepare all packages

```
# install.packages("ggplot2")
# install.packages("GGally")
# install.packages("ggparallel")
library("ggplot2")
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
library("GGally")
```

```
## Warning: package 'GGally' was built under R version 3.5.3
```

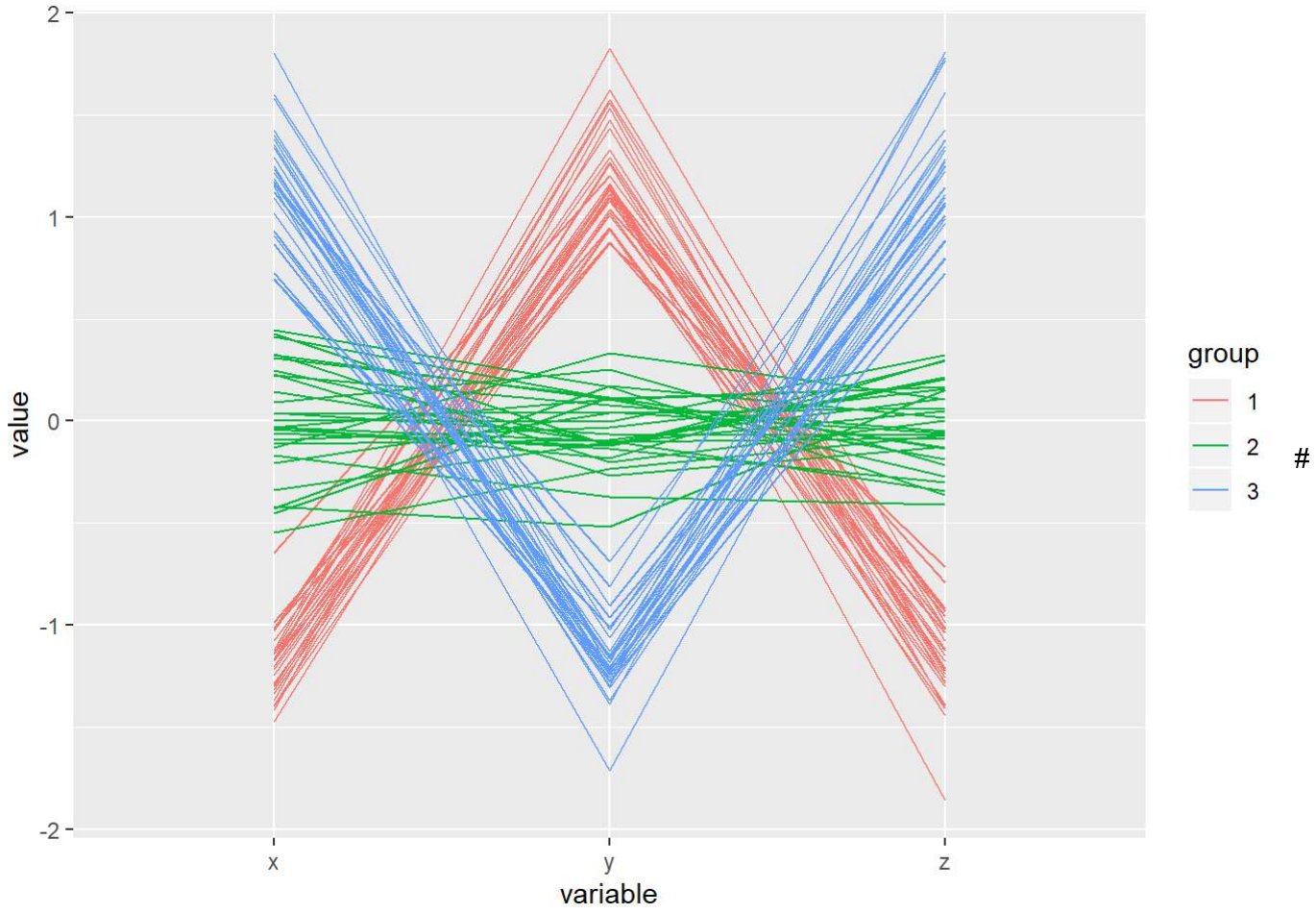
```
library("ggparallel")
```

```
## Warning: package 'ggparallel' was built under R version 3.5.3
```

Simple Example

In the simple example of ggparcoord, I used generating data to show the visualization.

