R Notebook

## Libraries

library(tidyverse)

## -- Attaching packages -------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.2.1 v purrr 0.3.3  
## v tibble 2.1.3 v dplyr 0.8.4  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## Warning: package 'readr' was built under R version 3.6.3

## Warning: package 'forcats' was built under R version 3.6.3

## -- Conflicts ----------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(XLConnect)

## Warning: package 'XLConnect' was built under R version 3.6.3

## Loading required package: XLConnectJars

## XLConnect 0.2-15 by Mirai Solutions GmbH [aut],  
## Martin Studer [cre],  
## The Apache Software Foundation [ctb, cph] (Apache POI),  
## Graph Builder [ctb, cph] (Curvesapi Java library)

## http://www.mirai-solutions.com  
## https://github.com/miraisolutions/xlconnect

library(openxlsx)

##   
## Attaching package: 'openxlsx'

## The following objects are masked from 'package:XLConnect':  
##   
## getTables, loadWorkbook, mergeCells, saveWorkbook

library(countrycode)

## Warning: package 'countrycode' was built under R version 3.6.3

library(purrr)  
library(dplyr)  
library(ggplot2)  
library(gganimate)

## Warning: package 'gganimate' was built under R version 3.6.3

theme\_set(theme\_bw())  
library(modelr)  
library(broom)

## Warning: package 'broom' was built under R version 3.6.3

##   
## Attaching package: 'broom'

## The following object is masked from 'package:modelr':  
##   
## bootstrap

## Part 1: Creating our own gapminder animation

All the important URLS

if(!file.exists("./data")) {dir.create("./data")}  
  
  
fileUrls <- c(  
 "https://docs.google.com/spreadsheet/pub?key=0AkBd6lyS3EmpdHo5S0J6ekhVOF9QaVhod05QSGV4T3c&output=xlsx",  
 "https://docs.google.com/spreadsheet/pub?key=phAwcNAVuyj2tPLxKvvnNPA&output=xlsx",  
 "https://docs.google.com/spreadsheet/pub?key=tSUr\_yZVbM6a3AGJEq\_Z2Pw&output=xlsx",  
 "https://docs.google.com/spreadsheet/pub?key=0ArfEDsV3bBwCdHBzUVVSMDlTX1ZCUnNJQ3ZFdkFXVFE&output=xlsx",  
 "https://docs.google.com/spreadsheet/pub?key=phAwcNAVuyj0XOoBL\_n5tAQ&output=xlsx")  
  
var\_names <- c("GDP","life\_expectancy", "alt\_GDP", "blood press", "population")

get\_clean function

round\_df <- function(x, digits) {  
 # round all numeric variables  
 # x: data frame   
 # digits: number of digits to round  
 numeric\_columns <- sapply(x, mode) == 'numeric'  
 x[numeric\_columns] <- round(x[numeric\_columns], digits)  
 x  
}  
  
get\_clean <- function(url\_in, var\_name){  
 tmp = tempfile(fileext = ".xlsx")  
 download.file(url\_in, destfile = tmp, mode="wb")  
 data<-readWorkbook(tmp)  
   
 data1<-data %>%   
 rename(country=colnames(data[1])) %>%   
 pivot\_longer(-country,names\_to = "year",values\_to = var\_name) %>%   
 na.omit()  
   
 data2<-round\_df(data1,0)  
 data2$year<-as.numeric(data2$year)  
 return(data2)  
}  
  
setwd("data/")  
out1 <- get\_clean(fileUrls[1],var\_names[1])  
head(out1)

## # A tibble: 6 x 3  
## country year GDP  
## <chr> <dbl> <dbl>  
## 1 Albania 1980 1061  
## 2 Albania 1981 1100  
## 3 Albania 1982 1111  
## 4 Albania 1983 1101  
## 5 Albania 1984 1065  
## 6 Albania 1985 1060

all\_data<-map2(fileUrls,var\_names,get\_clean)  
head(all\_data)

