R Demo

Daniel J Carter

This notebook contains a short demo for the BeginRs group. Whitespace in this notebook surrounds "chunks" of code. This notebook runs code to read in a dataset, produce some summary statistics on that dataset, and then do some visualisation. Run code by pressing Ctrl-Enter with your cursor on the line you wish to run. Ctrl-Shift-Enter runs a whole chunk.

Note you may wish to insert your own code chunks - Ctrl-Alt-I is the default but on some computers this produces an I with an accent on it, so you may wish to go to Tools -> Modify Keyboard Shortcuts to adjust this (I used Ctrl-Shift-I). You can also select a theme from Global Options -> Appearance.

```
## Loading required package: foreign
```

```
## Loading required package: survival
```

```
## Loading required package: MASS
```

```
## Loading required package: nnet
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:epiDisplay':
##
##     alpha
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  1.4.2     v purrr   0.2.5
## v tidyr   0.7.2     v dplyr   0.7.6
## v readr   1.1.1     v stringr 1.3.1
## v tibble  1.4.2     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------- tidyverse_conflicts() --
## x ggplot2::alpha() masks epiDisplay::alpha()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x dplyr::select()  masks MASS::select()
```

We read in an ecological dataset containing some data relevant to TB & the SDG goals, pulled from the SDG data repository and do some basic operations to it. First, we read it in. Reading in the file stores it as a dataframe. A dataframe is the same as a dataset in Stata - it's like a matrix with multiple different types of objects (e.g. string, numeric, factor). We examine the rows and also some of the variables. Note that we must specify which dataframe we want to access, then access the column using the $ - this is to avoid confusion with multiple datasets.

We then do a more complex operation using functions from the tidyverse. The %>% operator, the %>%, takes the result of what's on the left, and puts it into the leftmost argument of what's on the right. So it's like writing sentences. To get mean TB incidence by LMIC descriptor, we take the complete dataset and then we filter() it. How do we filter it? By extracting only those countries with no missing data on TB incidence (complete cases). Once we've done that, we want to tell R to group the data into groups defined by LMIC descriptor using group_by(). Now that we've grouped, we can summarise() the data by extracting the mean TB incidence. We could include any function into summarise, even a user defined one - an example of how flexible R could be.

```r
#---Get & set working directory
getwd()
```

```
## [1] "E:/LSHTM/Teaching/beginRs"
```

```r
#--- Read in a csv file
eleven <- read.csv("./elevendemo.csv")

#--- Examine the first few rows
head(eleven)
```

```
##       country code reg   gdp gini       pop delta.pop int.migrant  urb
## 1 Afghanistan  AFG EMR   562   NA  34656032     33.84        1.18 27.1
## 2      Angola  AGO AFR  3111   NA  28813463     42.20        0.43 44.8
## 3   Argentina  ARG AMR 12449 42.7  43847430     10.84        4.81 91.9
## 4     Armenia  ARM EUR  3606 32.4   2924816     -1.14        6.34 62.6
## 5  Bangladesh  BGD SEA  1359   NA 163000000     12.10        0.88 35.0
## 6      Belize  BLZ AMR  4811   NA    366954     26.21       14.99 43.9
##   delta.urb emp.ratio slums pop.density largest.city sanitation water
## 1      3.90      48.0  62.7        53.1         51.5       45.1  78.2
## 2      7.88      63.8  55.5        23.1         44.4       88.6  75.4
## 3      1.63      57.0  16.7        16.0         38.1       96.2  99.0
## 4     -1.59      52.9  14.4       102.7         56.9       96.2 100.0
## 5      7.52      59.7  55.1      1251.8         31.9       57.7  86.5
## 6     -2.20      62.2  10.8        16.1           NA       93.5  98.9
##   million  tb urb.pov electric pollution urban.pov.hc primary health.exp
## 1    14.0 189    27.6     98.7      48.0         27.6      NA       63.9
## 2    24.6 370      NA     51.0      36.4           NA    84.0       24.0
## 3    43.9  25    13.6       NA      13.4          4.7    99.3       30.7
## 4    35.6  41      NA    100.0      25.5         30.0    96.1       53.5
## 5    14.7 225      NA     90.7      89.4         21.3    90.5       67.0
## 6      NA  25     4.7    100.0      27.0           NA    96.1       23.0
##   tb.cure case.d diarrhea.trt imm.dpt mat.mort nurse.mw beds  ari
## 1      87     58         40.7      65   1291.0    0.360  0.5 61.5
## 2      34     64           NA      64       NA       NA   NA   NA
## 3      52     87         59.1      92     32.4       NA  4.7 94.3
## 4      78     89           NA      94     19.0    4.994  3.9 56.8
## 5      93     57         66.1      97    210.0    0.213  0.6 42.0
## 6      35     87         42.5      95     45.0    1.959  1.1 82.2
##                  lmic
## 1          Low income
## 2 Lower middle income
## 3 Upper middle income
## 4 Lower middle income
## 5 Lower middle income
## 6 Upper middle income
```

