Relationship between global cesarean delivery rates and GDP

Nimita Gaggar

July 12, 2023

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(readr)
library(broom)
library(testthat)
```

```
##
## Attaching package: 'testthat'

## The following objects are masked from 'package:readr':
##
##     edition_get, local_edition
## The following object is masked from 'package:dplyr':
##
##     matches
```

```
CS_data <- read_csv("data/cesarean.csv")
```

```
## Rows: 137 Columns: 7
```

```
## --· Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): Country_Name, CountryCode, Income_Group, Region
## dbl (3): Births_Per_1000, GDP_2006, CS_rate
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

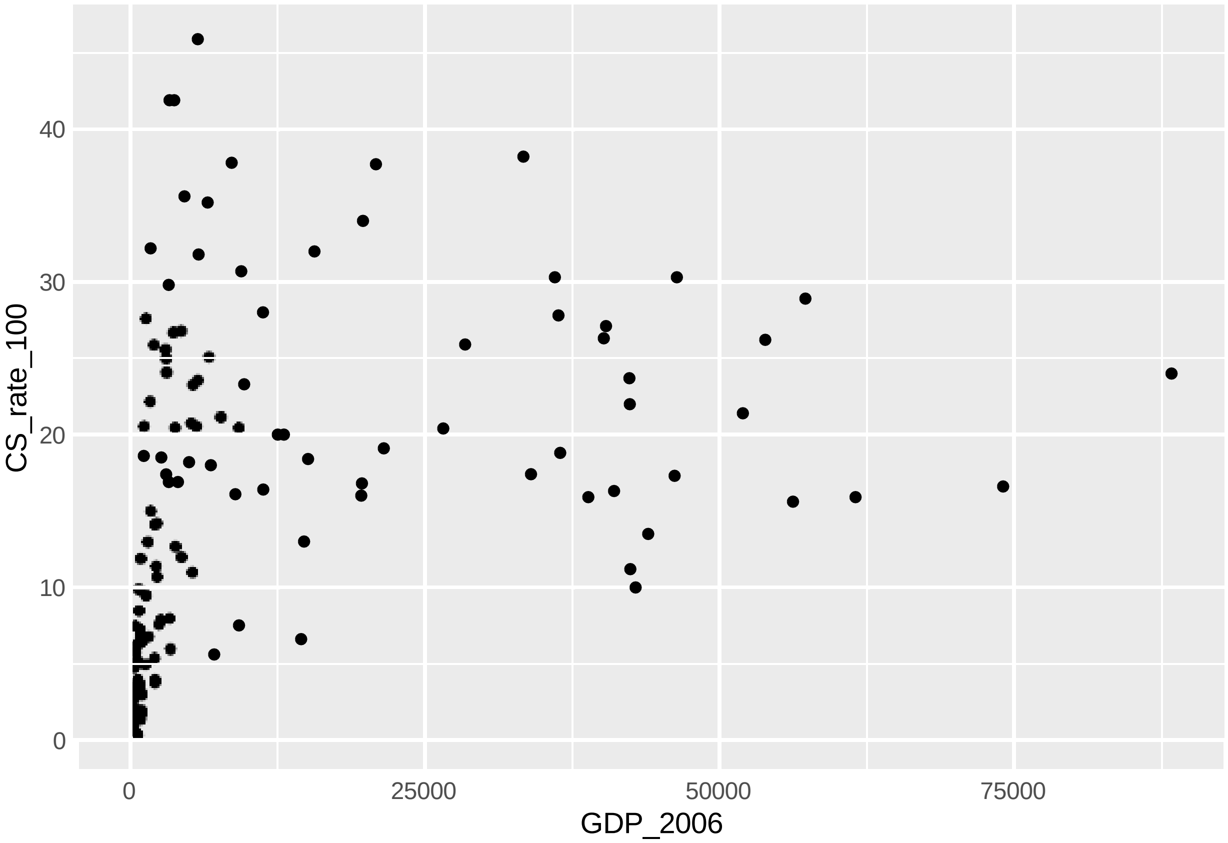
```
# The code below re-orders the variable Income_Group in the specified order.
# Note that it *does not* change the order of the data frame (like arrange() does)
# Rather, it specifies the order the data will be plotted.
# This will make more sense when we plot the data using Income_Group, and then again using  Income_Group
CS_data$Income_Group  <-  forcats::fct_relevel(CS_data$Income_Group,
                                          "Low income", "Lower middle income",
                                    "Upper middle income", "High income: nonOECD",
                                            "High income: OECD")

CS_data <- CS_data %>% mutate(CS_rate_100 = CS_rate*100)
CS_data
```

```
## # A tibble: 137 x 8
##    Country_Name CountryCode Births_Per_1000 Income_Group Region GDP_2006 CS_rate
##    <chr>        <chr>                 <dbl> <fct>        <chr>     <dbl>   <dbl>
##  1 Albania      ALB                      46 Upper middl~ Europ~    3052.   0.256
##  2 Andorra      AND                       1 High income~ Europ~   42417.   0.237
##  3 United Arab~ ARE                      63 High income~ Middl~   42950.   0.1
##  4 Argentina    ARG                     689 High income~ Latin~    6649.   0.352
##  5 Armenia      ARM                      47 Lower middl~ Europ~    2127.   0.141
##  6 Australia    AUS                     267 High income~ East ~   36101.   0.303
##  7 Austria      AUT                      76 High income~ Europ~   40431.   0.271
##  8 Azerbaijan   AZE                     166 Upper middl~ Europ~    2473.   0.076
##  9 Belgium      BEL                     119 High income~ Europ~   38936.   0.159
## 10 Benin        BEN                     342 Low income   Sub-S~     557.   0.036
## # i 127 more rows
## # i 1 more variable: CS_rate_100 <dbl>
```