## [[1]]  
## # A tibble: 7,988 x 3  
## country year GDP  
## <chr> <dbl> <dbl>  
## 1 Albania 1980 1061  
## 2 Albania 1981 1100  
## 3 Albania 1982 1111  
## 4 Albania 1983 1101  
## 5 Albania 1984 1065  
## 6 Albania 1985 1060  
## 7 Albania 1986 1092  
## 8 Albania 1987 1054  
## 9 Albania 1988 1014  
## 10 Albania 1989 1092  
## # ... with 7,978 more rows  
##   
## [[2]]  
## # A tibble: 43,857 x 3  
## country year life\_expectancy  
## <chr> <dbl> <dbl>  
## 1 Afghanistan 1800 28  
## 2 Afghanistan 1801 28  
## 3 Afghanistan 1802 28  
## 4 Afghanistan 1803 28  
## 5 Afghanistan 1804 28  
## 6 Afghanistan 1805 28  
## 7 Afghanistan 1806 28  
## 8 Afghanistan 1807 28  
## 9 Afghanistan 1808 28  
## 10 Afghanistan 1809 28  
## # ... with 43,847 more rows  
##   
## [[3]]  
## # A tibble: 7,334 x 3  
## country year alt\_GDP  
## <chr> <dbl> <dbl>  
## 1 Afghanistan 1970 1731  
## 2 Afghanistan 1971 1748  
## 3 Afghanistan 1972 2120  
## 4 Afghanistan 1973 2119  
## 5 Afghanistan 1974 2148  
## 6 Afghanistan 1975 2263  
## 7 Afghanistan 1976 2270  
## 8 Afghanistan 1977 2121  
## 9 Afghanistan 1978 2205  
## 10 Afghanistan 1979 2121  
## # ... with 7,324 more rows  
##   
## [[4]]  
## # A tibble: 5,771 x 3  
## country year `blood press`  
## <chr> <dbl> <dbl>  
## 1 Afghanistan 1980 122  
## 2 Afghanistan 1981 122  
## 3 Afghanistan 1982 122  
## 4 Afghanistan 1983 123  
## 5 Afghanistan 1984 123  
## 6 Afghanistan 1985 123  
## 7 Afghanistan 1986 123  
## 8 Afghanistan 1987 123  
## 9 Afghanistan 1988 124  
## 10 Afghanistan 1989 124  
## # ... with 5,761 more rows  
##   
## [[5]]  
## # A tibble: 20,176 x 3  
## country year population  
## <chr> <dbl> <dbl>  
## 1 Afghanistan 1800 3280000  
## 2 Afghanistan 1810 3280000  
## 3 Afghanistan 1820 3323519  
## 4 Afghanistan 1830 3448982  
## 5 Afghanistan 1840 3625022  
## 6 Afghanistan 1850 3810047  
## 7 Afghanistan 1860 3973968  
## 8 Afghanistan 1870 4169690  
## 9 Afghanistan 1880 4419695  
## 10 Afghanistan 1890 4710171  
## # ... with 20,166 more rows

1. Join the outputs into one tibble with a column for each variable (hint – perhaps use one of your purrr functions)

dat1<-all\_data[[1]]  
dat2<-all\_data[[2]]  
dat3<-all\_data[[3]]  
dat4<-all\_data[[4]]  
dat5<-all\_data[[5]]  
  
dat6<-full\_join(dat2,dat1)

## Joining, by = c("country", "year")

dat7<-full\_join(dat3,dat4)

## Joining, by = c("country", "year")

dat8<-full\_join(dat6,dat7)

## Joining, by = c("country", "year")

join\_data<-full\_join(dat8,dat5)

## Joining, by = c("country", "year")

join\_data

## # A tibble: 48,270 x 7  
## country year life\_expectancy GDP alt\_GDP `blood press` population  
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Afghanistan 1800 28 NA NA NA 3280000  
## 2 Afghanistan 1801 28 NA NA NA NA  
## 3 Afghanistan 1802 28 NA NA NA NA  
## 4 Afghanistan 1803 28 NA NA NA NA  
## 5 Afghanistan 1804 28 NA NA NA NA  
## 6 Afghanistan 1805 28 NA NA NA NA  
## 7 Afghanistan 1806 28 NA NA NA NA  
## 8 Afghanistan 1807 28 NA NA NA NA  
## 9 Afghanistan 1808 28 NA NA NA NA  
## 10 Afghanistan 1809 28 NA NA NA NA  
## # ... with 48,260 more rows

new\_gapminder<-join\_data %>%   
 mutate(continent=countrycode(sourcevar = country,  
 origin = "country.name",  
 destination = "continent"))