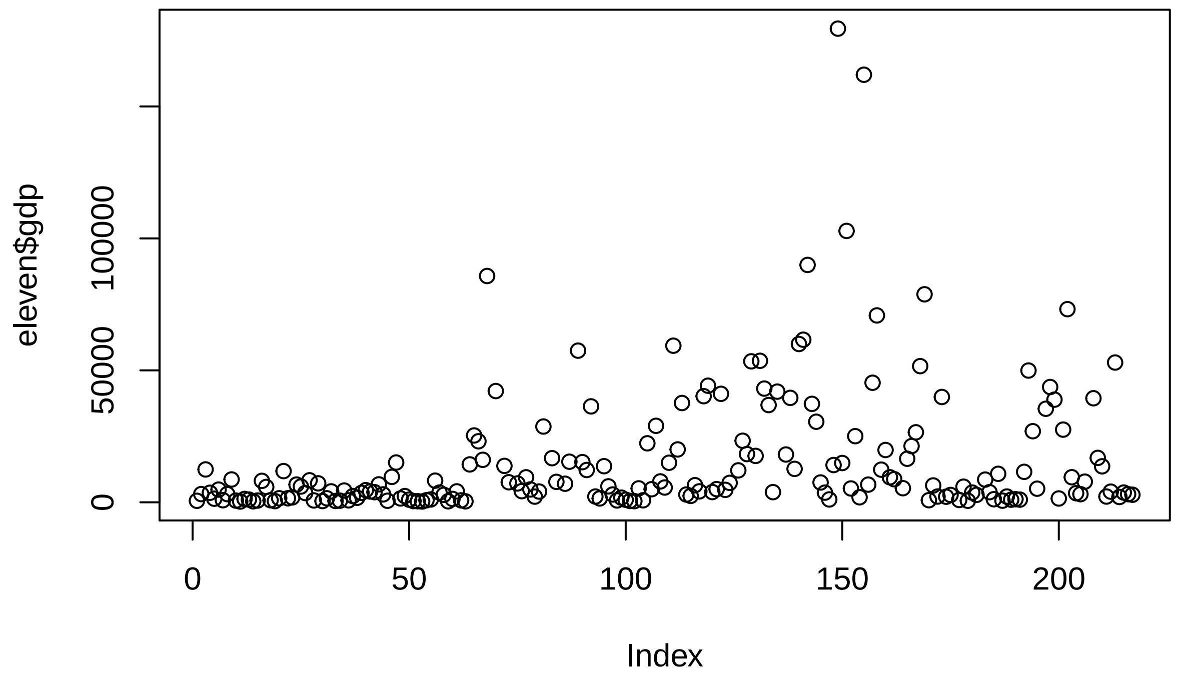
```r
#--- Examine the LMIC variable (categorical; known as 'factor' in R)
summary(eleven$lmic)
```

```
##         High income          Low income Lower middle income
##                  77                  31                  53
## Upper middle income
##                  56
```

```
#--- Examine the GDP variable
summary(eleven$gdp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##     286    1980    5480   15600   17000  179000      13
```

```
plot(eleven$gdp)
```



```
#--- Get mean TB Incidence by LMIC
eleven %>% filter(complete.cases(tb)) %>% group_by(lmic) %>% summarise(mean(tb))
```

```
## # A tibble: 4 x 2
##   lmic                `mean(tb)`
##   <fct>                    <dbl>
## 1 High income               18.9
## 2 Low income               206.
## 3 Lower middle income      202.
## 4 Upper middle income       88.7
```

In our next chunk, we introduce a new data frame based on removing the high income countries. This time we use the assignment operator <- to tell R to name the result that we get from our analysis and store it as a new object (it should appear in the environment window). We perform some tidying operations on it, excluding countries with missing data on slums and (arbitrarily!) remove the urban poverty variable.

```
#--- Filter out high income countries and store the result
lmic <- eleven %>% filter(lmic != "High income")

#--- Exclude countries with missing data on slums and then remove the urban poverty variable
```

```r
lmic <- lmic %>% filter(!is.na(slums)) %>% dplyr::select(-urban.pov.hc)

#--- Get help!
?filter
```

```
## starting httpd help server ... done
```

