# R GRAPHICS

Ozan Bakış[1]

[1]Bahcesehir University, Department of Economics and BETAM

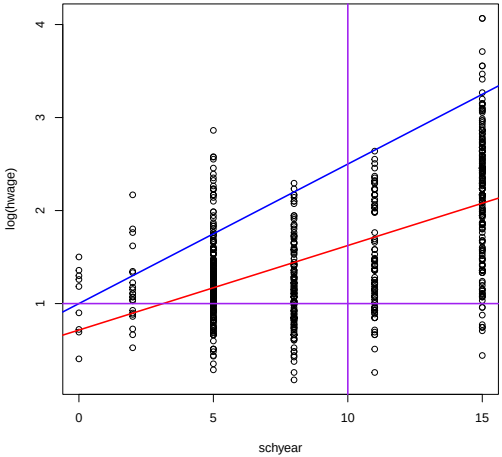
# Outline

# Load data I

```
f_url = "https://github.com/obakis/econ_data/raw/master/hls2011.rds"
download.file(url = f_url, destfile = "hls2011.rds", mode="wb")
hls = readRDS("hls2011.rds")
hls$educ = factor(hls$educ,labels=c("Ill","Lit","PS","MS","HS","Col"))
```

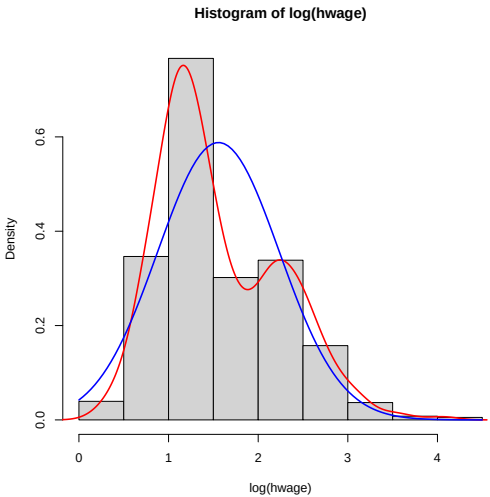
# plot()l

```r
plot(log(hwage)~schyear, data=hls)
abline(lm(log(hwage)~schyear,
    hls), lwd=2, col="red")
#a, b :intercept and slope
abline(a=1,b=0.15, lwd=2, col="blue")
# h:horizontal line, v:vertical line
abline(h=1,v=10,lwd=2, col="purple")
```



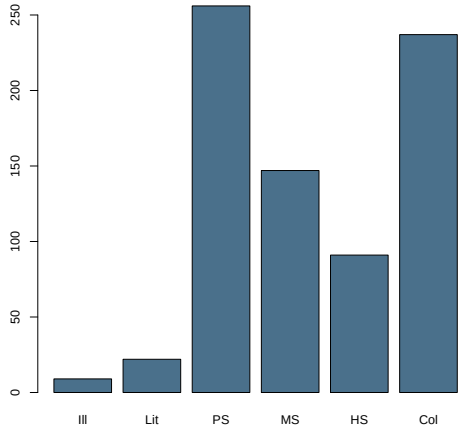
# hist()|

```r
with(hls, hist(log(hwage),
                    freq=FALSE))
# See also: freq=TRUE
with(hls,
     lines(density(log(hwage)),
           col = "red", lwd=2))
m=mean(log(hls$hwage))
sd = sd(log(hls$hwage))
curve(dnorm(x,m,sd),add=TRUE,
      col = "blue", lwd=2)
```



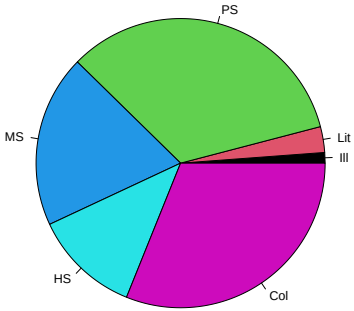
# barplot()|

```
tab = table(hls$educ)
barplot(tab, col="skyblue4")
```