## Warning in countrycode(sourcevar = country, origin = "country.name", destination = "continent"): Some values were not matched unambiguously: Akrotiri and Dhekelia, Central African Rep., Channel Islands, Cocos Island, Czechoslovakia, East Germany, Eritrea and Ethiopia, Kosovo, North Yemen (former), Serbia and Montenegro, South Yemen (former), St. Martin, Yugoslavia

## Warning in countrycode(sourcevar = country, origin = "country.name", destination = "continent"): Some strings were matched more than once, and therefore set to <NA> in the result: Eritrea and Ethiopia,Africa,Africa

custom\_match<-c("Akrotiri and Dhekelia"="Europe",  
 "Central African Rep."="Africa",  
 "Channel Islands"="Europe",  
 "Cocos Island"="Asia",  
 "Czechoslovakia"="Europe",  
 "East Germany"="Europe",  
 "Eritrea and Ethiopia"="Africa",  
 "Kosovo"="Europe",  
 "North Yemen (former)"="Asia",  
 "Serbia and Montenegro"="Asia",  
 "St. Martin"="North America",  
 "Yugoslavia"="Europe",  
 "South Yemen (former)"="Asia")  
  
#new\_gapminder$continent<-countrycode(sourcevar=new\_gapminder$country,  
# origin=custom\_match,  
# destination="continent")

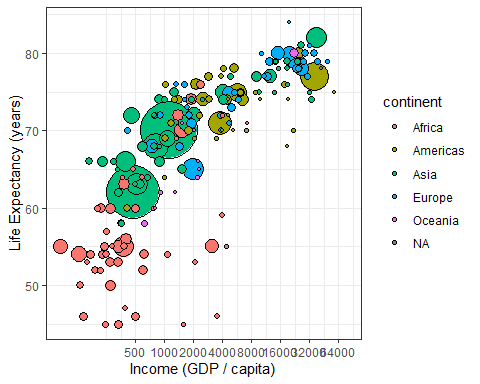
new\_gapminder %>%  
 arrange(country, year)

## # A tibble: 48,270 x 8  
## country year life\_expectancy GDP alt\_GDP `blood press` population  
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Afghan~ 1800 28 NA NA NA 3280000  
## 2 Afghan~ 1801 28 NA NA NA NA  
## 3 Afghan~ 1802 28 NA NA NA NA  
## 4 Afghan~ 1803 28 NA NA NA NA  
## 5 Afghan~ 1804 28 NA NA NA NA  
## 6 Afghan~ 1805 28 NA NA NA NA  
## 7 Afghan~ 1806 28 NA NA NA NA  
## 8 Afghan~ 1807 28 NA NA NA NA  
## 9 Afghan~ 1808 28 NA NA NA NA  
## 10 Afghan~ 1809 28 NA NA NA NA  
## # ... with 48,260 more rows, and 1 more variable: continent <chr>

## Just for my practice

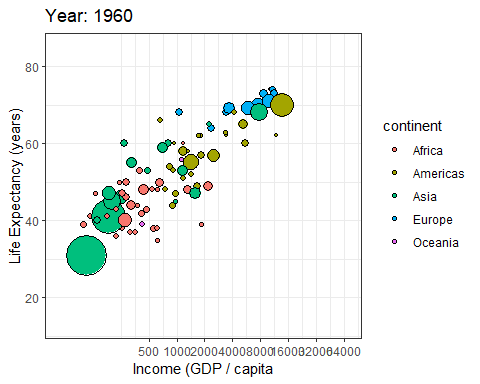
p<-new\_gapminder %>%   
 filter(year=="2002") %>%   
 arrange(desc(population)) %>%   
 ggplot(aes(x=GDP,y=life\_expectancy))+  
 geom\_point(aes(size=population,fill=continent),shape=21)+  
 scale\_x\_log10(breaks = 2^(-1:7)\*1000)+  
 scale\_size(range = c(1,20),guide=F)+  
 labs(  
 x="Income (GDP / capita)",  
 y="Life Expectancy (years)"  
 )+  
 theme\_bw()  
p

