STAT 374 Final Project Report

Xiaoyu Sun (12207473)

2019/11/30

1 Description of the data

In this project, the dataset I study is the annual suicide rates of 101 countries in the world with their socio-economic information from 1985 to 2016, downloaded from https://www.kaggle.com/russellyates88/suicide-rates-overview-1985-to-2016. This dataset was in turn collected from four sources: human development reports from United Nations Development Program, each country's GDP from World Bank, suicide data from World Health Organization, and one Kaggle notebook (see the previous URL for details).

The original dataset contains 27820 entries and 12 columns, providing the following information:

- `country` - country name
- `year` - ranging from 1985 to 2016
- `sex` - male / female
- `age` - age groups, which include 5-14, 15-24, 25-34, 35-54, 55-74, and 75+ years
- `suicides_no` - number of people who committed suicide
- `population` - population size of that specific country, year, sex and age group
- `suicides_100k` - suicide rate, represented by number of suicides per 100000 people
- `country.year` - a string combining the country name and year
- `HDI.for.year` - HDI (human development index)
- `gdp_for_year` - the country's annual GDP
- `gdp_per_capita` - the country's GDP per capita
- `generation` - in chronological order, this can be "G.I. Generation", "Silent", "Boomers", "Generation X", "Millenials", "Generation Z".

The **objective** of my analysis is to **model the suicide rate of youths corresponding to the age group of 15-24** (for Asian, European, and American countries mainly).

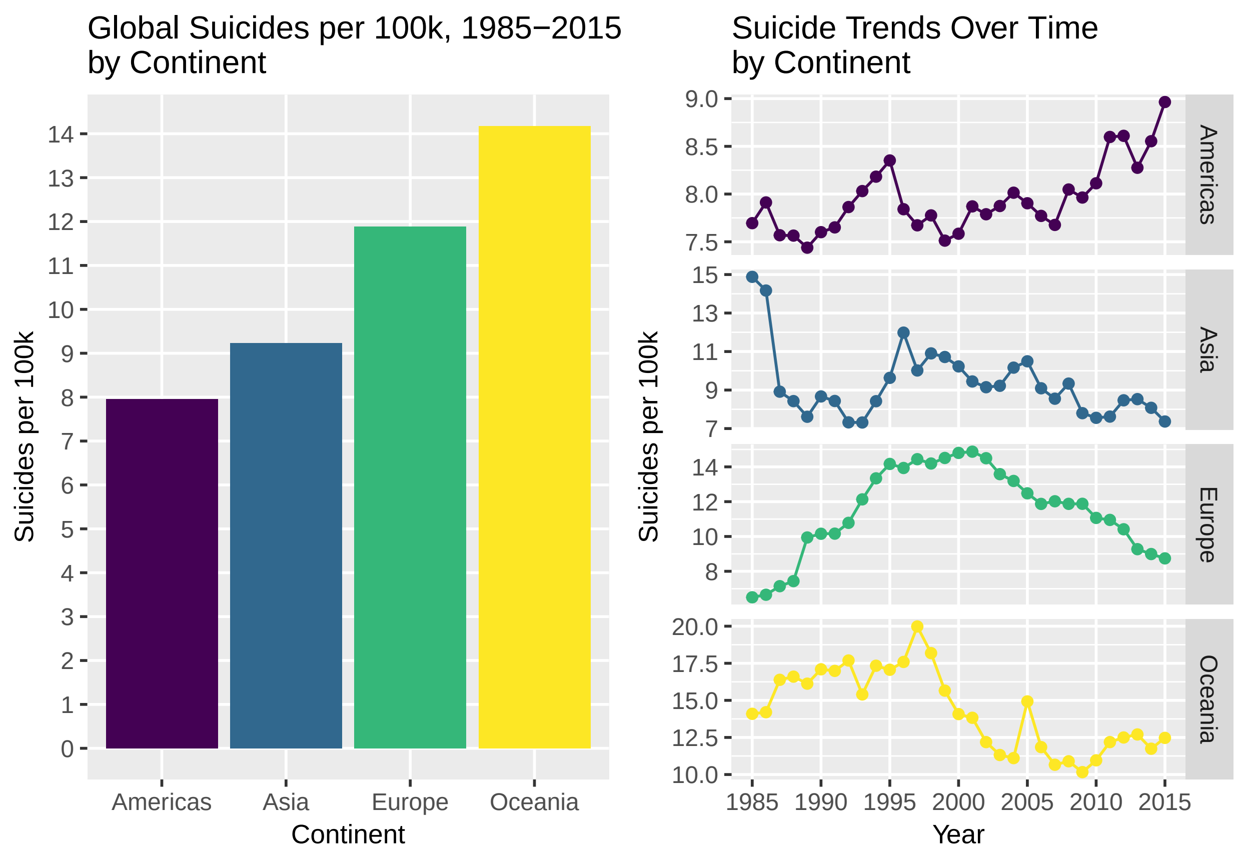
Before searching for appropriate models, I carried out some cleaning and preprocessing on the original data. First of all, I filtered the data to keep only the 15-24 age group, and thus the `age` variable is no longer needed. I also removed those countries with no more than 3 years of data. 2016 data was removed because few countries had records for this year, and for similar reason the `HDI.for.year` column was discarded. I decided to get rid of `generation`, because an entire age group in a given year cannot necessarily be classified as one generation.