1. [1 point] Make a scatter plot between CS_rate_100 and GDP_2006.

```
p1 <- ggplot(CS_data, aes(x = GDP_2006, y = CS_rate_100)) +
        geom_point()
p1
```



```
. = ottr::check("tests/p1.R")
```

```
##
## All tests passed!
```

In your plot, you might notice that many of the points are condensed towards the lower left corner. And you might recall from the lab and assignment that the distributions of both cesarean delivery rate and GDP covered a wide range of values. Both of these variables are good candidates for log transformations to spread out the range of data at the lowest levels.

2. [1 point] Using the mutate() function, add two new logged variables to the CS_data dataset and assign this new dataset to CS_data_log. Call the variables log_CS and log_GDP. Use base e, also known as the natural logarithm, to create the logged variables.

```
CS_data_log <- CS_data %>% mutate (log_CS = log(CS_rate_100),
                                   log_GDP= log (GDP_2006))
CS_data_log
```

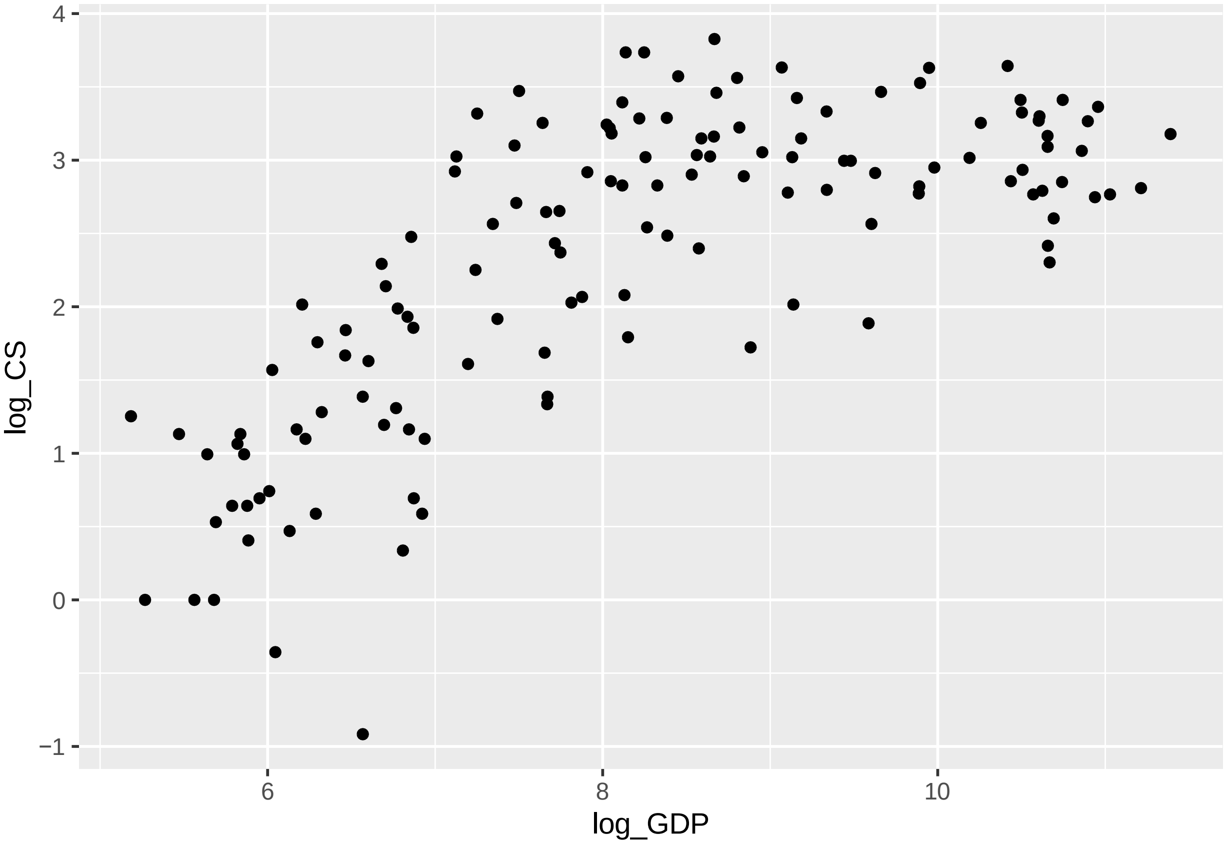
```
## # A tibble: 137 x 10
##    Country_Name CountryCode Births_Per_1000 Income_Group Region GDP_2006 CS_rate
##    <chr>        <chr>                 <dbl> <fct>        <chr>     <dbl>   <dbl>
##  1 Albania      ALB                      46 Upper middl~ Europ~    3052.   