## Warning: Removed 89 rows containing missing values (geom\_point).



## animation

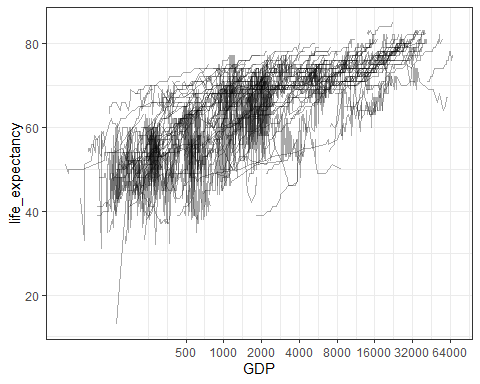
plot\_data<-new\_gapminder %>%   
 select(country,continent,year,GDP,life\_expectancy,population) %>%   
 drop\_na()  
  
  
p <- ggplot(  
 plot\_data,   
 aes(x = GDP, y=life\_expectancy)) +  
 geom\_point(aes(size = population, fill = continent),shape=21)+  
 labs(  
 x="Income (GDP / capita",  
 y="Life Expectancy (years)"  
 )+  
 theme\_bw()+  
 scale\_x\_log10(breaks = 2^(-1:7)\*1000)+  
 scale\_size(range = c(1,20),guide=F)  
  
p + transition\_time(year) +  
 labs(title = "Year: {frame\_time}")



## Part 2: Analyze life expectancy as function of GDP

1. Now, model life expectancy for each country as a function of GDP? (hint – use plot\_data from your plot above)

plot\_data %>%  
 ggplot(aes(GDP, life\_expectancy, group = country)) +  
 geom\_line(alpha = 1/3)+  
 scale\_x\_log10(breaks = 2^(-1:7)\*1000)



by\_country <- plot\_data %>%  
 group\_by(country, continent) %>%  
 nest()  
  
by\_country

## # A tibble: 190 x 3  
## # Groups: country, continent [190]  
## country continent data   
## <chr> <chr> <list>   
## 1 Albania Europe <tibble [32 x 4]>  
## 2 Algeria Africa <tibble [52 x 4]>  
## 3 Andorra Europe <tibble [19 x 4]>  
## 4 Angola Africa <tibble [27 x 4]>  
## 5 Antigua and Barbuda Americas <tibble [35 x 4]>  
## 6 Argentina Americas <tibble [52 x 4]>  
## 7 Armenia Asia <tibble [22 x 4]>  
## 8 Aruba Americas <tibble [17 x 4]>  
## 9 Australia Oceania <tibble [52 x 4]>  
## 10 Austria Europe <tibble [52 x 4]>  
## # ... with 180 more rows

country\_mod <- function(df){  
 lm(life\_expectancy ~ GDP, data = df)  
}  
  
by\_country <- by\_country %>%  
 mutate(model = map(data, country\_mod))  
by\_country

## # A tibble: 190 x 4  
## # Groups: country, continent [190]  
## country continent data model   
## <chr> <chr> <list> <list>  
## 1 Albania Europe <tibble [32 x 4]> <lm>   
## 2 Algeria Africa <tibble [52 x 4]> <lm>   
## 3 Andorra Europe <tibble [19 x 4]> <lm>   
## 4 Angola Africa <tibble [27 x 4]> <lm>   
## 5 Antigua and Barbuda Americas <tibble [35 x 4]> <lm>   
## 6 Argentina Americas <tibble [52 x 4]> <lm>   
## 7 Armenia Asia <tibble [22 x 4]> <lm>   
## 8 Aruba Americas <tibble [17 x 4]> <lm>   
## 9 Australia Oceania <tibble [52 x 4]> <lm>   
## 10 Austria Europe <tibble [52 x 4]> <lm>   
## # ... with 180 more rows

1. Graph your residuals by country and facet by continent.

by\_country <- by\_country %>%  
 mutate(resids = map2(data, model, add\_residuals))

## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading

by\_country

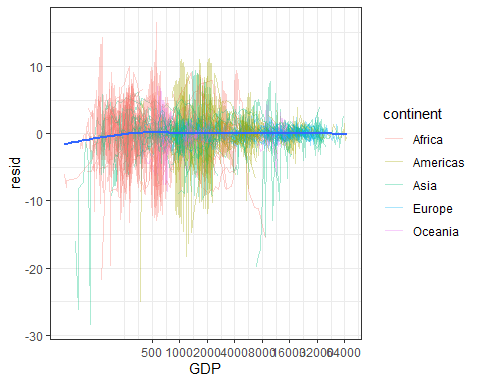
## # A tibble: 190 x 5  
## # Groups: country, continent [190]  
## country continent data model resids   
## <chr> <chr> <list> <list> <list>   
## 1 Albania Europe <tibble [32 x 4]> <lm> <tibble [32 x 5]>  
## 2 Algeria Africa <tibble [52 x 4]> <lm> <tibble [52 x 5]>  
## 3 Andorra Europe <tibble [19 x 4]> <lm> <tibble [19 x 5]>  
## 4 Angola Africa <tibble [27 x 4]> <lm> <tibble [27 x 5]>  
## 5 Antigua and Barbuda Americas <tibble [35 x 4]> <lm> <tibble [35 x 5]>  
## 6 Argentina Americas <tibble [52 x 4]> <lm> <tibble [52 x 5]>  
## 7 Armenia Asia <tibble [22 x 4]> <lm> <tibble [22 x 5]>  
## 8 Aruba Americas <tibble [17 x 4]> <lm> <tibble [17 x 5]>  
## 9 Australia Oceania <tibble [52 x 4]> <lm> <tibble [52 x 5]>  
## 10 Austria Europe <tibble [52 x 4]> <lm> <tibble [52 x 5]>  
## # ... with 180 more rows

resids <- unnest(by\_country, resids)  
resids

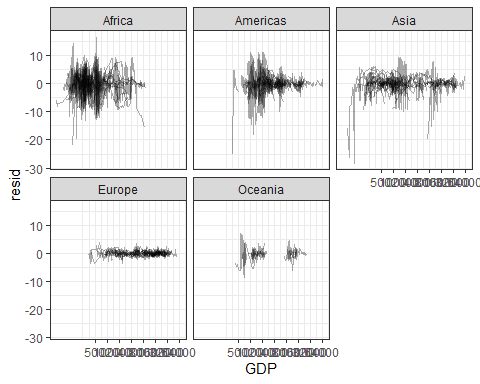
## # A tibble: 7,672 x 9  
## # Groups: country, continent [190]  
## country continent data model year GDP life\_expectancy population resid  
## <chr> <chr> <list> <lis> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 Albania Europe <tibbl~ <lm> 1980 1061 71 2681245 -2.75   
## 2 Albania Europe <tibbl~ <lm> 1981 1100 72 2735329 -1.90   
## 3 Albania Europe <tibbl~ <lm> 1982 1111 72 2788315 -1.94   
## 4 Albania Europe <tibbl~ <lm> 1983 1101 72 2842620 -1.90   
## 5 Albania Europe <tibbl~ <lm> 1984 1065 72 2901590 -1.77   
## 6 Albania Europe <tibbl~ <lm> 1985 1060 73 2966799 -0.747  
## 7 Albania Europe <tibbl~ <lm> 1986 1092 73 3041003 -0.868  
## 8 Albania Europe <tibbl~ <lm> 1987 1054 73 3121336 -0.724  
## 9 Albania Europe <tibbl~ <lm> 1988 1014 73 3197064 -0.574  
## 10 Albania Europe <tibbl~ <lm> 1989 1092 73 3253659 -0.868  
## # ... with 7,662 more rows

resids %>%  
 ggplot(aes(GDP,resid)) +  
 geom\_line(aes(group = country, color = continent), alpha = 1/3) +  
 scale\_x\_log10(breaks = 2^(-1:7)\*1000)+  
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'



resids %>%  
 ggplot(aes(GDP, resid, group = country)) +  
 geom\_line(alpha = 1/3) +  
 scale\_x\_log10(breaks = 2^(-1:7)\*1000)+  
 facet\_wrap(~continent)