Then I added a new column `continent` to the data, indicating which continent each country belongs to. Only 3 countries are from Africa, 4 from Oceania, 21 from Asia, 29 from North/South America, and 36 from Europe. Given the extremely small number of countries from Africa, I decided to exclude Africa from subsequent analysis.

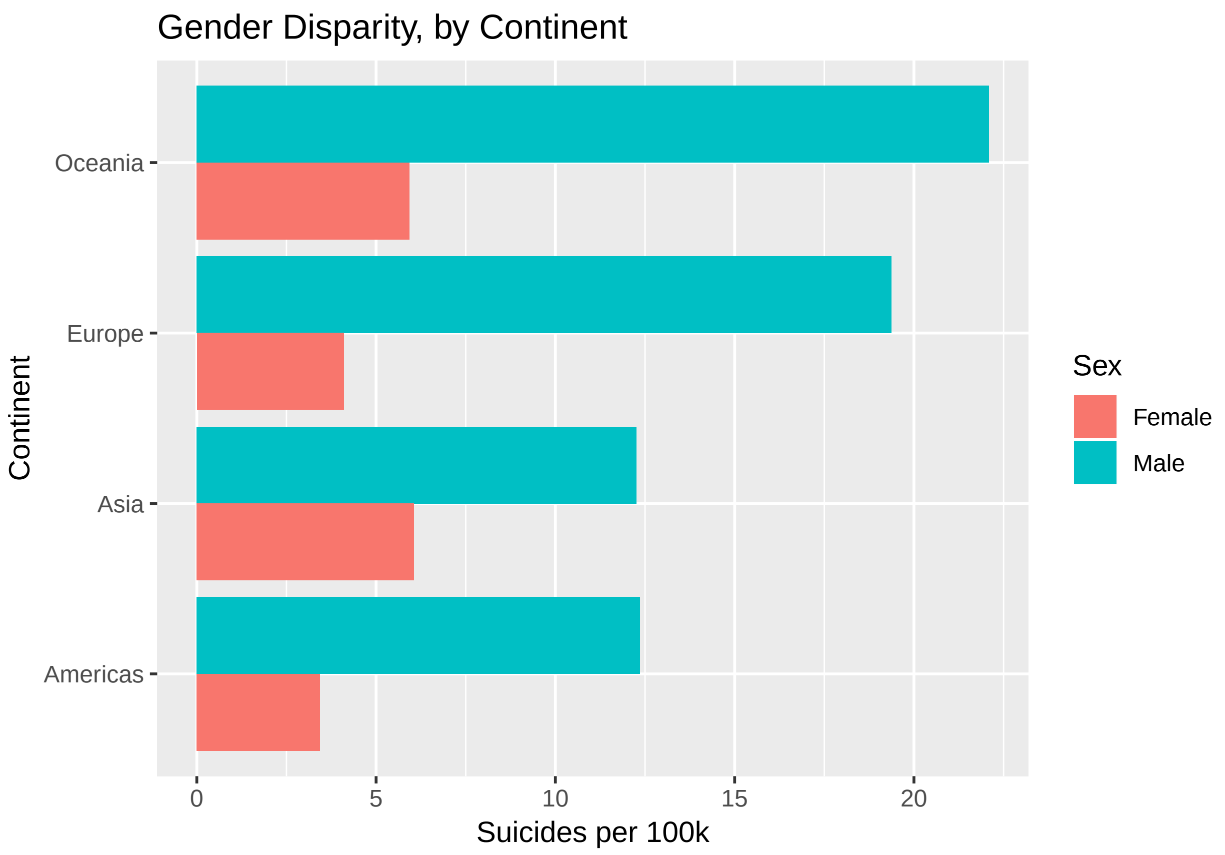
```r
library(countrycode)
library(dplyr)
data = read.csv("master.csv")
names(data)[names(data) == "gdp_for_year...."] = "gdp_for_year"
names(data)[names(data) == "gdp_per_capita...."] = "gdp_per_capita"
names(data)[names(data) == "suicides.100k.pop"] = "suicides_100k"
data$continent = countrycode(sourcevar = data[, "country"],
                             origin = "country.name",
                             destination = "continent")
data = data[data$age == "15-24 years",]
data = data %>%
  select(-c("age", "HDI.for.year", "generation", "country.year")) %>%
  filter(year != 2016)
minimum_years = data %>%
  group_by(country) %>%
  summarize(rows=n(), years=rows/2) %>%
  arrange(years)
data = data %>%
  filter(!(country %in% head(minimum_years$country, 7))) %>%
  filter(continent %in% c("Asia", "Americas", "Europe", "Oceania"))

data$sex <- ifelse(data$sex == "male", "Male", "Female")
# Nominal factors
data_nominal <- c('country', 'sex', 'continent')
data[data_nominal] <- lapply(data[data_nominal], function(x){factor(x)})
```