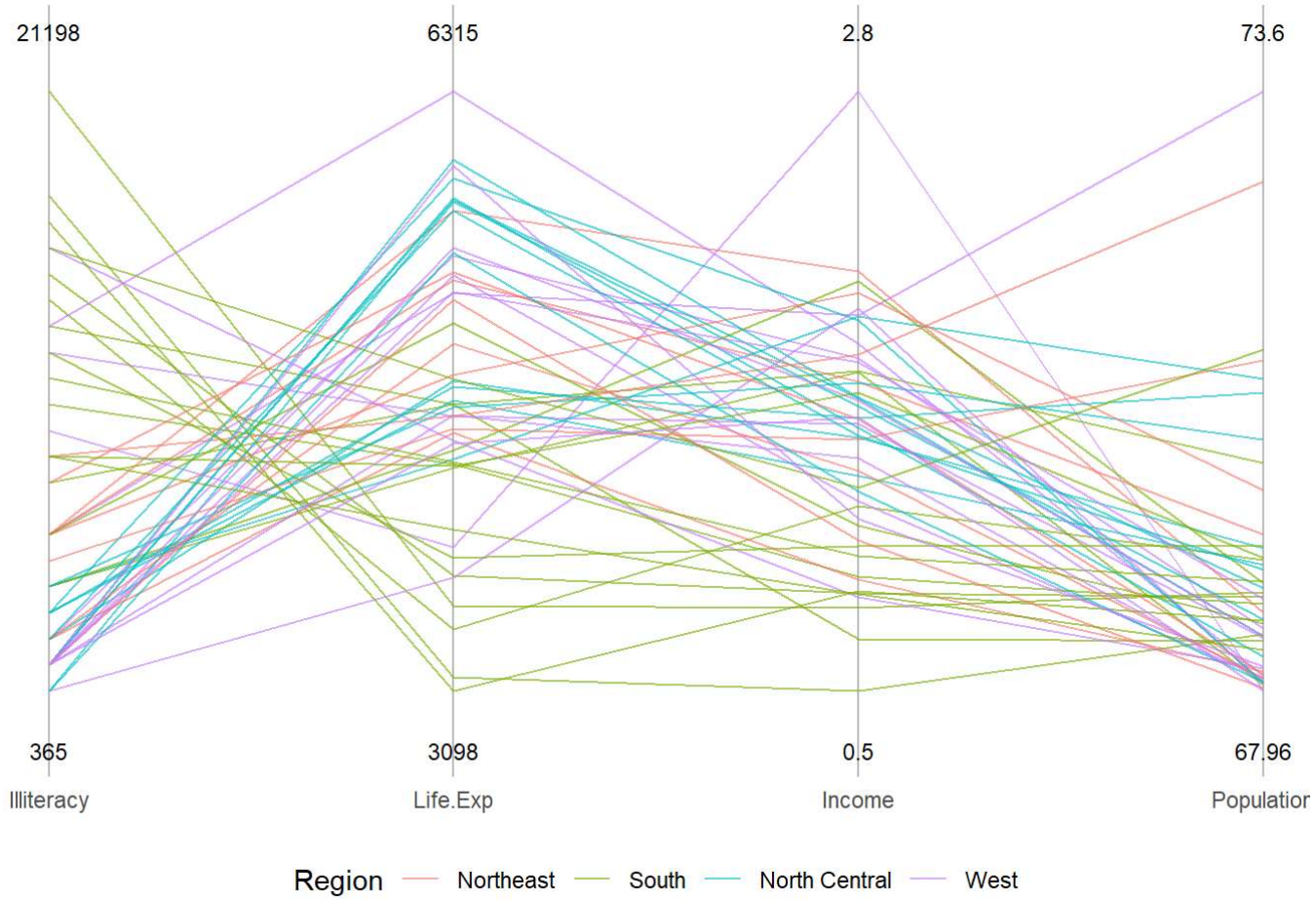
```
set.seed(100)

# generate data
k <- rep(1:3, each=30)
x <- k + rnorm(mean=10, sd=.2,n=90)
y <- -2*k + rnorm(mean=10, sd=.4,n=90)
z <- 3*k + rnorm(mean=10, sd=.6,n=90)

# change it into dataframe
dat <- data.frame(group=factor(k),x,y,z)

# do the simple visualization using ggparcoord
ggparcoord(dat,columns=2:4,groupColumn = 1)
```