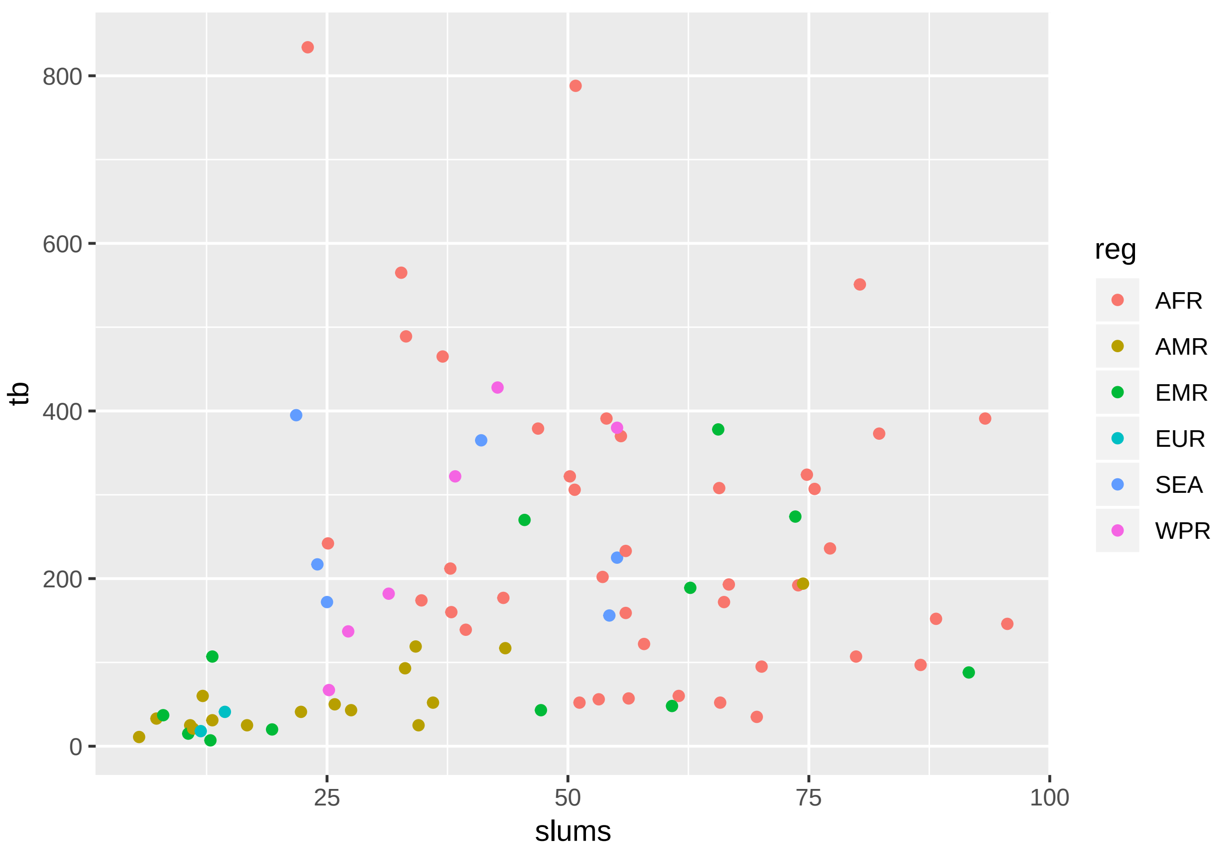
The code below makes a rather pretty plot. It looks complex but is fairly straightforward if you break it down into parts. We pipe in the dataset to the ggplot() function. One argument of the core ggplot function is aes() for aesthetics, and within aes() I specify the x and y axis. Alone, that won't do anything as ggplot() doesn't know what type of plot you want. So we use geom_point() to tell it we want a scatter plot, and the color should correspond to the region.

We can also add a size parameter to geom_point. We can use geom_smooth to add a linear regression by specifying method = "lm" - the optional arguments add a 95% CI, change the transparency of that CI, and the color of the line.

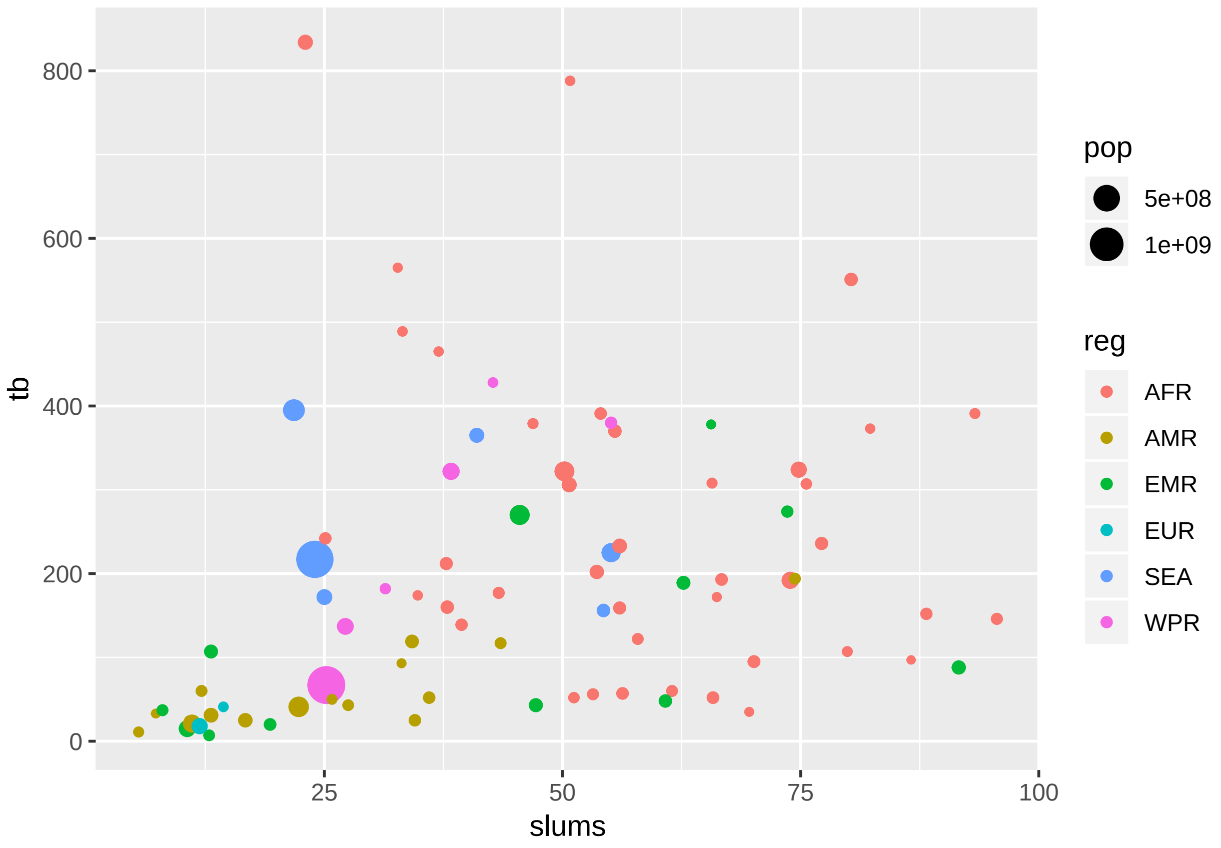
Part of the customisability of R comes from user created packages - ggrepel, for example, extends ggplot to have text as points. You can check the arguments for ggrepel by using the ? functionality, or directly search for its "vignette" which will provide a number of examples.