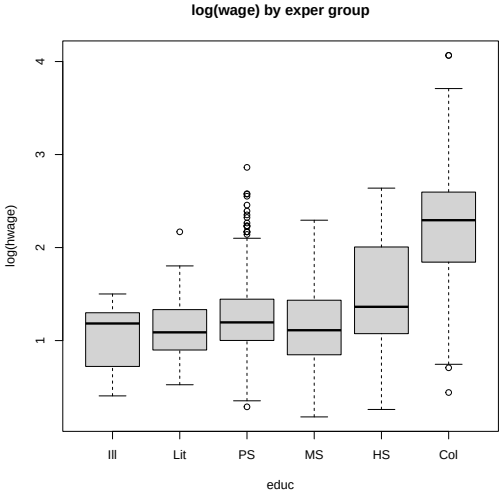
# pie()|

```
pie(tab, col=1:6)
```



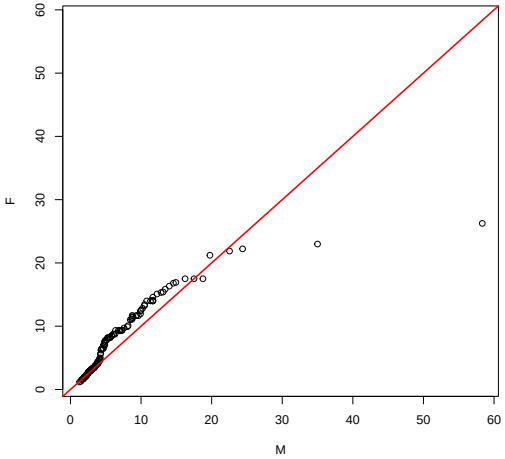
# boxplot()|

```
boxplot(log(hwage) ~ educ,
        data=hls,
    main="log(wage) by exper group")
```



# qqplot()|

```
mwage = subset(hls,
               female==0)$hwage
fwage = subset(hls,
               female==1)$hwage
w_range = range(hls$hwage)
qqplot(mwage,fwage, xlim=w_range,
     ylim=w_range, xlab="M", ylab="F")
abline(a=0,b=1,lwd=2,col="red")
```



# Graphical parameters I

**Modifications:** `plot()` has many arguments, including

- `type`: modify plot type, e.g., points (`type = "p"`, default), lines (`type = "l"`), both (`type = "b"`), stair steps (`type = "s"`).
- `main`, `xlab`, `ylab`: modify title and axis labels.
- Further graphical parameters (see `?par`) can be passed to `plot()` or set separately via `par()`.
- `col`: set color(s).
- `xlim`, `ylim`: adjust plotting ranges.
- `pch`: modify the plotting character for points.
- `cex`: corresponding character extension.

# Graphical parameters II

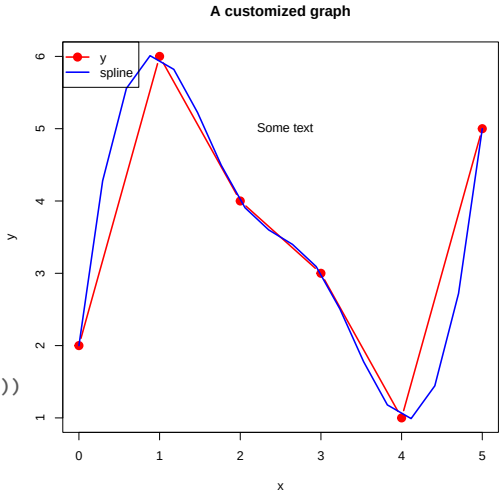
- `lty`, `lwd`: line type and width.
- `cex.lab`, `cex.axis`, `cex.`foo: size of labels, axis ticks, etc.