Here are some visualizations of the data. Looking by continent, we see that America (North & South combined) has the lowest suicide rate over the 1985-2015 period, but it is also the only continent that exhibits an increasing trend during recent years. Looking by gender, we see that suicide rate of male is much higher than that of female, no matter in which continent. Looking at the scatterplot of each country's suicide rate against GDP per capita (averaged across the years), we do not observe a very clear correlation between these two variables; on the other hand, Sri Lanka (the point in the upper-left corner) seems to be an outlier, so I removed this country to prevent influencing our subsequent models too much. After all these preprocessing steps, we are left with 89 countries.

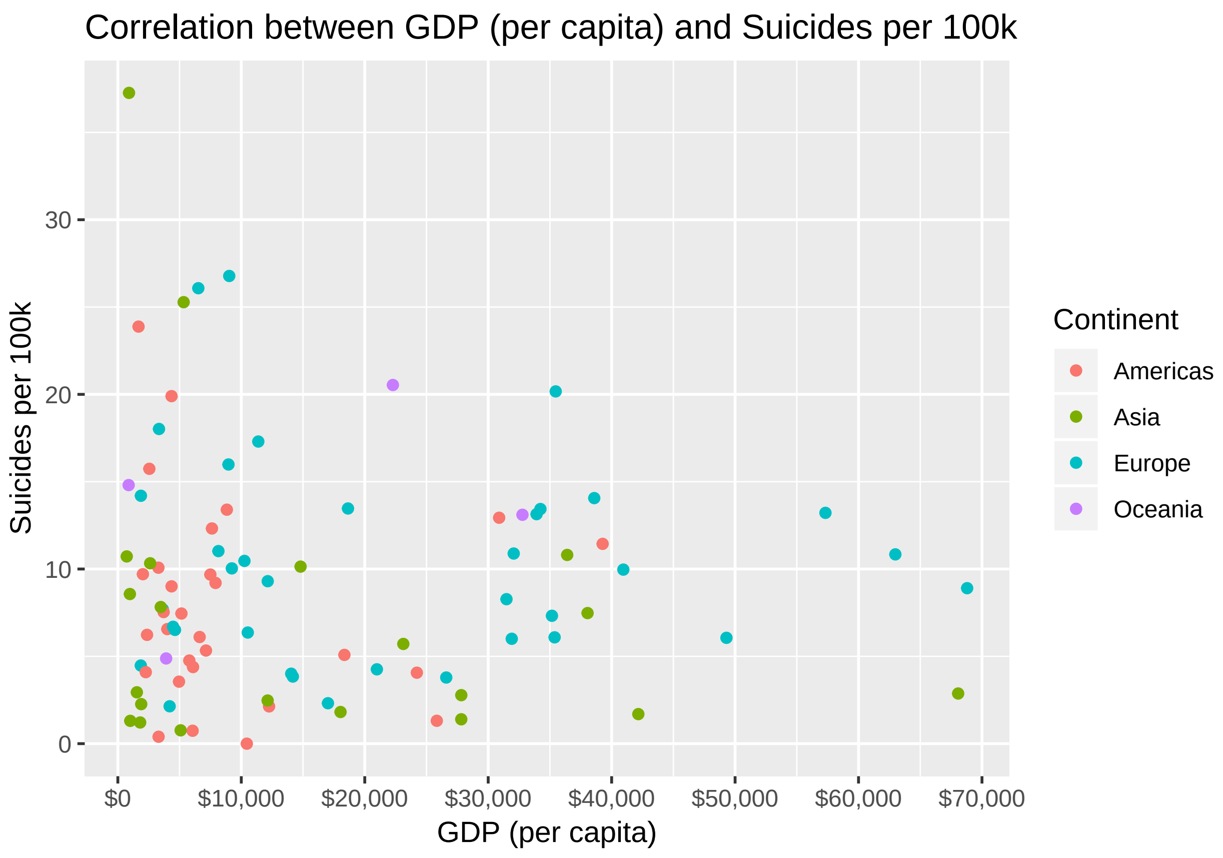
```r
library(ggplot2)
library(gridExtra)
continent = data %>%
  group_by(continent) %>%
  summarize(suicide_per_100k = (sum(as.numeric(suicides_no)) / sum(as.numeric(population))) * 100000) %
  arrange(suicide_per_100k)

continent$continent <- factor(continent$continent, ordered = T, levels = continent$continent)

continent_plot <- ggplot(continent, aes(x = continent, y = suicide_per_100k, fill = continent)) +
  geom_bar(stat = "identity") +
  labs(title = "Global Suicides per 100k, 1985-2015\nby Continent",
  x = "Continent",
  y = "Suicides per 100k",
  fill = "Continent") +
  theme(legend.position = "none", title = element_text(size = 10)) +
  scale_y_continuous(breaks = seq(0, 20, 1), minor_breaks = F)

continent_time <- data %>%
  group_by(year, continent) %>%
  summarize(suicide_per_100k = (sum(as.numeric(suicides_no)) / sum(as.numeric(population))) * 100000)

continent_time$continent <- factor(continent_time$continent, ordered = T, levels = continent$continent)

continent_time_plot <- ggplot(continent_time, aes(x = year, y = suicide_per_100k, col = factor(continen
  facet_grid(continent ~ ., scales = "free_y") +
  geom_line() +
  geom_point() +
  labs(title = "Suicide Trends Over Time \nby Continent",
       x = "Year",
       y = "Suicides per 100k",
       color = "Continent") +
  theme(legend.position = "none", title = element_text(size = 10)) +
  scale_x_continuous(breaks = seq(1985, 2015, 5), minor_breaks = F)

grid.arrange(continent_plot, continent_time_plot, ncol = 2)
```



```
data %>%
  group_by(continent, sex) %>%
  summarize(n = n(),
            suicides = sum(as.numeric(suicides_no)),
            population = sum(as.numeric(population)),
            suicide_per_100k = (suicides / population) * 100000) %>%
  ggplot(aes(x = continent, y = suicide_per_100k, fill = sex)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Gender Disparity, by Continent",
   x = "Continent",
   y = "Suicides per 100k",
   fill = "Sex") +
  coord_flip()
```



```
country_mean_gdp <- data %>%
  group_by(country, continent) %>%
  summarize(suicide_per_100k = (sum(as.numeric(suicides_no)) / sum(as.numeric(population))) * 100000,
            gdp_per_capita = mean(gdp_per_capita))

ggplot(country_mean_gdp, aes(x = gdp_per_capita, y = suicide_per_100k, col = continent)) +
  geom_point() +
  scale_x_continuous(labels=scales::dollar_format(prefix="$"), breaks = seq(0, 70000, 10000)) +
  labs(title = "Correlation between GDP (per capita) and Suicides per 100k",
       x = "GDP (per capita)",
       y = "Suicides per 100k",
       col = "Continent")
```



```
# remove Sri Lanka
data = data %>% filter(country != "Sri Lanka")
```