We also add labels to the plot for a more professional finish. Note that this could have all been done in one command. Note in fact that because of the way pipes work, we could have done produced this plot without storing any intermediate steps, starting from the dataset itself. See the commented code at the bottom of the chunk (note you can uncomment by highlighting and pressing Ctrl-Shift-C)

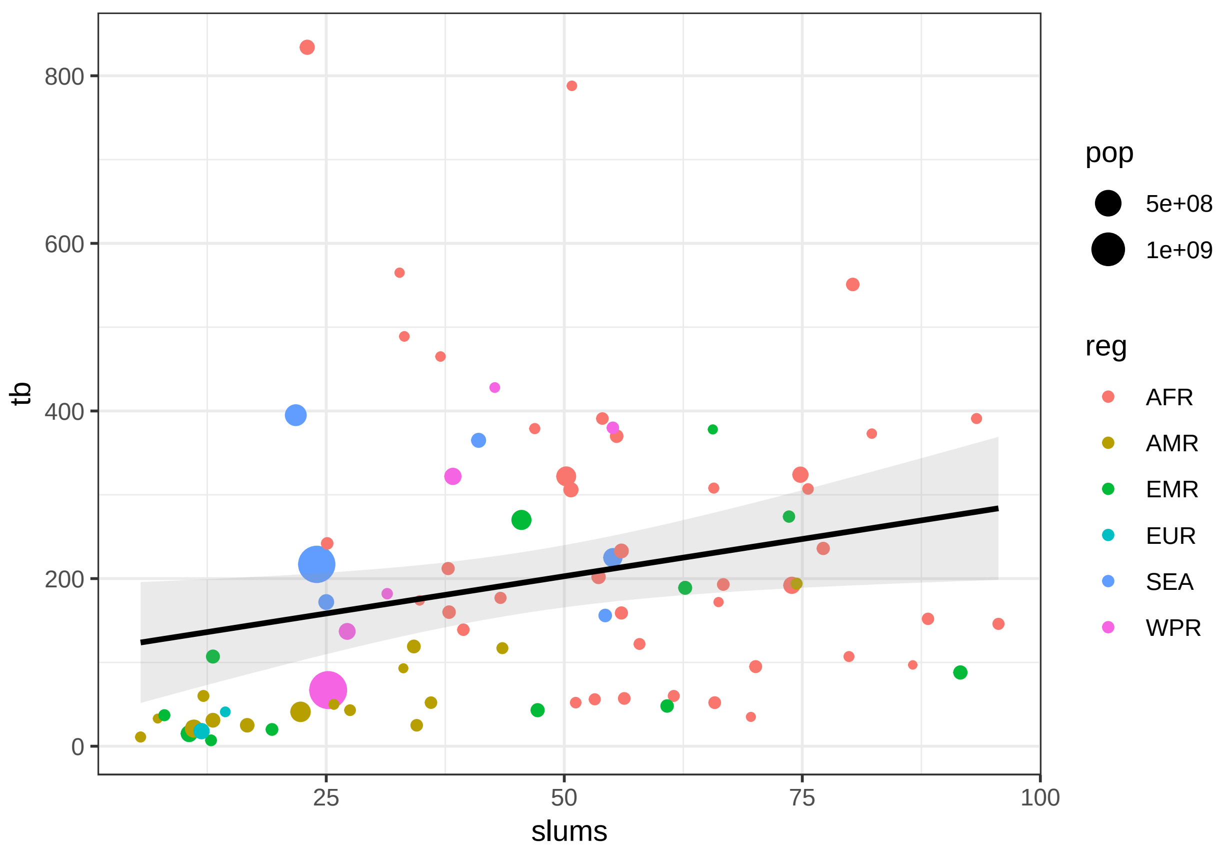
```r
#--- Make a scatter plot
slumplot <- lmic %>% ggplot(aes(x = slums, y = tb)) +
  geom_point(aes(color = reg))
slumplot
```