1. Are there countries and continents for which this is a particularly bad model? Use broom::glance to make this determination and provide an explanation for your conclusions.

glance <- by\_country %>%  
 mutate(glance = map(model, glance)) %>%  
 unnest(glance, .drop = TRUE)

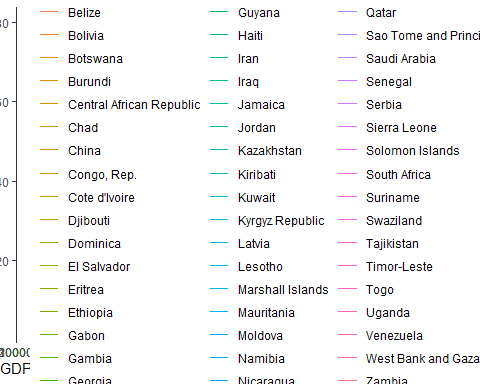
## Warning: The `.drop` argument of `unnest()` is deprecated as of tidyr 1.0.0.  
## All list-columns are now preserved.  
## This warning is displayed once per session.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

glance %>%  
 arrange(r.squared)

## # A tibble: 190 x 16  
## # Groups: country, continent [190]  
## country continent data model resids r.squared adj.r.squared sigma  
## <chr> <chr> <lis> <lis> <list> <dbl> <dbl> <dbl>  
## 1 Sao To~ Africa <tib~ <lm> <tibb~ 0. 0 NaN   
## 2 Haiti Americas <tib~ <lm> <tibb~ 4.49e-7 -0.0526 6.40   
## 3 Eritrea Africa <tib~ <lm> <tibb~ 8.44e-5 -0.0555 5.23   
## 4 Burundi Africa <tib~ <lm> <tibb~ 4.27e-4 -0.0196 5.34   
## 5 Sierra~ Africa <tib~ <lm> <tibb~ 6.30e-4 -0.0194 5.65   
## 6 West B~ Asia <tib~ <lm> <tibb~ 1.07e-3 -0.0988 0.605  
## 7 Iraq Asia <tib~ <lm> <tibb~ 1.13e-3 -0.0757 0.865  
## 8 Jamaica Americas <tib~ <lm> <tibb~ 1.32e-3 -0.0280 1.85   
## 9 Cote d~ Africa <tib~ <lm> <tibb~ 2.45e-3 -0.0175 5.31   
## 10 Tajiki~ Asia <tib~ <lm> <tibb~ 4.11e-3 -0.0357 2.19   
## # ... with 180 more rows, and 8 more variables: statistic <dbl>, p.value <dbl>,  
## # df <int>, logLik <dbl>, AIC <dbl>, BIC <dbl>, deviance <dbl>,  
## # df.residual <int>

bad\_fit <- filter(glance, r.squared < 0.5)  
new\_gapminder %>%  
 semi\_join(bad\_fit, by = "country") %>%  
 ggplot(aes(GDP, life\_expectancy, colour = country)) +  
 geom\_line()

## Warning: Removed 10419 rows containing missing values (geom\_path).

 # these are bad fits as there rsquare value is less than 0.5 which means there is less corelation between the life Expectency and GDP.

## Part 3:

1. Transform year so that it has a mean of 0

transform\_data<-plot\_data  
  
transform\_data$scale\_year<-scale(transform\_data$year)  
mean(transform\_data$scale\_year)

## [1] 7.247346e-15

##close to zero

1. Model with a quadratic polynomial. How can you interpret the coefficients of the quadratic for Belgium?

country\_mod1 <- function(df){  
 lm(life\_expectancy ~ year+scale\_year+population, data = df)  
}  
  
by\_country <- transform\_data %>%  
 group\_by(country, continent) %>%  
 nest()  
  
by\_country <- by\_country %>%  
 mutate(model = map(data, country\_mod1))  
  
Belgium<-by\_country %>%   
 filter(country=="Belgium")  
  
Belgium$model

## [[1]]  
##   
## Call:  
## lm(formula = life\_expectancy ~ year + scale\_year + population,   
## data = df)  
##   
## Coefficients:  
## (Intercept) year scale\_year population   
## -3.911e+02 2.387e-01 NA -7.954e-07

the coeffxicients here tells yes about how each variable effect the life\_expectancy and from what we can see here, scale\_year does not effect the life\_expectancy and other have less relation as well.