Code for these descriptive plots is adapted from https://www.kaggle.com/lmorgan95/r-suicide-rates-in-depth-stats-insights

2 Parametric analysis

For this part, we consider fitting linear model with interaction terms. Such models contain the following assumptions: (1) linear relationship between response and predictors; (2) little or no collinearity between predictors; (3) error terms (residuals) follow normal distribution, and have constant variance; (4) little or no autocorrelation in the residuals.

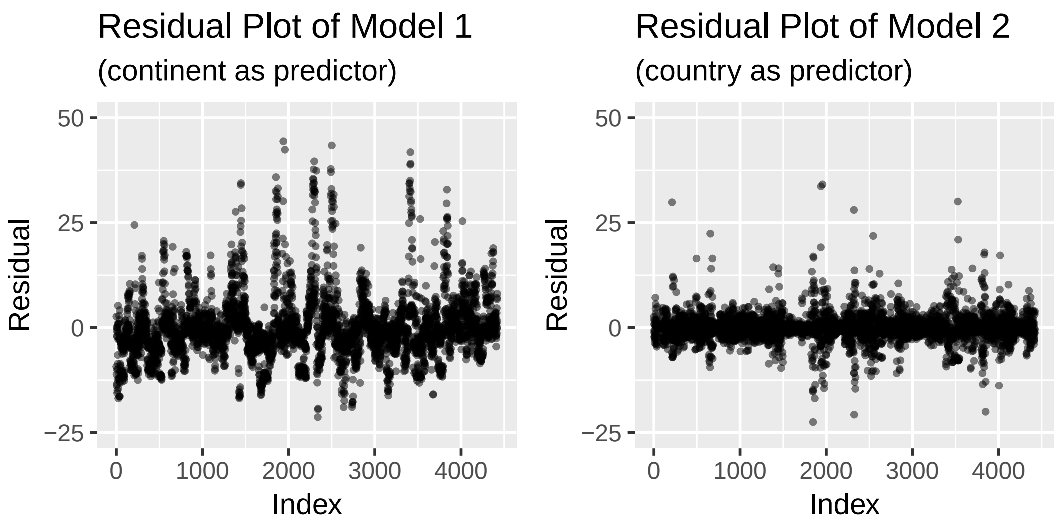
Model 1: First, let's include `year`, `sex`, `continent` as categorical variables, and the quantitative variable `gdp_per_capita`. We also include interaction terms `year:continent`, `sex:continent` and `sex:gdp_per_capita`, so that each year and gender for different continents can have different intercepts, and each sex can take on different slopes with respect to GDP per capita. The model expression in R is `lm(suicides_100k ~ sex * gdp_per_capita + (year + sex) * continent, data = ...)`.

This is a pretty poor fit. The adjusted R-squares (R^2_{adj}), which is indicative of goodness of fit, is only 0.3282. The residual plot shows that a lot of the residuals exceed 10 in absolute value, which is bad given the scale of our target variable.

Model 2: As an alternative, we try using `country` instead of `continent` as a predictor. In this case, we treat `year` as a quantitative variable rather than a factor, otherwise the `year:country` interaction term alone would produce about $89 \times 21 = 