Exam 1

Henry Surjono

2024-03-13

Introduction

The Defense Advanced Research Projects Agency (DARPA) is interested in estimating the average error in drone position given a fixed distance between the drone and missile. The model created will then be used to adjust the perceived position of the drone.

When a hostile flying object enters protected airspace, DARPA launches a missile to intercept and destroy that hostile object. Objectively, the missile needs to be able to track the distance between itself and the hostile object accurately.

For testing the interceptors, drones were used to simulate hostile flying objects. Drone gps provide very accurate information on the location of the drones.Please provide the average error will be at 0.1, 1 and 10 km respectively. Provide an explanation of the oscillating behavior in the response variable.

Summary of Results

Please provide the average error will be at 0.1, 1 and 10 km respectively.

lease provide the average error will be at 0.1, 1 and 10 km respectively. The significance level we used was 95%. For each error, the confidence interval shows that if we input a distance(X), we are 95% confident the true value of Y will lie within the interval. Below shows the confidence interval for when X = 0.1, 1, and 8.

The average error for 0.1.

```
confid_interval_.1
```

```
##        2.5%       97.5%
## -0.14136225  0.01392081
```

The average error for 1.

```
confid_interval_1
```

```
##        2.5%       97.5%
## -0.14324316  0.01331923
```

The average error for 10 cannot be calculated because our data set only has a maximum value of 8 km. Since our model is based off our data set, we do not know if the oscillating pattern will continue when X gets larger. So anything pass 8km would not be feasible to calculate the average error. The below is the average error for 8 km, which is the maximum of our data.

```
confid_interval_8
```

```
##        2.5%       97.5%
## -0.14383941  0.01170303
```

Provide an explanation of the oscillating behavior in the response variable.

The oscillating behavior from the data set maybe due to how the data was captured from a camera. The drone used to capture and record the distance will show in 2 dimensions, when flying objects are in 3 dimensions. The camera used to record the data may have issues with depth perceptions, and the data show an oscillating behavior.

Methodology

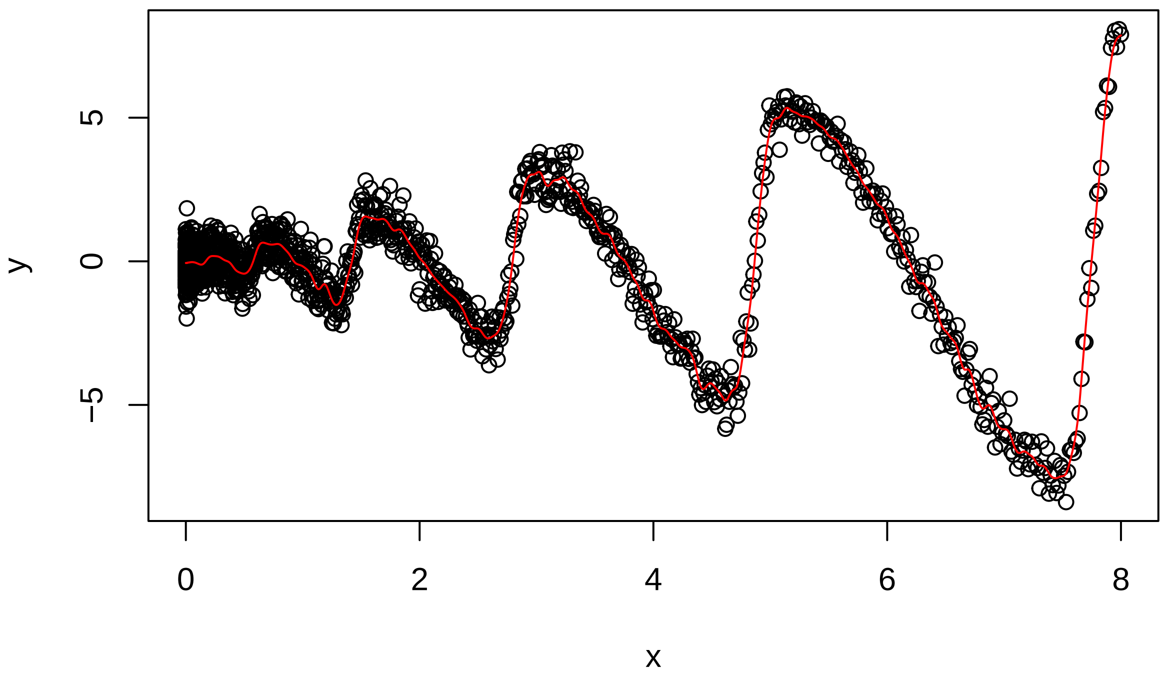
Nonparametric Model using Kernel Density Regression

When plotting the data, we observed a relationship that between distance and error. The error shows an oscillating behavior which makes linear regression unsuitable. Since the data does not follow any distribution, I opted to use kernel density regression to estimate the average error at different distances.

For the kernel density regression, a Gaussian distribution was used as my kernel density. To pick a specific bandwidth, we used the leave one out method to calculate the sum of squared residuals. After calculating the sum of squared residuals, we than calculated our optimal h for our model.