Real Data Example

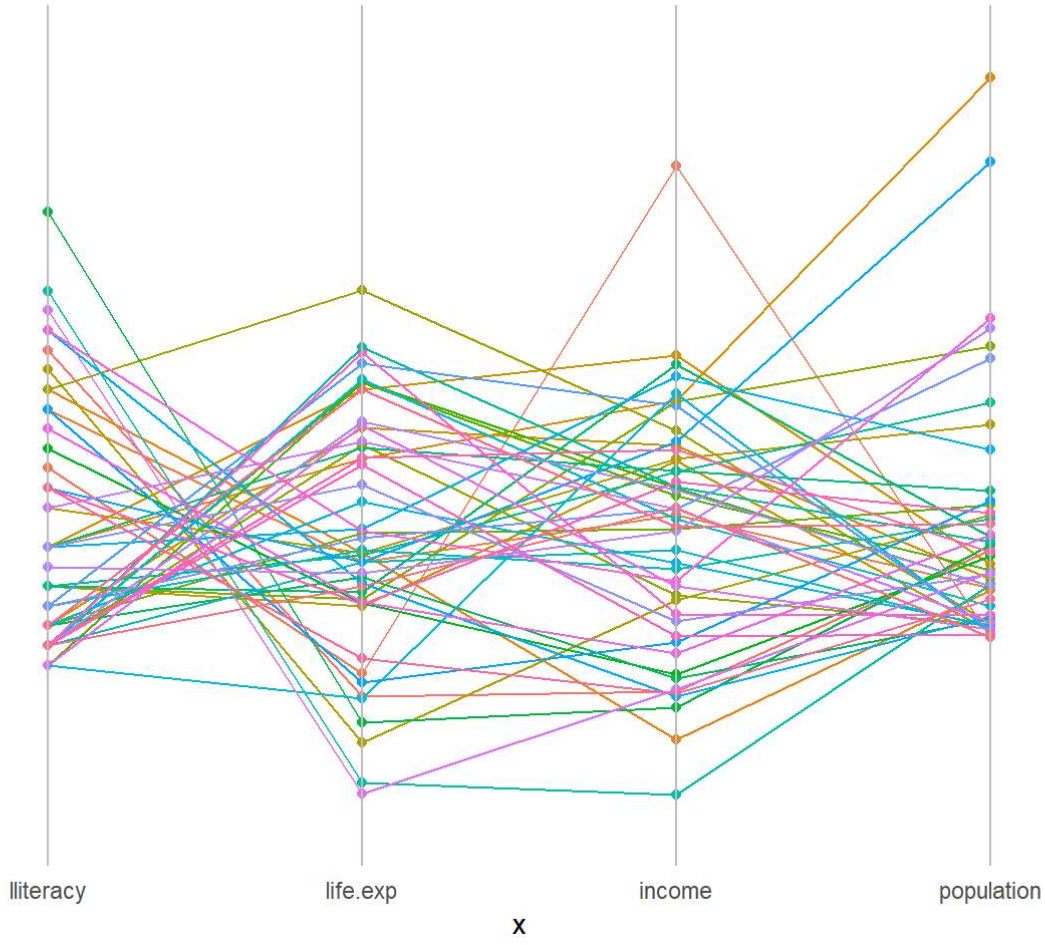
```
# Load required packages
require(GGally)

# Load datasets
data(state)

#change to dataframe
df <- data.frame(state.x77,
                 State = state.name,
                 Abbrev = state.abb,
                 Region = state.region,
                 Division = state.division)

# Generate basic parallel coordinate plot
p <- ggparcoord(data = df,
                columns = 1:4,
                groupColumn = 11,
                order = "anyClass",
                showPoints = FALSE,
                alphaLines = 0.6,
                shadeBox = NULL,
                scale = "uniminmax" # try "std" also
)

# Start with a basic theme
p <- p + theme_minimal()

# Decrease amount of margin around x, y values
p <- p + scale_y_continuous(expand = c(0.02, 0.02))
p <- p + scale_x_discrete(expand = c(0.02, 0.02))

# Remove axis ticks and labels
p <- p + theme(axis.ticks = element_blank())
p <- p + theme(axis.title = element_blank())
p <- p + theme(axis.text.y = element_blank())

# Clear axis lines
p <- p + theme(panel.grid.minor = element_blank())
p <- p + theme(panel.grid.major.y = element_blank())

# Darken vertical lines
p <- p + theme(panel.grid.major.x = element_line(color = "#bbbbbb"))

# Move label to bottom
p <- p + theme(legend.position = "bottom")

# Figure out y-axis range after GGally scales the data
min_y <- min(p$data$value)
max_y <- max(p$data$value)
pad_y <- (max_y - min_y) * 0.1

# Calculate label positions for each veritcal bar
lab_x <- rep(1:4, times = 2) # 2 times, 1 for min 1 for max
```