1869$ different levels, which would definitely lead to overfitting. We also include a quadratic term for `year` (and its interaction with `country`) to allow for nonlinear trends over time. Other interaction terms are `sex:country` and `sex:gdp_per_capita`. The model expression in R is `lm(suicides_100k ~ sex * gdp_per_capita + (I(year^2) + year + sex) * country, data = ...)`.

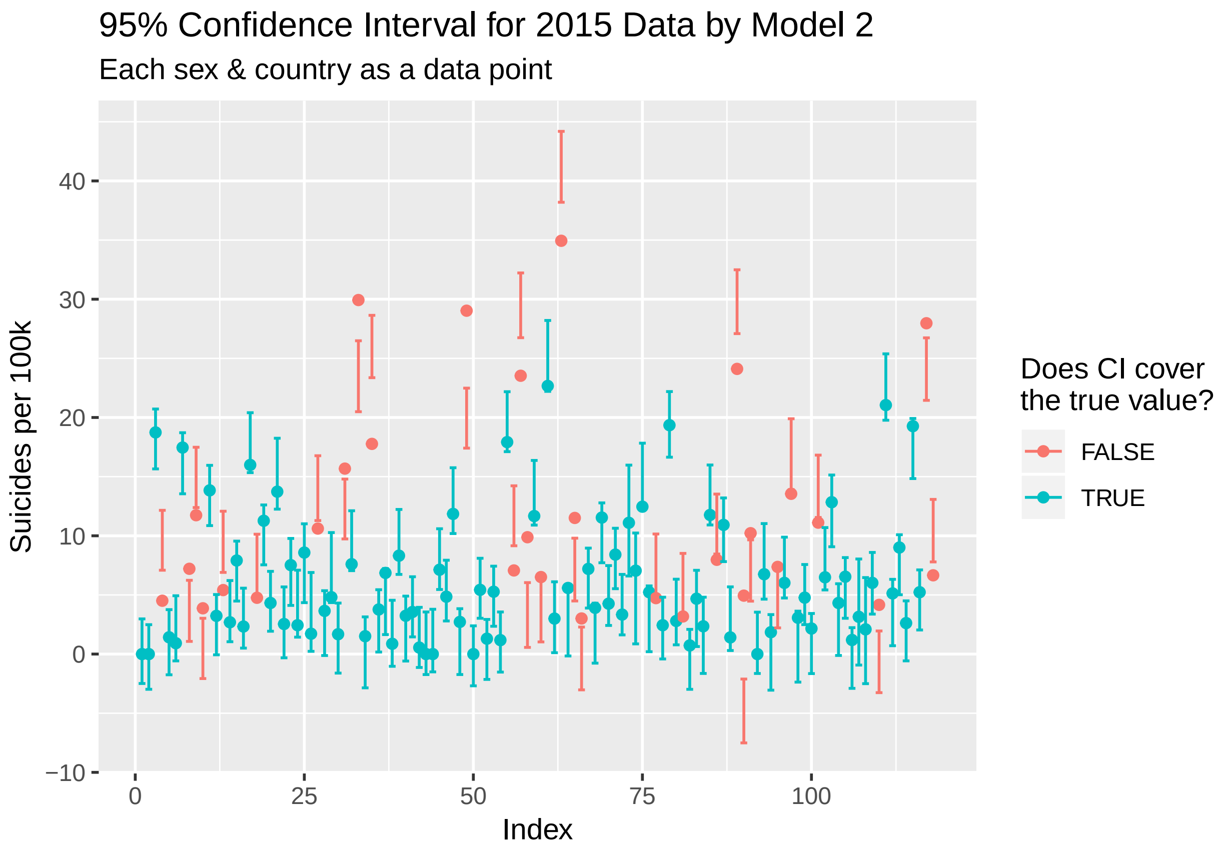
This model provides a much better fit than Model 1. Now the R^2_{adj} is 0.8682, and the residuals are much more concentrated around 0. Based on comparison of fit, we choose Model 2 as our parametric model.

```r
data1 = data.frame(data)
data1$year = as.factor(data1$year)
m1 = lm(suicides_100k ~ sex * gdp_per_capita + (year + sex) * continent, data=data1)

# now treat year as quantitative variable
m2 = lm(suicides_100k ~ sex * gdp_per_capita + (I(year^2) + year + sex) * country, data=data)

temp1 = data.frame("index" = 1:length(m1$residuals), "residual" = m1$residuals)
plot1 = ggplot(temp1, aes(x=index, y=residual)) + geom_point(alpha=0.5, size=0.7) +
  ylim(-25, 50) +
  labs(title = "Residual Plot of Model 1", subtitle = "(continent as predictor)",
       x = "Index", y = "Residual")

temp2 = data.frame("index" = 1:length(m2$residuals), "residual" = m2$residuals)
plot2 = ggplot(temp2, aes(x=index, y=residual)) + geom_point(alpha=0.5, size=0.7) +
  ylim(-25, 50) +
  labs(title = "Residual Plot of Model 2", subtitle = "(country as predictor)",
       x = "Index", y = "Residual")

grid.arrange(plot1, plot2, ncol = 2)
```