Our Model is $y = \frac{\sum(exp(-.5*((x1-x)/h)^2)*y}{\sum(exp(-.5*((x1-x)/h)^2))}$.

```
plot(x,y)
lines(x2,y2, col = "red")
```



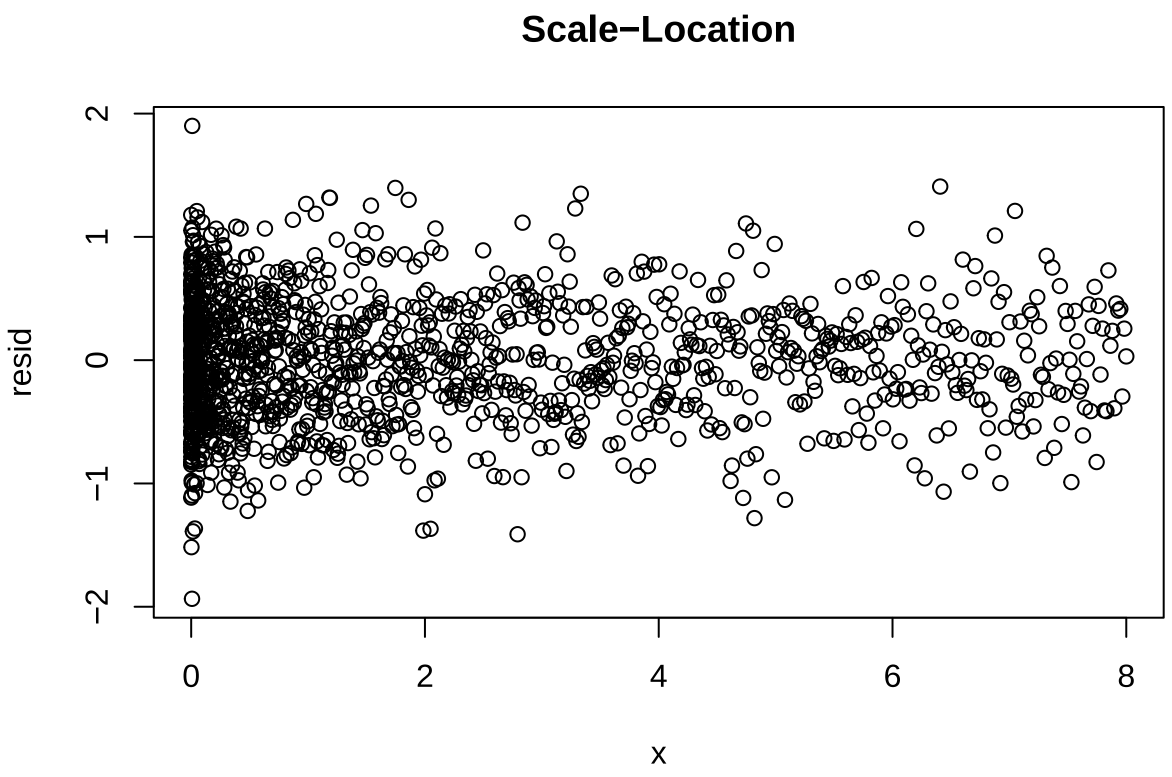
When plotting our data, and our model(red), it seems that with the optimal h, our model fits with our data.

To obtain the average error for a specific distance, we bootstrap our distance and used the model to calculate our predicted error for each distance. The residuals from our model and the original data set was added to our predicted errors. The distance for 0.1, 1, 10(or 8) was than plugged into our model and we obtained the predicted errors. This process was reiterated 10,000 times to create our confidence interval to estimate the average errors.