# Graphical parameters I

| Argument | Description |
|---|---|
| `axes` | should axes be drawn? |
| `bg` | background color |
| `cex` | size of a point or symbol |
| `col` | color |
| `las` | orientation of axis labels |
| `lty, lwd` | line type and line width |
| `main, sub` | title and subtitle |
| `mar` | size of margins |
| `mfcol, mfrow` | array defining layout for several graphs on a plot |
| `pch` | plotting symbol |
| `type` | types (see text) |
| `xlab, ylab` | axis labels |
| `xlim, ylim` | axis ranges |
| `xlog, ylog, log` | logarithmic scales |

# text() and lines() I

```
set.seed(12)
x=0:5; y=sample(6)
plot(y~x, type="b",col="red",
     lwd=2, pch=20,cex=2,
  main = "A customized graph")
text(3.0, 5.0, "Some text",
     pos = 2)
lines(spline(x,y), col="blue",
      lwd=2)
legend("topleft", col=c("red","blue"),
       lty=1,lwd=2,pt.cex=c(2,NA),
     pch=c(20,NA),legend=c("y","spline"))
```



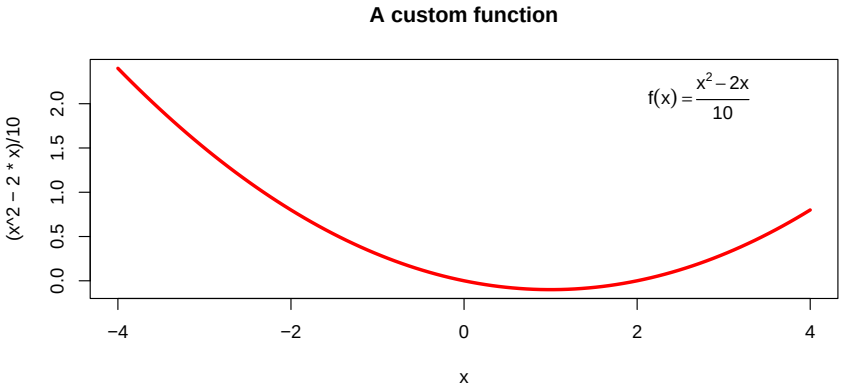
# Mathematical annotation of plots I

**Overview:** `?plotmath` and `demo("plotmath")`.
**Syntax:** Somewhat similar to LaTeX.
**Illustration:** Let us plot the following function for $-4 \leqslant x \leqslant 4$.

$$f(x) = \frac{x^2 - 2x}{10}$$

```
curve((x^2-2*x)/10, from = -4, to = 4, col = "red", lwd = 3,
  main = "A custom function")
text(2.0, 2.0, expression(f(x) == frac(x^2-2*x, 10)), pos = 4)
# pos: 1(below),2(left),3(above) and 4(right), of the specified coord.
```