To get an idea of the level of uncertainty in our estimates, we plot the 95% confidence interval for each data point of year 2015 (The entire dataset contains 4422 entries, which make it impossible to plot here). Of all the 118 data points, 88 are contained in their corresponding confidence intervals. This coverage probability of 74.6% indicates that our fit is still not ideal.

```r
temp = data %>% filter(year == 2015)
m2.CI = data.frame(predict(m2, newdata = temp, interval="confidence"))
m2.CI$true = temp$suicides_100k
m2.CI$CI_covers_true_value = (m2.CI$true >= m2.CI$lwr & m2.CI$true <= m2.CI$upr)

ggplot(m2.CI, aes(x=1:nrow(m2.CI), y=true, col=CI_covers_true_value)) +
  geom_point() +
  geom_errorbar(aes(ymin=lwr, ymax=upr), width=1) +
  labs(title = "95% Confidence Interval for 2015 Data by Model 2",
       subtitle = "Each sex & country as a data point",
       x = "Index",
       y = "Suicides per 100k",
       col = "Does CI cover\nthe true value?")
```



3 Nonparametric analysis

In this section, we switch to nonparametric models. Here is an application.

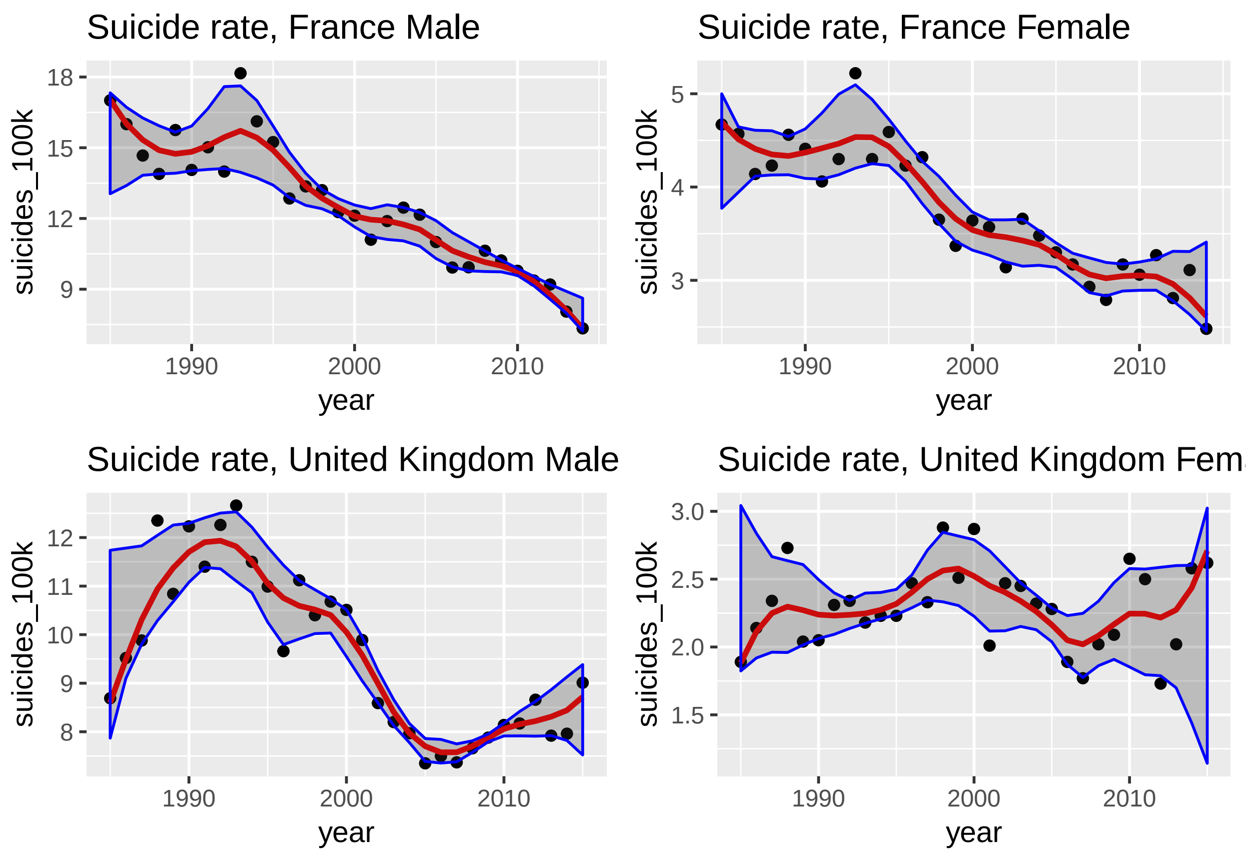
Model 3: For each combination of `country` and `sex`, we fit a **local linear regression** of `suicides_100k` with respect to `year`, with bandwidth chosen by cross validation. For this local linear regression, we do not need those assumptions which are required by the parametric models in the previous section; instead, we only use a much weaker assumption of a smooth population regression function. The model is fitted using `locfit` package in R.

To obtain a measure of uncertainty, we use bootstrap to get 95% confidence bands for each combination of `country` and `sex`. Specifically, we sample with replacement from the data points, fit the local linear regression, and calculate point estimates at each year for 500 times; then we use the 2.5- and 97.5-percentile of point estimates for each year as the corresponding confidence bands. As an example, I have plotted the fitted values and 95% confidence bands for France and United Kingdom.