Checking our assumptions

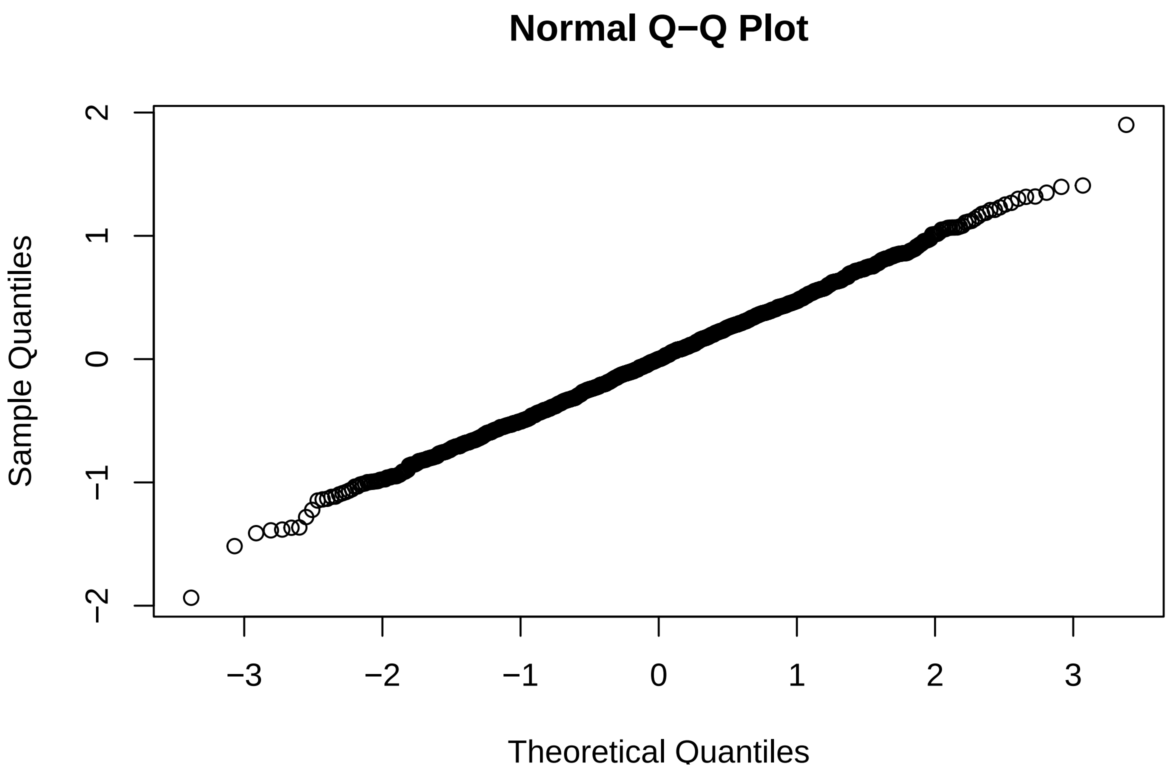
There are two assumptions that needs to be check for is that our residuals are centered and 0 and that the variance is constant. To check for these assumptions, we use the scale- location and the qqnorm plot to check for both assumptions.

```
plot(x, resid, main = "Scale-Location")
```



From the scale-location plot, we can see that the error is centered around 0 and that the variances constant. Most of our errors are evenly spread.

```
qqnorm(resid)
```



For the Q-Q plot, it shows that the residuals are normally distributed.

Coding Section

This section of code shows how I imported my dataset and what I assign each column from my dataset.

```
data <- read.csv("~/Downloads/exam1.csv")

#x is the explanatory variable
#y is the predictor variable
#h is the bandwidth, initalize it to 10
x = data$Distance..km.
y = data$Error..m.
n = dim(data)[1]
```

This code shows my cross validation function with the sum of squared residuals. To get the optimal h, we used the optim function with our cross validation function. To calculate our residuals, we got our predicted y values and added our residuals from our original data.

```
#cross validation, getting the optimal h
CV = function(h){
  SSR = 0
  for(i in 1:n){
    #compute the leave one out method for cross validation
    y1 = sum(exp(-.5*((x[i]-x[-i])/h)^2)*y[-i])/sum(exp(-.5*((x[i]-x[-i])/h)^2))

    #calculate the residuals
    resid = y[i] - y1

    #Use sum of squared residuals
```

```r
    SSR = SSR + resid^2
  }
  SSR
}

#use the optim function with the CV function to get the optimal h
h=optim(1,CV)$par

# get your residuals
x1 = x
y1 = rep(0,length(x1))
for(i in 1:length(x1)){
  y1[i] = sum(exp(-.5*((x1[i]-x)/h)^2)*y)/sum(exp(-.5*((x1[i]-x)/h)^2))
}
resid = y - y1
```

This code is for bootstrapping 10000 times from X (distance) and getting our Y(error). We than add our estimates for Y(BSy_hat) and add in our residuals from the original data set (resid). Once we computed our bootstrapped Y, it is used to calculate for when we plug in a specific x value. After repeating this process 10,000 times, we can calculate the confidence interval for a specific X. The code below specifically shows how we calculated our confidence interval when X = 1.

```r
#code for getting the confidence interval when x1 = 1
  y_hat_1 =rep(0,10000)

  for(k in 1:10000){

    #bootstrap my BSx, from my x dataset
    BSx = sample(x,n, replace = TRUE)


    #initialize the vectors for BSy_hat, input BSx1
    x1 = BSx
    y1 = rep(0,length(x1))
    for(i in 1:length(x1)){
      y1[i] = sum(exp(-.5*((x1[i]-x)/h)^2)*y)/sum(exp(-.5*((x1[i]-x)/h)^2))
    }

    BSy_hat = y1

    # to get bs_y = BSy_hat + resid

    BSy = BSy_hat + resid

    x1 = .1
    y1 = rep(0,1)
    for(i in 1:1){
      y1[i] = sum(exp(-.5*((x1[i]-BSx)/h)^2)*BSy)/sum(exp(-.5*((x1[i]-BSx)/h)^2))
      y_hat_1[k] <- y1[i]
    }
    y_hat_1
  }

  #compute confidence interval
```

```r
confid_interval_1 <- quantile(y_hat_1, c(0.025, .975))
```