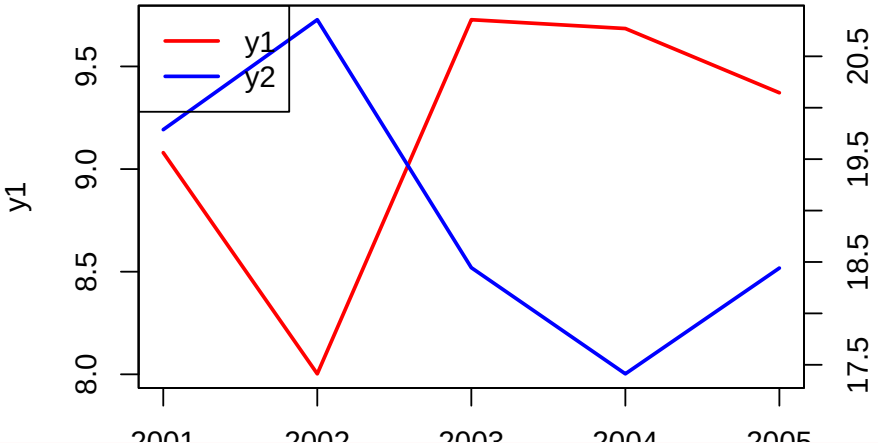
# Mathematical annotation of plots I



# Double Y axes I

```r
## See also,
## stackoverflow.com/questions/6142944/how-can-i-plot-with-2-different-y-axes
x <- 2001:2005
y1 <- rnorm(5,10,1)
y2 <- rnorm(5,20,2)
plot(x,y1,type="l",col="red",lwd=2)
par(new=TRUE)
plot(x, y2,type="l",col="blue",lwd=2,
     xaxt="n",yaxt="n",xlab="",ylab="")
axis(4)
mtext("y2",side=4,line=3)
legend("topleft",col=c("red","blue"),
       lty=1,lwd=2,legend=c("y1","y2"))
```

# Double Y axes II



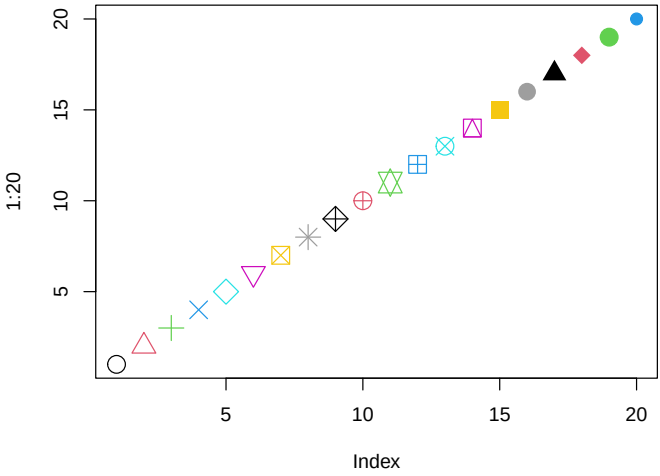
# Exporting graphics I

We can save graphics in various formats including **PDF**, **PS**, **EPS**, **PNG**, **JPG**, **BMP**, **WMF**, **SVG**. In R language it is known as starting a device driver. For instance a **PDF** graphic may be created by

```
pdf("myfile.pdf", height = 5, width = 6)
plot(1:20, pch = 1:20, col = 1:20, cex = 2)
dev.off()
```

After graphic is done we should terminate the device driver by issuing the command `dev.off()`.

# Exporting graphics II



# Outline

# ggplot2 I

- The main function is `ggplot()`. The key components of this function are data and aesthetics (`aes`). The aesthetics specify the variables to be plotted and the optional arguments regarding plotting size, shape color, etc.

- To below command specifies the data and the variables to be plotted.
  `ggplot(data = my_df, aes(x = my_x, y = my_y))`

- However we may have many ”geom”s at te same time (points, line,bars etc.) The most widely used ones are
  $\Rightarrow$ `geom_point` used for scatter plots and dot plots.
  $\Rightarrow$ `geom_line` for lines.

- For adding geoms to a plot we need to use `+` operator.

# ggplot2 II

- In examples below, we use the gapminder data. There are six variables: country, continent, year, lifeExp (life expectancy at birth), pop (total population), gdpPercap (per-capita GDP).
- The per-capita GDP is in units of 2005 international dollars.