```
library(locfit)
country.all = unique(data$country)
model.list = vector(mode="list", length=length(country.all))
```

```r
for (i in 1:length(country.all)) {
  temp = data %>% filter(country == country.all[i])
  male = temp %>% filter(sex == "Male")
  female = temp %>% filter(sex == "Female")
  model.list[[i]] = list(
    locfit(male$suicides_100k ~ male$year, alpha=c(0,4), deg=1, maxk=1000),
    locfit(female$suicides_100k ~ female$year, alpha=c(0,4), deg=1, maxk=1000))
}
```

```r
get_plot = function(country.name, sex.name) {
  temp = data %>% filter(country == country.name & sex == sex.name)
  ind = which(country.all == country.name)
  if (sex.name == "Male")
    temp$pred = predict(model.list[[ind]][[1]], newdata=temp$year)
  else
    temp$pred = predict(model.list[[ind]][[2]], newdata=temp$year)
  temp$lwr = 0
  temp$upr = 0

  n = nrow(temp)
  set.seed(1202)
  bs.values = matrix(0, nrow=n, ncol=500)
  for (iter in 1:500) {
    newdata = temp[sample(n, n, replace=TRUE), ]
    fit = locfit(newdata$suicides_100k ~ newdata$year, alpha=c(0,4), deg=1, maxk=1000)
    bs.values[,iter] = predict(fit, newdata=temp$year)
  }
  for (i in 1:n) {
    temp$lwr[i] = quantile(bs.values[i,], 0.025)
    temp$upr[i] = quantile(bs.values[i,], 0.975)
  }

  result = ggplot(temp, aes(x=year, y=suicides_100k)) +
    geom_point() +
    geom_line(aes(y=pred), size=1, col="red") +
    geom_ribbon(aes(ymin=lwr, ymax=upr, col=NULL), alpha=0.25, col="blue") +
    labs(title = paste("Suicide rate,", country.name, sex.name),
         xlab = "Year",
         ylab = "Suicides per 100k")
  return(result)
}

p1 = get_plot("France", "Male")
p2 = get_plot("France", "Female")
p3 = get_plot("United Kingdom", "Male")
p4 = get_plot("United Kingdom", "Female")

grid.arrange(p1, p2, p3, p4, nrow = 2, ncol = 2)
```



Using this method, I calculated 95% confidence bands for the entire dataset. Among the 4422 data points in total, 3235 are contained in their corresponding confidence bands, which gives a coverage probability of 73.2%, not very different from that of Model 2.