0.256
##  2 Andorra      AND                       1 High income~ Europ~   42417.   0.237
##  3 United Arab~ ARE                      63 High income~ Middl~   42950.   0.1  
##  4 Argentina    ARG                     689 High income~ Latin~    6649.   0.352
##  5 Armenia      ARM                      47 Lower middl~ Europ~    2127.   0.141
##  6 Australia    AUS                     267 High income~ East ~   36101.   0.303
##  7 Austria      AUT                      76 High income~ Europ~   40431.   0.271
##  8 Azerbaijan   AZE                     166 Upper middl~ Europ~    2473.   0.076
##  9 Belgium      BEL                     119 High income~ Europ~   38936.   0.159
## 10 Benin        BEN                     342 Low income   Sub-S~     557.   0.036
## # i 127 more rows
## # i 3 more variables: CS_rate_100 <dbl>, log_CS <dbl>, log_GDP <dbl>
```

```
. = ottr::check("tests/p2.R")
```

```
## 
## All tests passed!
```

3. [1 point] Remake the scatter plot using the logged variables.

```
p3 <-ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) +
        geom_point()
p3
```



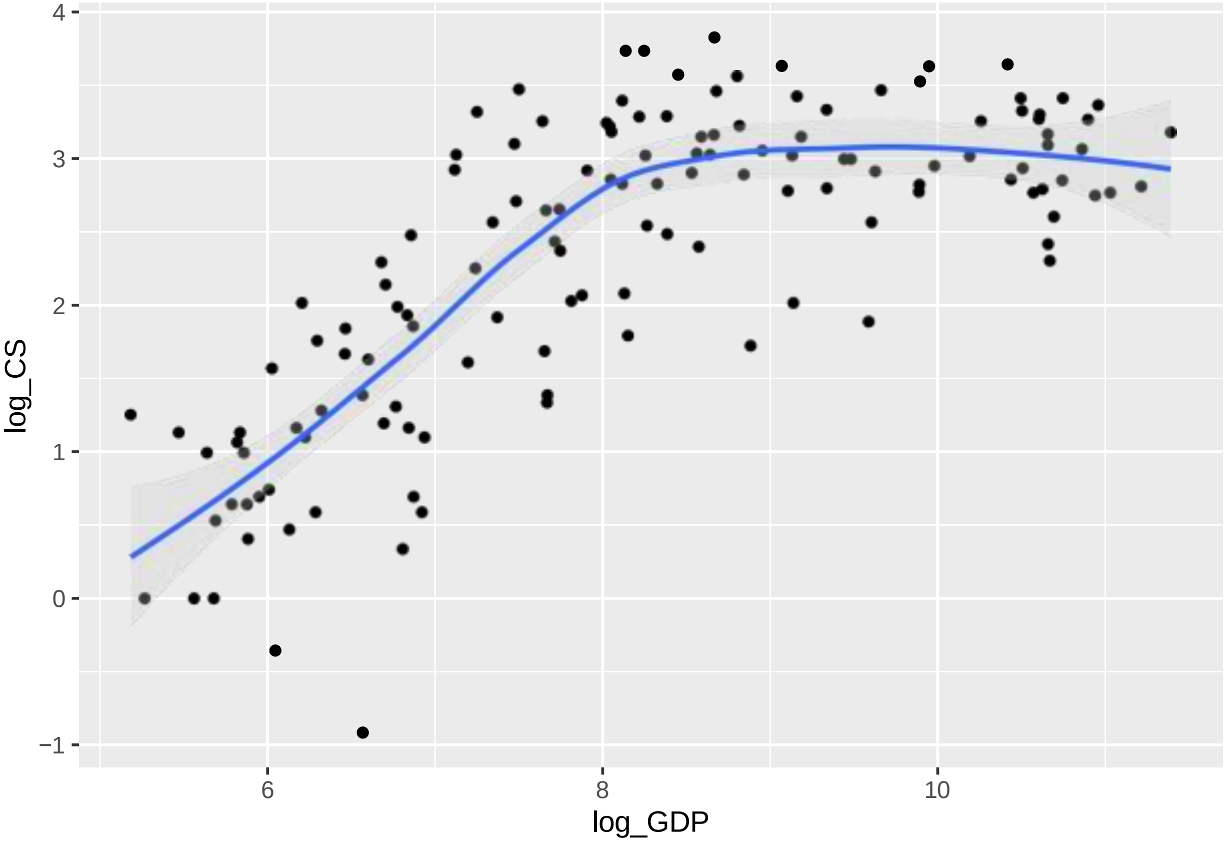
```
. = ottr::check("tests/p3.R")
```

```
##
## All tests passed!
```