1. Use glance() to identify all countries that do not fit the model well and plot the residuals for the countries that do not fit the model well.

by\_country <- by\_country %>%  
 mutate(resids = map2(data, model, add\_residuals))

## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading  
  
## Warning in predict.lm(model, data): prediction from a rank-deficient fit may be  
## misleading

by\_country

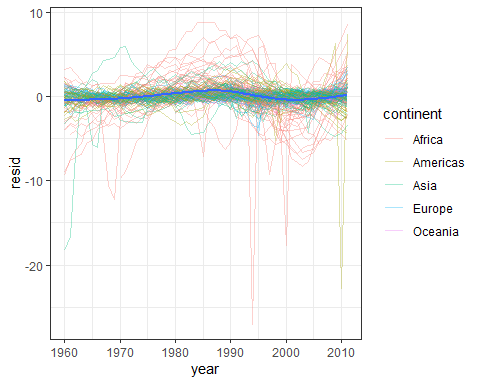
## # A tibble: 190 x 5  
## # Groups: country, continent [190]  
## country continent data model resids   
## <chr> <chr> <list> <list> <list>   
## 1 Albania Europe <tibble [32 x 5]> <lm> <tibble [32 x 6]>  
## 2 Algeria Africa <tibble [52 x 5]> <lm> <tibble [52 x 6]>  
## 3 Andorra Europe <tibble [19 x 5]> <lm> <tibble [19 x 6]>  
## 4 Angola Africa <tibble [27 x 5]> <lm> <tibble [27 x 6]>  
## 5 Antigua and Barbuda Americas <tibble [35 x 5]> <lm> <tibble [35 x 6]>  
## 6 Argentina Americas <tibble [52 x 5]> <lm> <tibble [52 x 6]>  
## 7 Armenia Asia <tibble [22 x 5]> <lm> <tibble [22 x 6]>  
## 8 Aruba Americas <tibble [17 x 5]> <lm> <tibble [17 x 6]>  
## 9 Australia Oceania <tibble [52 x 5]> <lm> <tibble [52 x 6]>  
## 10 Austria Europe <tibble [52 x 5]> <lm> <tibble [52 x 6]>  
## # ... with 180 more rows

resids <- unnest(by\_country, resids)  
resids

## # A tibble: 7,672 x 10  
## # Groups: country, continent [190]  
## country continent data model year GDP life\_expectancy population  
## <chr> <chr> <lis> <lis> <dbl> <dbl> <dbl> <dbl>  
## 1 Albania Europe <tib~ <lm> 1980 1061 71 2681245  
## 2 Albania Europe <tib~ <lm> 1981 1100 72 2735329  
## 3 Albania Europe <tib~ <lm> 1982 1111 72 2788315  
## 4 Albania Europe <tib~ <lm> 1983 1101 72 2842620  
## 5 Albania Europe <tib~ <lm> 1984 1065 72 2901590  
## 6 Albania Europe <tib~ <lm> 1985 1060 73 2966799  
## 7 Albania Europe <tib~ <lm> 1986 1092 73 3041003  
## 8 Albania Europe <tib~ <lm> 1987 1054 73 3121336  
## 9 Albania Europe <tib~ <lm> 1988 1014 73 3197064  
## 10 Albania Europe <tib~ <lm> 1989 1092 73 3253659  
## # ... with 7,662 more rows, and 2 more variables: scale\_year[,1] <dbl>,  
## # resid <dbl>

resids %>%  
 ggplot(aes(year,resid)) +  
 geom\_line(aes(group = country, color = continent), alpha = 1/3) +  
 geom\_smooth(se = FALSE)

## `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'



#glance1 <- by\_country %>%  
# mutate(glance = map(model, glance)) %>%  
# unnest(glance, .drop = TRUE)