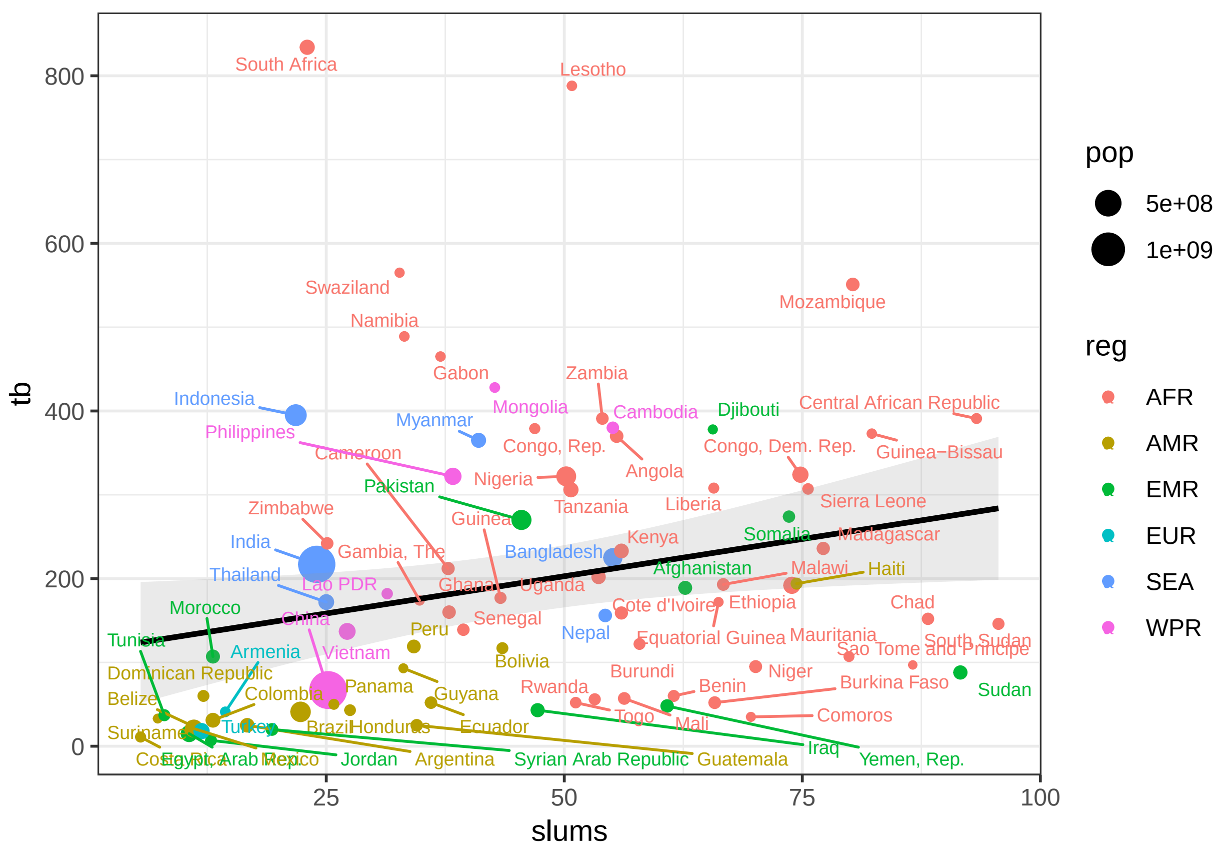
```r
#--- Turn it into a bubble plot
slumplot <- lmic %>% ggplot(aes(x = slums, y = tb)) +
  geom_point(aes(color = reg, size = pop))
slumplot
```



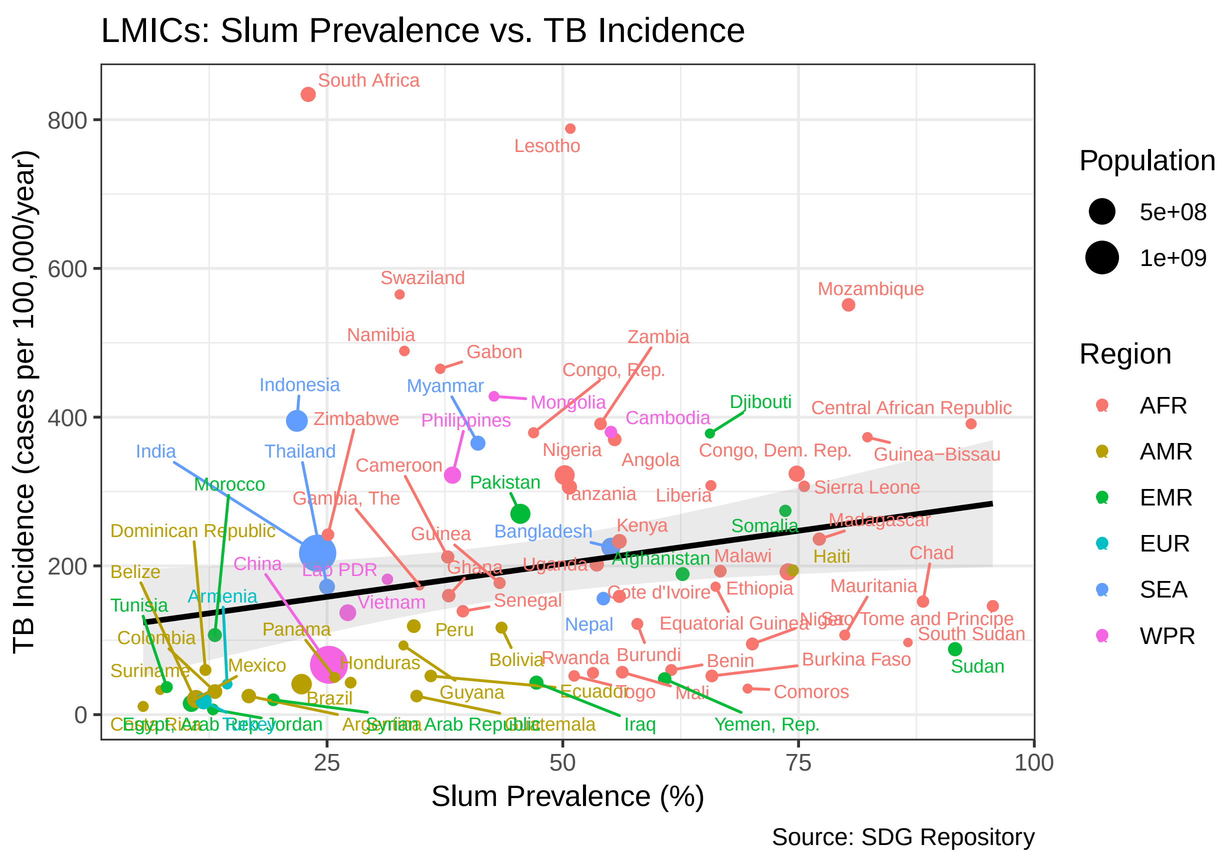
```
#--- Add a regression line and change the theme to black & white
slumplot <- slumplot + geom_smooth(method = "lm", se = T, alpha = 0.2, color = "black") +
  theme_bw()
slumplot
```



```
#--- Add names of countries using geom_text_repel (from a package we loaded)
slumplot <- slumplot + geom_text_repel(aes(color = reg, label = country), size = 2.5)
slumplot
```



```r
#--- Add labels
slumplot <- slumplot + labs(title = "LMICs: Slum Prevalence vs. TB Incidence",
       y = "TB Incidence (cases per 100,000/year)",
       x = "Slum Prevalence (%)",
       caption = "Source: SDG Repository",
       color = "Region",
       size = "Population")
slumplot
```



```r
ggsave("rdemoslumplot.pdf")
```

```
## Saving 6.5 x 4.5 in image
```

```r
#--- All together now...
# eleven %>%
#   filter(lmic != "High income") %>%
#   filter(!is.na(slums)) %>%
#   dplyr::select(-urban.pov.hc) %>%
#   ggplot(aes(x = slums, y = tb)) +
#   geom_point(aes(color = reg, size = pop)) +
#   geom_smooth(method = "lm", se = T, alpha = 0.2, color = "black") +
#   theme_bw() +
#   labs(title = "LMICs: Slum Prevalence vs. TB Incidence",
#        y = "TB Incidence (cases per 100,000/year)",
#        x = "Slum Prevalence (%)",
#        caption = "Source: SDG Repository",
#        color = "Region",
#        size = "Population")
```

