Example 2-7 & Figure 2-4

September 12, 2020

[ ]:
```
# first install the following packages and library
install.packages("pder")
install.packages("plm")
library("plm")

# create a vector of the model names
models <- c("within", "random", "pooling", "between")

# the following code allows us to plot our results later on
baw <- FALSE
library("ggplot2")
plotplm <- function(x, N = 10, seed = 1, lgth = 0.1){
    mydata <- model.frame(x)
    onames <- names(mydata)
    names(mydata) <- c("y", "x")
    LGTH <- (max(mydata$x) - min(mydata$x)) ^ 2 +
        (max(mydata$y) - min(mydata$y)) ^ 2
    lgth <- lgth * sqrt(LGTH) / 2
    seed <- set.seed(seed)
    theids <- sample(unique(index(mydata)[[1]]), N)
    small <- subset(mydata, index(mydata)[[1]] %in% theids)
    small <- cbind(small, id = index(small)[[1]])
    ymean <- with(small, tapply(y, id, mean)[as.character(theids)])
    xmean <- with(small, tapply(x, id, mean)[as.character(theids)])
    within <- update(x, model = "within")
    alpha <- mean(mydata[[1]]) - coef(within) * mean(mydata[[2]])
    beta <- as.numeric(coef(within))
    random <- update(within, model = "random")
    between <- update(within, model = "between")
    ols <- update(within, model = "pooling")
    FE <- fixef(within)[as.character(theids)]
    DATA <- data.frame(id = names(FE), FE = as.numeric(FE), slope = beta,
                       xmean = xmean, ymean = ymean,
                       xmin = xmean - lgth / sqrt(1 + beta ^ 2),
                       xmax = xmean + lgth / sqrt(1 + beta ^ 2),
                       ymin = ymean - lgth * beta / sqrt(1 + beta ^ 2),
                       ymax = ymean + lgth * beta / sqrt(1 + beta ^ 2))
    MODELS <- data.frame(models = c("ols", "random", "within", "between"),
                         intercept = c(coef(ols)[1], coef(random)[1], alpha,␣
↪coef(between)[1]),
                         slope = c(coef(ols)[2], coef(random)[2], coef(within),␣
↪coef(between)[2]))
    if (! baw){
        ggplot(data = small, aes(x = x, y = y, color = id)) + geom_point(size =␣
↪1) +
            geom_segment(aes(x = xmin, xend = xmax, y = ymin, yend = ymax,␣
↪color = id), data = DATA) +
            geom_abline(aes(intercept = intercept, slope = slope, lty =␣
↪models), data = MODELS) +
            geom_point(aes(x = xmean, y = ymean, color = id), size = 2, shape =␣
↪13, data = DATA) +
            xlab(onames[2]) + ylab(onames[1]) +
            theme(legend.text = element_text(size = 10),
                  legend.title= element_text(size = 12),
                  axis.title = element_text(size = 12))
    } else {
        ggplot(data = small, aes(x = x, y = y)) + geom_point(size = 1,␣
↪aes(shape = id)) +
            geom_segment(aes(x = xmin, xend = xmax, y = ymin, yend = ymax),␣
↪data = DATA) +
            geom_abline(aes(intercept = intercept, slope = slope, lty =␣
↪models), data = MODELS) +
            geom_point(aes(x = xmean, y = ymean, shape = id), size = 2,  data =␣
↪DATA) +
            scale_shape_manual(values=1:N) +
            xlab(onames[2]) + ylab(onames[1]) +
            theme(legend.text = element_text(size = 10),
                  legend.title= element_text(size = 12),
                  axis.title = element_text(size = 12))
    }
}
```

[4]:
```
##---------------------------------Block 1---------------------------------

#### Example 2-7 ####

## -----------------------------------------------------------------------
data("DemocracyIncome25", package = "pdder")

# create the data frame
DI <- pdata.frame(DemocracyIncome25)
summary(lag(DI$income))
```

total 135.016636559175 **between_id** 58.0267124124002 **between_time** 66.0351965070845