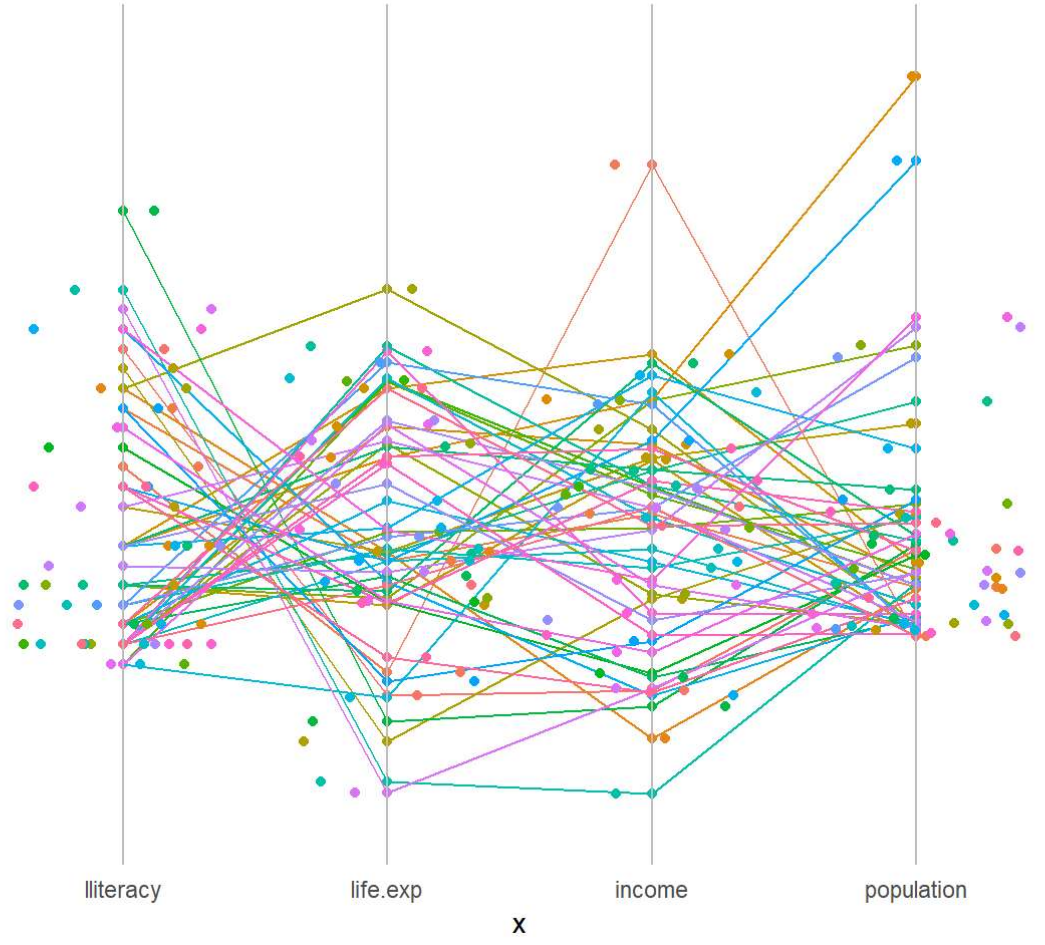
```
lab_y <- rep(c(min_y - pad_y, max_y + pad_y), each = 4)

# Get min and max values from original dataset
lab_z <- c(sapply(df[, 1:4], min), sapply(df[, 1:4], max))

# Convert to character for use as labels
lab_z <- as.character(lab_z)

# Add labels to plot
p <- p + annotate("text", x = lab_x, y = lab_y, label = lab_z, size = 3)

# Display parallel coordinate plot
print(p)
```



```
# Load required packages
require(GGally)

# Load datasets
data(state)

# Define a function
ggplot.palleral <- function(x.num,col.num,n){

    # Prepare data we will use
    mini_labels = c(1:x.num)
    labels = rep(mini_labels,length(col.num))
    colour = gl(col.num,x.num,x.num * col.num)

    # Scale all data
    lliteracy.scale = scale(state.x77[,3])
    life.exp.scale = scale(state.x77[,4])
    income.scale = scale(state.x77[,2])
    population.scale = scale(state.x77[,1])

    # Change to dataframe
    df.temp = data.frame(lliteracy = lliteracy.scale,
                         life.exp = life.exp.scale,
                         income = income.scale,
                         population = population.scale)

    # Change data to vector
    data = c(t(as.matrix(df.temp)))

    # Change data to final dataframe and show that
    df_ggplot <- data.frame(x = labels,
                            y = data,