Now let's do a quick logistic regression and extract the results. The glm() function (unsurprisingly) fits generalised linear models in R. First we create (artificial!) binary outcome and exposure variables using mutate() in the tidyverse. The code again looks complex but can be broken down quite simply: first we run an ifelse() statement that says if TB is higher than the 75th percentile, replace with 1, else replace with 0.

We wrap that in the factor() command to tell R it's working with a factor not a number, and we give it labels. Note that labels takes a list of strings, so we generate one using the c() [for concatenate] command.

We generate quartiles of slums using the ntile() command, which should be clear, and then wrap that again in the function()

We use filter() and select() from the tidyverse to check out some data on our new variables from Brazil and use the cc() function in epiDisplay to get a 2x4 table easily, alongside an OR.

We then conduct the logistic regression by piping in the new dataset with the added values. Here, we do not pipe to the first argument (which is always the formula for the regression), but instead pipe to where the fullstop is placed, in this case, the 'data' argument. Type ?glm if you need to see the full argument specification.

To run a regression in R of any kind, you need a formula. For an outcome Y and exposure X, with confounders C, the formula is set up as Y ~ X + C1 + C2 + C3 etc., where the tilde means 'predict what's on the left from on the right'. Note you can specify interactions with C1*C2 (which by default includes C1 + C2). Play around with the formula that is below (here we clearly have many continuous variables and a small sample size, so don't read into it too much, it's just a demo!!)