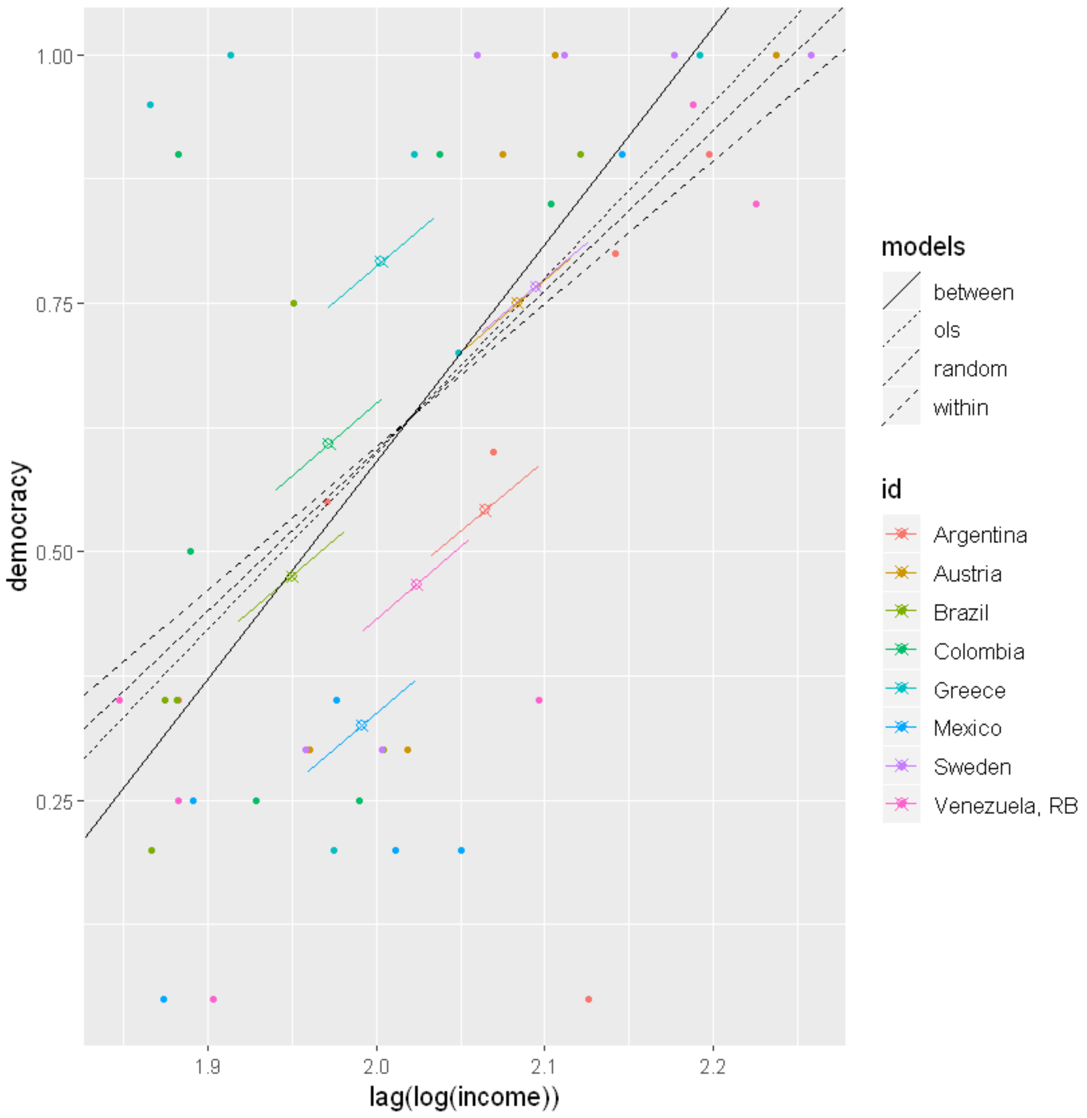
[5]:
```
##----------------------------------Block 2----------------------------------

# compute the variances of the error components
ercomp(democracy ~ lag(income), DI)
```

```
                  var std.dev share
idiosyncratic 0.05864 0.24215 0.791
individual    0.01545 0.12431 0.209
theta: 0.3776
```

[6]:
```
##----------------------------------Block 3----------------------------------

# extract the coefficients of the models
sapply(models, function(x)
       coef(plm(democracy ~ lag(income), DI, model = x))["lag(income)"])
```

within.lag(income)	0.186998893269288	**random.lag(income)**	0.210090198104884
pooling.lag(income)	0.230909458349127	**between.lag(income)**	0.289170058238228

[3]:
```
##----------------------------------Block 4----------------------------------

#### Figure 2.4 ####

plotplm(plm(democracy~lag(log(income)), DemocracyIncome25), N = 8)
```



democracy
1.00
0.75
0.50
0.25
lag(log(income))
1.9
2.0
2.1
2.2
models
between
ols
random
within
id
Argentina
Austria
Brazil
Colombia
Greece
Mexico
Sweden
Venezuela, RB