                            colour = colour)
    df_ggplot

    # General base of ggplot
    p <- ggplot(df_ggplot, aes(x=x,
                               y=y,
                               colour=colour)) + geom_point() + geom_line()

    # Set white background
    p <- p + theme_bw()

    # Delete the background line
    p <- p + theme(panel.grid =element_blank())

    # Delete the outline
    p <- p + theme(panel.border = element_blank())

    # Delete the legend
    p <- p + theme(legend.position="none")

    # Reset the axis
```

```
    p <- p + theme(axis.ticks = element_blank()) + theme(axis.title.y = element_blank()) + scale_y_discre
te(breaks = NULL)

    # Add lines in the graph
    p <- p + geom_vline(xintercept = c(1:x.num)
                        ,colour = "gray")

    # Change the name of the axis
    p <- p + scale_x_discrete(limits=c("lliteracy","life.exp","income","population"))

    # The harder one jitter
    p_jitter <- p + geom_jitter()

    # Print the plot of middle and hard level
    print(p)
    print(p_jitter)
}

# Input of function
x.num = 4 # how many number in x
col.num = length(state.name) # how many columns in y
n = length(state.name) * x.num # sum of all

# Use the function
ggplot.palleral(x.num,col.num,n)
```


lliteracy
life.exp
income
population
x


lliteracy
life.exp
income
population
x