Note the useful logistic.display() function which handily provides Stata-like output - the standard glm output is presented on the log scale with CIs and ain't nobody got time for that.

```
#--- Get quantiles of TB Incidence & Slums
quantile(lmic$tb)
```

```
##   0%  25%  50%  75% 100%
##    7   52  159  306  834
```

```
quantile(lmic$slums)
```

```
##   0%  25%  50%  75% 100%
##  5.5 25.5 45.5 64.2 95.6
```

```
#--- Generate binary variable of TB Incidence & Slum Quartiles
lmic2 <- lmic %>% mutate(tb.binary = factor(ifelse(tb > 306, 1, 0),
                                            labels = c("High TB", "Low TB")),
                        slum.quart = factor(ntile(slums, 4),
                                             labels = c("Q1", "Q2", "Q3", "Q4")))

#--- Examine some select data about Brazil
lmic2 %>% filter(country == "Brazil") %>% dplyr::select(country, gdp, gini, tb, tb.binary, slum.quart)
```

```
##   country  gdp gini tb tb.binary slum.quart
## 1  Brazil 8650 51.3 41   High TB         Q1
```

```
#--- Get a 2x4 table
cc(lmic2$slum.quart, lmic2$tb.binary, graph = F) # NB exposure first!
```

```
##
##                  lmic2$tb.binary
## lmic2$slum.quart High TB Low TB Total
##            Q1         19      2    21
##            Q2         15      6    21
##            Q3         14      7    21
##            Q4         14      6    20
##            Total      62     21    83
##
## OR =  3.8
## 95% CI =  0.67, 21.6
## Chi-squared = 3.84, 3 d.f., P value = 0.28
## Fisher's exact test (2-sided) P value = 0.259
```

```r
#--- Conduct a logistic regression
lmic2 %>% glm(tb.binary ~ slum.quart + emp.ratio + lmic, family = binomial(link = "logit"), data = .) %
```

```
##
## Logistic regression predicting tb.binary : Low TB vs High TB
##
##                         crude OR(95%CI)    adj. OR(95%CI)
## slum.quart: ref.=Q1
##    Q2                   3.8 (0.67,21.59)   4.19 (0.67,26.07)
##    Q3                   4.75 (0.85,26.43)  6.18 (0.93,41.09)
##    Q4                   4.38 (0.76,25.2)   8.85 (1.09,71.66)
##
## emp.ratio (cont. var.) 0.99 (0.95,1.03)   0.98 (0.94,1.03)
##
## lmic: ref.=Low income
##    Lower middle income 2 (0.64,6.28)      3.04 (0.68,13.62)
##    Upper middle income 0.65 (0.14,2.97)   1.62 (0.23,11.52)
##
##                         P(Wald's test) P(LR-test)
## slum.quart: ref.=Q1                    0.142
##    Q2                   0.124
##    Q3                   0.06
##    Q4                   0.041
##
## emp.ratio (cont. var.) 0.487          0.486
##
## lmic: ref.=Low income                  0.28
##    Lower middle income 0.146
##    Upper middle income 0.632
##
## Log-likelihood = -42.2813
## No. of observations = 82
## AIC value = 98.5625
```

```r
#--- All together now...
# lmic %>% mutate(tb.binary = factor(ifelse(tb > quantile(tb)[4], 1, 0),
#                            labels = c("High TB", "Low TB")),
#                 slum.quart = factor(ntile(slums, 4),
#                            labels = c("Q1", "Q2", "Q3", "Q4"))) %>%
#         glm(tb.binary ~ slum.quart + emp.ratio + lmic,
#             family = binomial(link = "logit"),
#             data = .) %>%
#         logistic.display()

#--- Bonus: See if you can figure out what the [4] in quantile(tb)[4] does...
```

All of these operations were on a clean dataset. But real world datasets are rarely like that…so what do we do with a messy dataset? The tidyverse comes with the built in dataset from a Global TB report 2014.

This code is not meant for you to fully understand, but just to demonstrate how easy it is to clean data in R - a few short lines of code and the WHO data is clean and ready to analyse.

However, if you're interested…intuitively & step by step:

1. Gather – converts data from wide to long, i.e. it takes several 'key' columns (in this case, all the

age-sex-casetype disaggregated incidences) and condenses them into one column (which we have called "key") alongside a single value for each one which we have fittingly called "incidence".

2. Mutate – changes all instances of "newrel" in the key column to "new_rel" to make for consistent column names.
3. Separate – split the values in "key" into the three new columns "new", "casetype", and "sexage" (see now why we needed to do the step above?) – the tidyverse function automatically detects the underscore is the separator.
4. Select – drop redundant country name columns.
5. Separate – separate the sex and age variables at the first character.

```
#--- Confirm for yourself that this dataset is messy if you like
View(who)

#--- Cleaning
who %>%
  gather(key, incidence, new_sp_m014:newrel_f65, na.rm = TRUE) %>%
  mutate(key = stringr::str_replace(key, "newrel", "new_rel")) %>%
  separate(key, c("new", "casetype", "sexage")) %>%
  dplyr::select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1)
```

```
## # A tibble: 76,046 x 6
##    country      year casetype sex   age   incidence
##  * <chr>       <int> <chr>    <chr> <chr>     <int>
##  1 Afghanistan  1997 sp       m     014           0
##  2 Afghanistan  1998 sp       m     014          30
##  3 Afghanistan  1999 sp       m     014           8
##  4 Afghanistan  2000 sp       m     014          52
##  5 Afghanistan  2001 sp       m     014         129
##  6 Afghanistan  2002 sp       m     014          90
##  7 Afghanistan  2003 sp       m     014         127
##  8 Afghanistan  2004 sp       m     014         139
##  9 Afghanistan  2005 sp       m     014         151
## 10 Afghanistan  2006 sp       m     014         193
## # ... with 76,036 more rows
```

Some short exercises you might want to try in the following blank chunk:

1. Extract from the full SDG dataset the value of the UK's Gini coefficient
2. Extract from the full SDG dataset the median case detection rate in each WHO Region (hint: you may need to include an option to handle NAs, check ?median)
3. Produce boxplots for the above medians using ggplot() and geom_boxplot() with a black and white theme and a color for each region
4. In the eleven dataset, create quartiles of population density (pop.dens.q), label them, and get the maximum TB incidence in each one.
5. Run a linear regression [hint: family = gaussian or ?lm] using only data from African low income countries predicting TB incidence from slum prevalence adjusting for quartiles of population density [hint: regress.display() is useful here]

```
#--- 1
eleven %>% filter(country == "United Kingdom") %>% dplyr::select(country, gini)
```