One potential weakness of this model is that it did not incorporate `gdp_per_capita` as a predictor. This may cost us some explanatory power.

Model 4: An alternative nonparametric method would be a **generalized additive model**, with `suicides_100k` being the response, `country`, `year`, `sex` and `gdp_per_capita` being the predictors. One advantage of this model is that unlike Model 3, we do not need to fit the model separately for each country.

```r
m4 = gam(suicides_100k ~ country + year + sex + gdp_per_capita, data=data)
res = predict(m4, newdata=data) - data$suicides_100k
```

4 Discussion

In this final section, we compare the prediction performance of these parametric and nonparametric models.

Consider this setting: given all the data from 1985 to 2013, our task is to predict the suicide rates for each combination of `country` and `sex` for years 2014 and 2015. I choose mean absolute error (MAE) as the evaluation metric, since it is more robust to outliers than metrics such as mean squared error. Note that Model 1 cannot be tested here, because it treats `year` as a factor, and it cannot give predictions for unseen `year` levels 2014 and 2015. Nonetheless, this does not really matter, as we have seen in Section 2 that this model gives a poor fit. Another thing to keep in mind is that whenever the calculated prediction is negative, we should reset it to zero, since the suicide rate can never be negative.

So we fit Models 2, 3 and 4 on the training set, and evaluate them on the test set. The results are as follows:

```r
train = data %>% filter(year < 2014)
test = data %>% filter(year >= 2014)

m2 = lm(suicides_100k ~ sex * gdp_per_capita + (I(year^2) + year + sex) * country,
              data = train)
m2.pred = predict(m2, newdata = test)
m2.pred = ifelse(m2.pred >= 0, m2.pred, 0)
m2.mae = mean(abs(m2.pred - test$suicides_100k))

m4 = gam(suicides_100k ~ country + year + sex + gdp_per_capita, data = train)
m4.pred = predict(m4, newdata = test)
m4.pred = ifelse(m4.pred >= 0, m4.pred, 0)
m4.mae = mean(abs(m4.pred - test$suicides_100k))

model.list = vector(mode="list", length=length(country.all))
for (i in 1:length(country.all)) {
  temp = train %>% filter(country == country.all[i])
  male = temp %>% filter(sex == "Male")
  female = temp %>% filter(sex == "Female")
  model.list[[i]] = list(
    locfit(male$suicides_100k ~ male$year, alpha=c(0,4), deg=1, maxk=1000),
    locfit(female$suicides_100k ~ female$year, alpha=c(0,4), deg=1, maxk=1000))
}
test$m3.pred = 0
for (i in 1:nrow(test)) {
  ind = which(country.all == test$country[i])
  if (test$sex[i] == "Male") {
    mod = model.list[[ind]][[1]]
  } else {
    mod = model.list[[ind]][[2]]
  }
  test$m3.pred[i] = predict(mod, newdata = test[i, ])
}
test$m3.pred = ifelse(test$m3.pred >= 0, test$m3.pred, 0)
m3.mae = mean(abs(test$m3.pred - test$suicides_100k))
```

```r
paste0("Mean absolute error for model 2: ", round(m2.mae, 4))
```

```
## [1] "Mean absolute error for model 2: 2.4136"
```

```r
paste0("Mean absolute error for model 3: ", round(m3.mae, 4))
```

```
## [1] "Mean absolute error for model 3: 4.3569"
```

```r
paste0("Mean absolute error for model 4: ", round(m4.mae, 4))
```

```
## [1] "Mean absolute error for model 4: 3.4093"
```

As the result suggests, among the models we discussed in this project, **Model 2** (the parametric linear model with quadratic and interaction terms) **is the most effective in predicting future observations**. Model 4 (generalized additive model) comes the second, and Model 3 (local linear regression) comes the third. In this sense, our parametric approach outperforms nonparametric ones.

Some additional commments: (1) Model 3 actually only used `country`, `sex` and `year` as predictors, and did not include `gdp_per_capita`. This might be one reason why it is inferior to Models 2 and 4. (2) When using these models for prediction, we should not predict too many years ahead, due to extrapolation issues.