# ggplot2 III

```
#install.packages("gapminder")
library(gapminder)
library(ggplot2)
library(dplyr)
library("gridExtra")
```

First few observations:

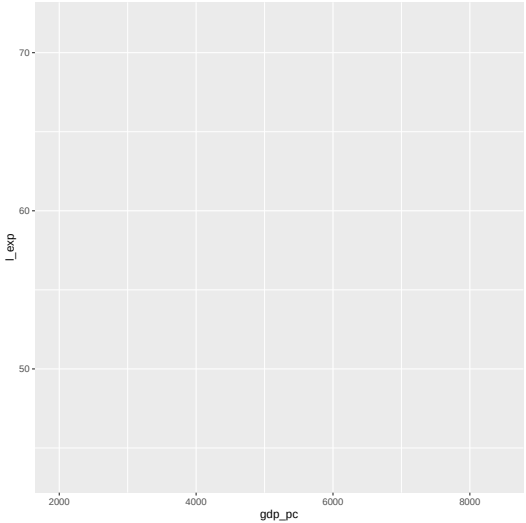
```
gm = gapminder
head(gm,4)
```

```
## # A tibble: 4 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
```

```
colnames(gm)=c("ctry","contin","yr","l_exp","pop","gdp_pc")
```

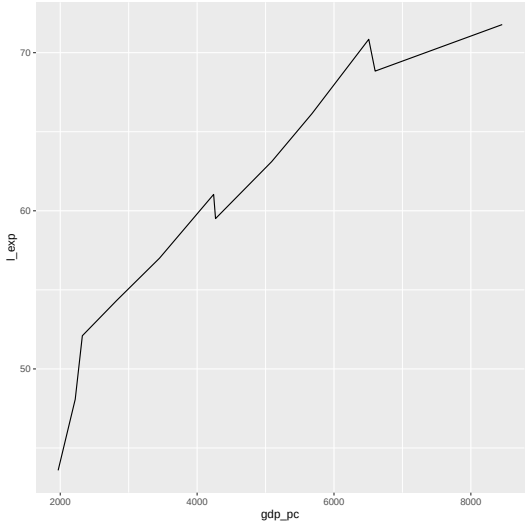
# ggplot2 IV

```
gm_t = filter(gm, ctry=="Turkey")

g1 = ggplot(gm_t,
            aes(x=gdp_pc, y=l_exp))
print(g1)
```



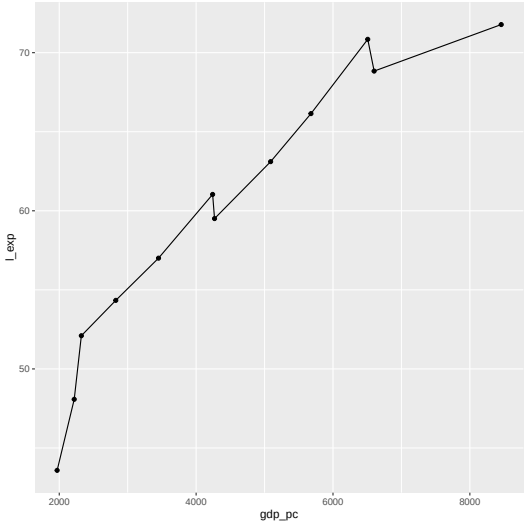
# ggplot2 V

```
g2 = g1 + geom_line()
print(g2)
```



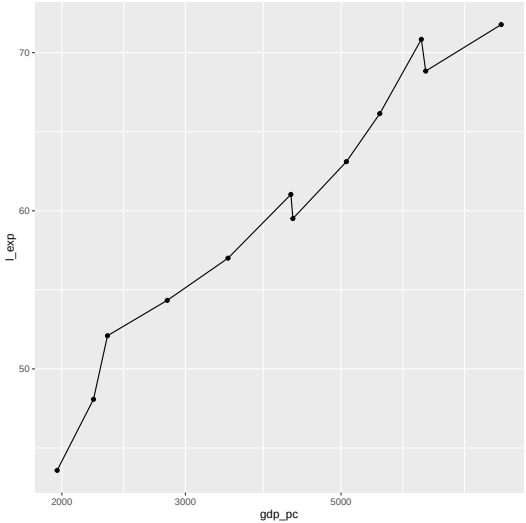
# ggplot2 VI

```
g3 = g1 + geom_line() + geom_point()
print(g3)
```