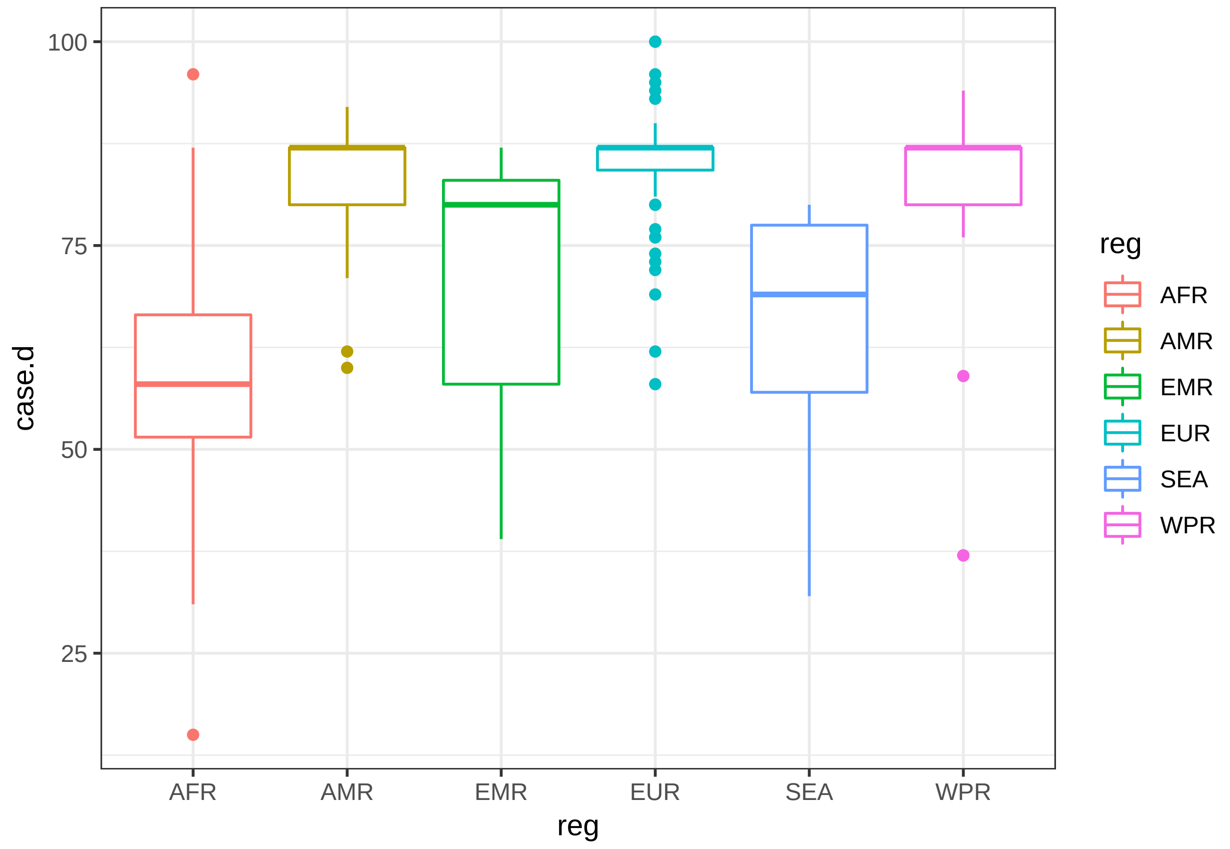
```
##          country gini
## 1 United Kingdom 34.1
```

```
#--- 2
eleven %>% group_by(reg) %>% summarise(median(case.d, na.rm = T))
```

```
## # A tibble: 6 x 2
##   reg   `median(case.d, na.rm = T)`
##   <fct>                       <dbl>
## 1 AFR                            58
## 2 AMR                            87
## 3 EMR                            80
## 4 EUR                            87
## 5 SEA                            69
## 6 WPR                            87
```

```
#--- 3
eleven %>% ggplot(aes(x = reg, y = case.d, color = reg)) + geom_boxplot() + theme_bw()
```

```
## Warning: Removed 12 rows containing non-finite values (stat_boxplot).
```



```
#--- 4
eleven %>% mutate(pop.dens.q = factor(ntile(pop.density, 5),
                                      labels = c("Q1", "Q2", "Q3", "Q4", "Q5"))) %>%
  group_by(pop.dens.q) %>%
  filter(complete.cases(tb)) %>%
  summarise(max(tb))
```

```
## # A tibble: 6 x 2
##   pop.dens.q `max(tb)`
##   <fct>           <dbl>
## 1 Q1                489
## 2 Q2                834
## 3 Q3                565
## 4 Q4                561
## 5 Q5                344
## 6 <NA>              146
```

```
#--- 5
eleven %>% filter(reg == "AFR" & lmic == "Low income") %>%
  mutate(pop.dens.q = factor(ntile(pop.density, 5), labels = c("Q1", "Q2", "Q3", "Q4", "Q5"))) %>%
  glm(tb ~ slums + pop.dens.q, family = gaussian, data = .) %>%
  regress.display()
```

```
## Linear regression predicting tb
##
##                       adj. coeff.(95%CI)      P(t-test) P(F-test)
## slums (cont. var.)    3.34 (0.03,6.66)        0.048     0.046
##
## pop.dens.q: ref.=Q1                                     0.106
##    Q2                 178.8 (13.18,344.42)    0.036
##    Q3                 73.26 (-90.57,237.08)   0.36
##    Q4                 -7.03 (-160.69,146.63)  0.924
##    Q5                 -10.05 (-177.41,157.3)  0.901
##
## Log-likelihood = -143.9506
## No. of observations = 24
## AIC value = 301.9011
```