4. [1 point] A geom that you have not yet learned is geom_smooth(). This geom can fit a curve to the data. Extend your ggplot() code by adding geom_smooth() to it.

```
p4 <- ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) +
        geom_point() + geom_smooth()
p4
```

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



```
. = ottr::check("tests/p4.R")
```

```
##
## All tests passed!
```

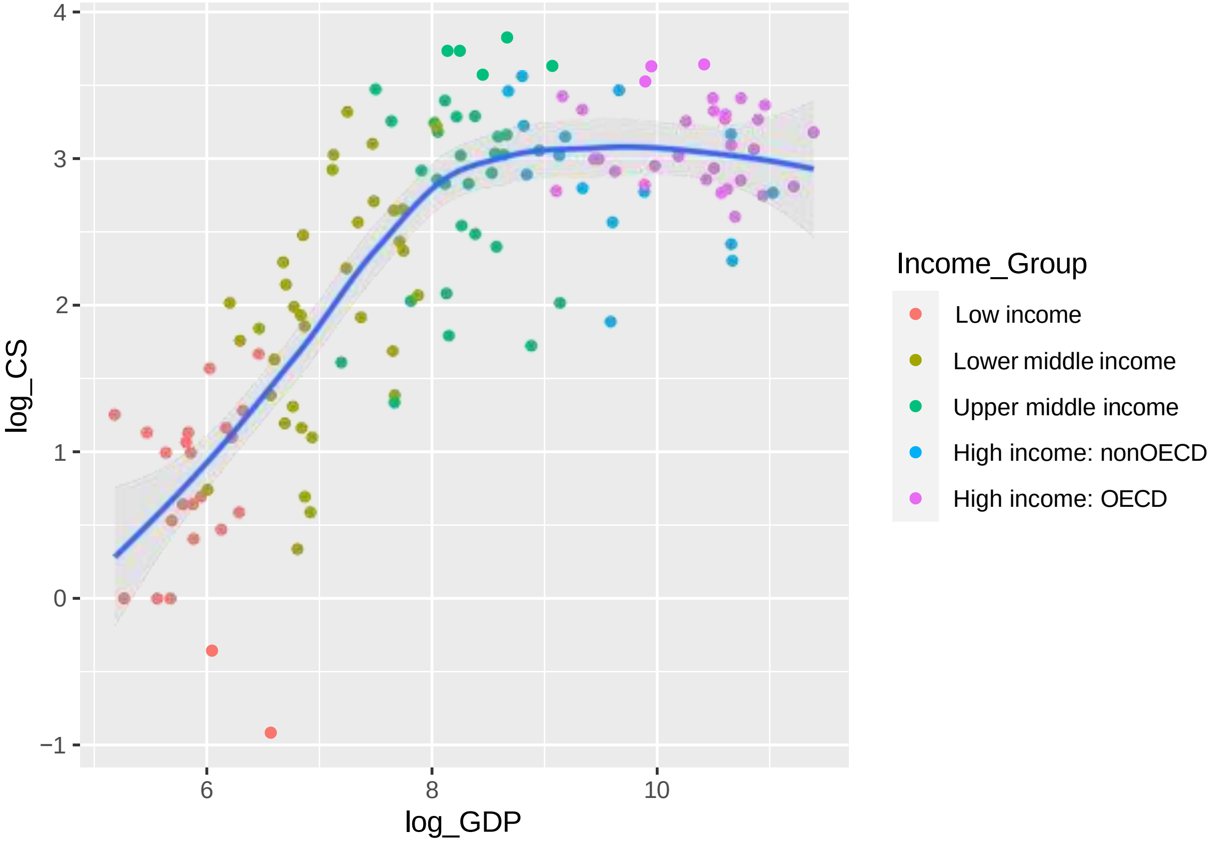
5. Does the relationship between logged GDP and logged CS look linear?