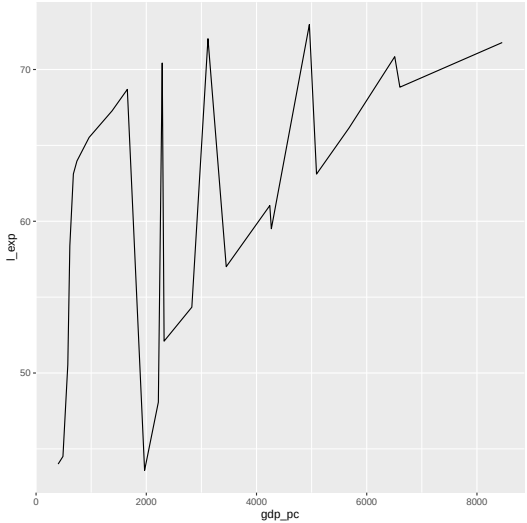
# ggplot2 VII

```
g4 = g1 + geom_line() +
  geom_point() + scale_x_log10()
print(g4)
```



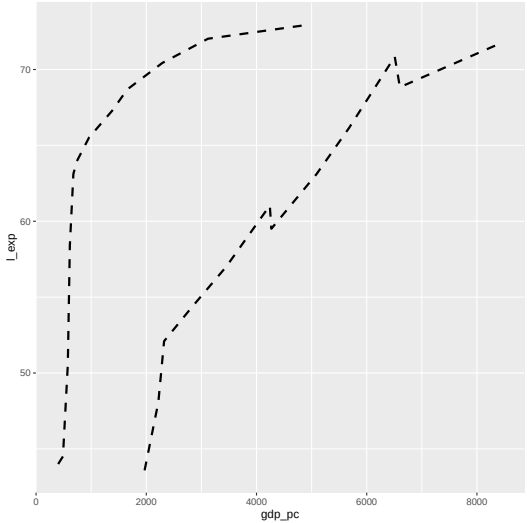
# ggplot2 VIII

```
gm_ct = filter(gm,
  ctry=="China" | ctry=="Turkey")

p_ct = ggplot(gm_ct,
          aes(x=gdp_pc, y=l_exp))
g5 = p_ct + geom_line()
print(g5)
```



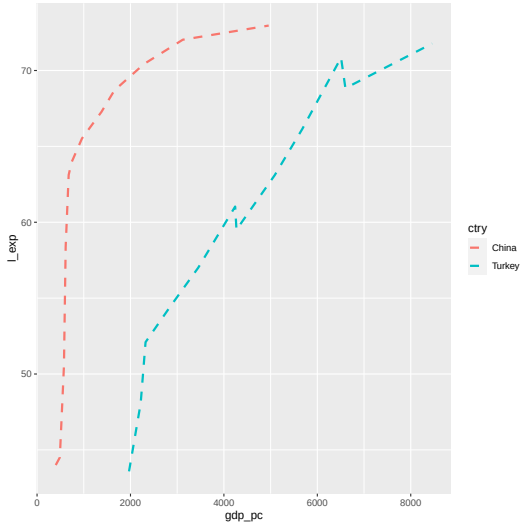
# ggplot2 IX

```
## ctry is defined as group now
p2_ct=ggplot(gm_ct,
        aes(x=gdp_pc, y=l_exp,
            group=ctry))
#lty=line type, lwd=line width
g6 = p2_ct +
  geom_line(lty=2, lwd=1)
print(g6)
```