No. It appears to be curved around avalue of 8 for the log_GDP .

6. [1 point] Modify your scatter plot by linking the color of the points to the variableIncome_Group.

```
p6 <- ggplot(CS_data_log, aes(x = log_GDP, y = log_CS)) +
        geom_point(aes(color = Income_Group)) + geom_smooth()
p6
```

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



```
. = ottr::check("tests/p6.R")
```

```
##
## All tests passed!
```

Does a linear relationship hold for any part of the data? What pattern do you notice? Only for high income group group and not the other groups

7. [1 point] For this lab, we would like to use linear regression. To do this, use a dplyr function to make a new dataset called CS_data_sub that only contains the low-, lower-middle, and upper-middle income countries (hint: You might want to look at the data to see exactly what these levels are called in the data set).

```
CS_data_sub <- CS_data_log %>%
  filter(Income_Group %in% c ('Low income', 'Lower middle income', 'Upper middle income'))
CS_data_sub
```

```
## # A tibble: 91 x 10
##    Country_Name CountryCode Births_Per_1000 Income_Group Region GDP_2006 CS_rate
##    <chr>        <chr>                 <dbl> <fct>        <chr>     <dbl>   <dbl>
##  1 Albania      ALB                      46 Upper middl~ Europ~    3052.   0.256
##  2 Armenia      ARM                      47 Lower middl~ Europ~    2127.   0.141
##  3 Azerbaijan   AZE                     166 Upper middl~ Europ~    2473.   0.076
##  4 Benin        BEN                     342 Low income   Sub-S~     557.   0.036
##  5 Burkina Faso BFA                     721 Low income   Sub-S~     422.   0.007
##  6 Bangladesh   BGD                    3430 Lower middl~ South~     496.   0.075
##  7 Bulgaria     BGR                      73 Upper middl~ Europ~    4371.   0.268
##  8 Belarus      BLR                      96 Upper middl~ Europ~    3849.   0.205
##  9 Bolivia      BOL                     263 Lower middl~ Latin~    1234.   0.186
## 10 Brazil       BRA                    3105 Upper middl~ Latin~    5809.   0.459
## # i 81 more rows
## # i 3 more variables: CS_rate_100 <dbl>, log_CS <dbl>, log_GDP <dbl>
```

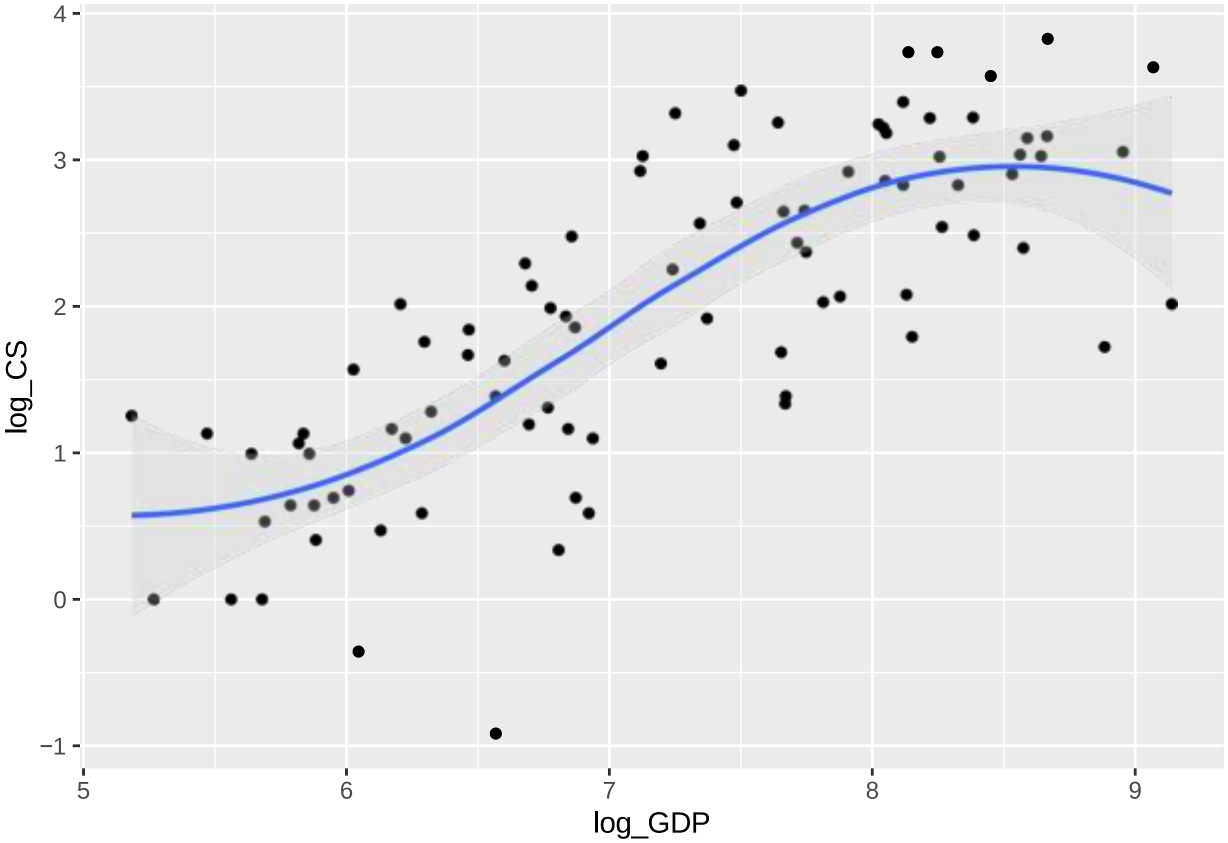
```
. = ottr::check("tests/p7.R")
```

```
##
## All tests passed!
```

8. [1 point] Remake the last scatter plot, this time using CS_data_sub to see if the relationship between the logged variables looks approximately linear.

```
p8 <- ggplot(CS_data_sub, aes(x = log_GDP, y = log_CS)) +
        geom_point() + geom_smooth()
p8
```

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



```
. = ottr::check("tests/p8.R")
```

```
##
## All tests passed!
```

9. [1 point] Given that the relationship is approximately linear, use linear regression to model the relationship between log_CS as the response variable and log_GDP as the explanatory variable. Don't forget to specify the correct dataset!

```
p9 <- lm(log_CS ~ log_GDP, data = CS_data_sub)
p9
```

```
##
## Call:
## lm(formula = log_CS ~ log_GDP, data = CS_data_sub)
```

```
##
## Coefficients:
## (Intercept)      log_GDP
##     -3.9405       0.8193
```

```
. = ottr::check("tests/p9.R")
```

```
##
## All tests passed!
```

10. Interpret the slope estimate in the context of the problem.

The slope can be interpreted as: for every one unit increase in Log GDP, the log CS Delivery rate is expected to increase by 0.8193

11. Estimate what the cesarean delivery rate would be for a country with a GDP of 2000. Outline the steps you take to calculate your answer and provide an interpretation. Round your final answer to one decimal place.

The cesarean delivery rate would be 9.8 units for a country with GDP of 2000.

```
#Log_CS = intercept + slope(Log_GDP)

#calculated_output
-3.9405 + 0.8193* log(2000)
```

```
## [1] 2.286919
```

```
#transformed_output
exp (2.286919)
```

```
## [1] 9.84456
```

12. Is it appropriate to use the model to predict the cesarean delivery rate for a country with a GDP of 50,000? Why or why not? Based on the relationship in the full dataset, would you expect the linear model to over- or under-predict? The cesarean delivery rate would be 137.57 units for a country with GDP of 50000. The linear model is over-predicted based on the relationship in the full data set.

```
#Log_CS = intercept + slope(Log_GDP)

#calculated_output
log_CS = -3.9405 + 0.8193* log(50000)

#transformed_output
exp (4.924144)
```

```
## [1] 137.5715
```