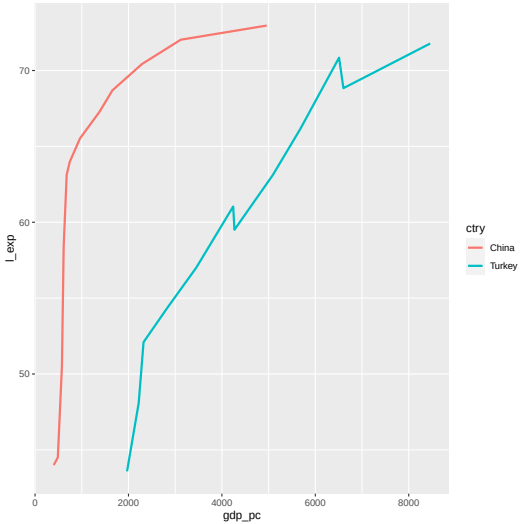
# ggplot2 X

```
#color is conditioned on country
g7=p2_ct +
  geom_line(
    aes(color=ctry),lty=2, lwd=1
    )
print(g7)
```



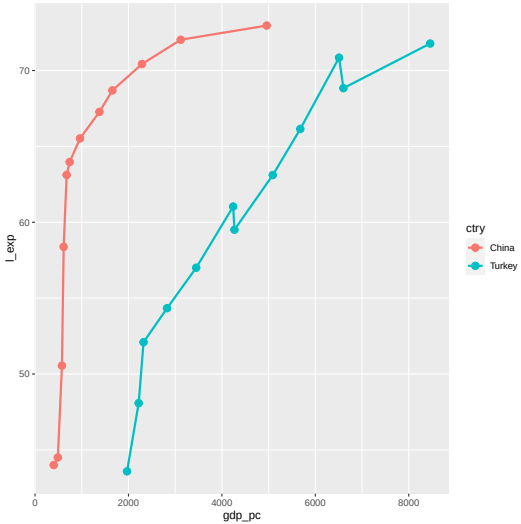
# ggplot2 XI

```
#color and group are condi-
tioned on country
p3_ct=ggplot(gm_ct,
        aes(x=gdp_pc, y=l_exp,
            group=ctry, color=ctry))
#lty=line type, lwd=line width
g8 = p3_ct +
  geom_line(lwd=1)
print(g8)
```



# ggplot2 XII

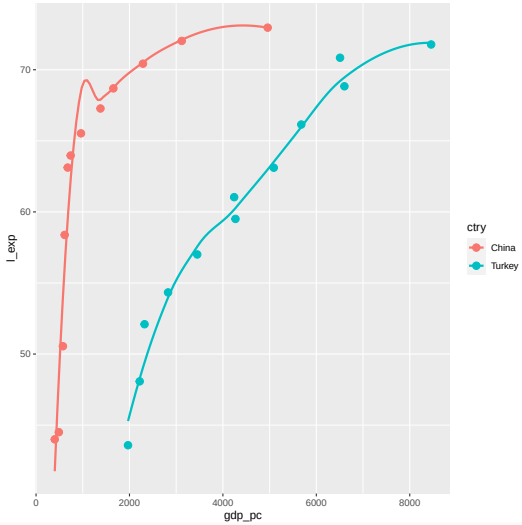
```
g12 = p3_ct +
  geom_line(lwd=1) + geom_point(size=3)
print(g12)
```



# ggplot2 XIII

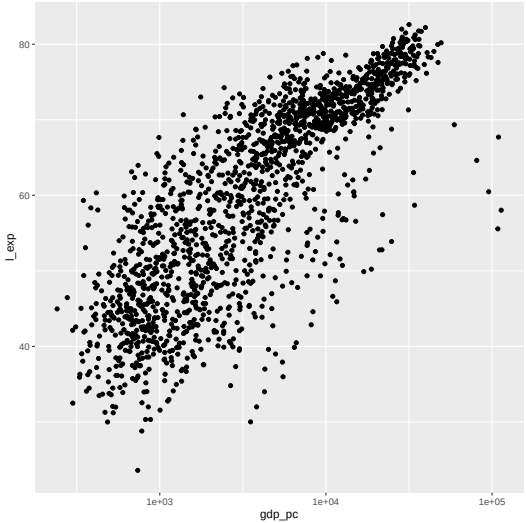
```
g13=p3_ct + geom_point(size=3) +
  geom_smooth(lwd = 1, se = FALSE)
print(g13)
```

```
## `geom_smooth()` using method = 'loess' and
formula = 'y ~ x'
```



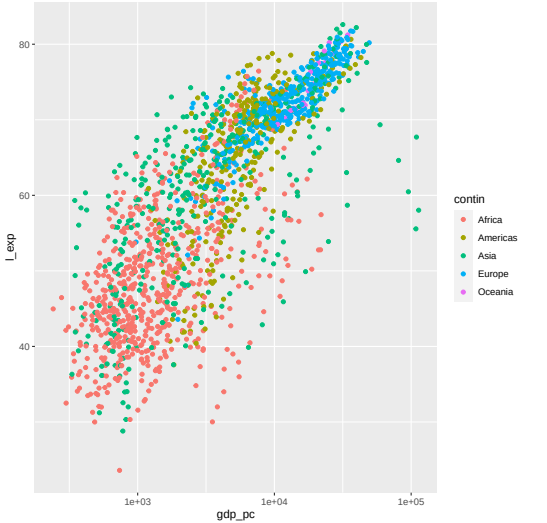
# ggplot2 XIV

```
## all countries
p1 = ggplot(gm,
            aes(x=gdp_pc, y=l_exp))
g14 = p1 +
  geom_point() + scale_x_log10()

print(g14)
```



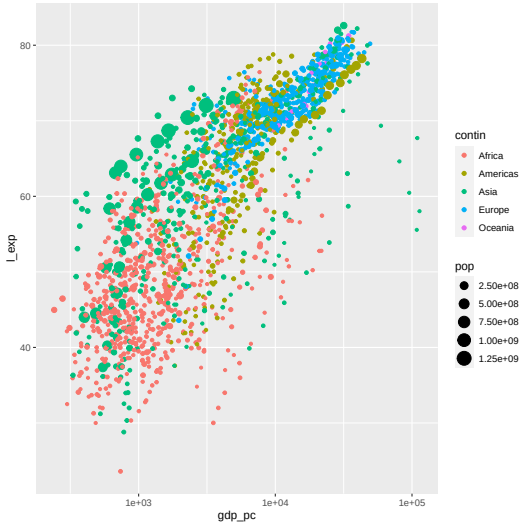
# ggplot2 XV

```
p3 = ggplot(gm,
 aes(x=gdp_pc, y=l_exp, color=contin))+
 scale_x_log10()

g15 = p3 + geom_point()
print(g15)
```



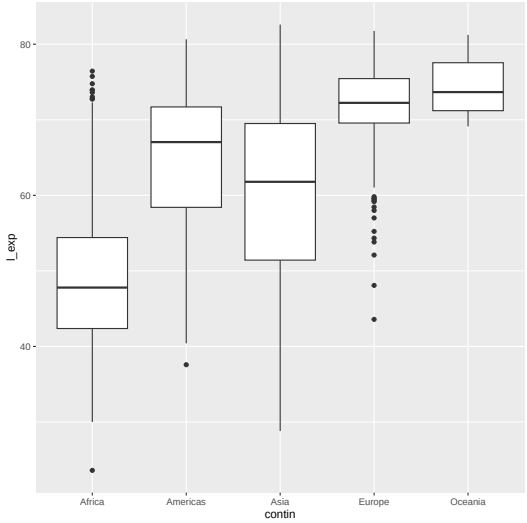
# ggplot2 XVI

```
g16 = p3 + geom_point(aes(size=pop))
print(g16)
```



# ggplot2 XVII

```
g17 = ggplot(gm,
  aes(x=contin, y=l_exp)) +
  geom_boxplot()

print(g17)
```



# ggplot2 XVIII

```
g18 = ggplot(gm,
  aes(x=contin, y=l_exp, fill=contin)) +
   geom_boxplot()
print(g18)
```