Final Project- Data Analysis of the Stormwater dataset

2022-12-01

Shaista Fathima Basheeruddin

ISE 201 MATHEMATICS AND DATA SCIENCE

SAN JOSE STATE UNIVERSITY

Shilpa Gupta

**Abstract**

The scope of this paper revolves around collected data from a stormwater pipe during the year 2018-2019 in San Francisco, California. Stormwater analysis is an important category of research for Hydraulic engineers. This report constructs an analysis of a stormwater dataset using R studio. The dataset hosts variables such as pipe length, slope, upstream elevation, downstream elevation, and diameter to understand the interdependency and effect of each variable on the other. This analysis uses Hazen William’s pipe equation to determine additional parameters such as hydraulic radius, area, and water velocities. These resultant variables to the original storm-water dataset will prove effective in developing correct interdependencies between variables using EDA, linear regression, and PCA. The methodology requires using R-studio to perform three types of data analysis methods: Exploratory data analysis, Principal Component Analysis (PCA), and Linear regression. Results will establish the relationship between the velocity of water and pipe slope which is of critical importance and their magnitudes which are required for the smooth operation of a stormwater pipe.

**Objective**

The main objective of this paper is to build an understanding of operating R-studio to summarize EDA, PCA, and linear regression on a complex stormwater dataset using graphical illustrations and mathematical computations to depict the behavior of variables.

**Introduction**

This paper contributes significantly to researchers and hydraulic pipe design engineers. They are required to design stormwater pipes based on critical parameters which apply the principles of Hazen Williams pipe equation. It is an empirical relationship that relates the flow of water in a pipe with the physical properties of the pipe. This equation doesn’t account for the temperature or viscosity of water. The research question this paper aims to comprehend is the effect of pipe parameters (diameter, slope, length) on the velocity of water flowing through the stormwater pipe. Collecting past data from stormwater pipes specific to location IDs will not only give the researcher information about the pipe parameters that should be allocated in each flood region but also comments on their durability and strengths.

Because of the time constraints pertaining to this research, a few assumptions were made. The computations are conducted on Concrete pipes and the head loss value uses the pipe slope. The hydraulic radius, area, and velocity we the computed parameters. The equation used is V(velocity)= k CR^0.5 S^0.5. K is the conversion factor i.e. 0.859 in SI units, C is the roughness coefficient of concrete that is taken as 100 and S is the slope of the energy line, which is assumed as pipe slope in this section (Wikipedia, 2022)

The intention of selecting this data was to understand real-world applications of hydraulic data from Storm-water pipes and to construct a suitable portfolio portraying expertise in R Markdown using analytical techniques to illustrate the correlation between variables.

**Data Collection**

This data set originally contains 26 columns and 355 rows with some missing values. To work with this data set, columns such as collecting and receiving and pipe ID’s were deleted and the unavailable values were removed. It is a combination of categorical, numerical, and discreet variables. The data is collected from water runoff over impermeable surfaces like roofs, concrete pavements, and roads. The information and data set was collected from the San Jose government website \*\*<https://data.sanjoseca.gov/dataset/storm-gravity-mains1>\*\*.

***A short note describing the variables:***

Water runoff in this data set is categorized into Upstream and Downstream elevations. This data set contains numerical variables such as upstream elevation, downstream elevation, pipe diameter, lengths, and slopes. The categorical variables include pipe material, installation year, and outfall ID. An outfall is defined as a point where sewer systems discharge into the territories of “Waters in the United States” whether the discharges are from ditches, swales, and other points of concentrated flow. “Waters of the United States” refers to surface waters and discharges to aquifers (groundwater) that are not termed outfalls. (Sanjoseca.gov, 2022)

***Questions the data set answers:***

1. The relation between the slope of the pipe with the upstream elevation of water will show graphical illustrations if they are positively or negatively correlated.
2. The mean range of diameters corresponds to certain pipe slopes using boxplots.
3. The frequencies of upstream and downstream elevations using a barplot, which represents a normal distribution
4. The variation in the slope of the pipe with increasing lengths and the mean length of the pipe pertaining to a specific slope parameter.

*Note: The proposed questions were tweaked in order to account for time constraints.*

***Data Analysis techniques carried out using R studio***

* Exploratory Data Analysis (EDA),
* (PCA) Principal Component Analysis i.e. matrix factorizations of data.
* Linear regression classification model, and confidence interval.

### Loading the required packages

library(tinytex)  
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.0 ✔ purrr 0.3.4   
## ✔ tibble 3.1.8 ✔ dplyr 1.0.10  
## ✔ tidyr 1.2.1 ✔ stringr 1.4.1   
## ✔ readr 2.1.2 ✔ forcats 0.5.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(tidyr)  
library(stats)  
library(ggplot2)  
library("dplyr")  
library(tidyverse)  
library(ggpubr)  
library(RcppEigen)

### Viewing the data Table and cleaning the dataset

library(tidyverse)  
library(readxl)  
Stormwater <- read\_excel("/Users/shaistafathima/Desktop/ISE 201/StormwaterData.xlsx")  
View(Stormwater)

stormdata <- Stormwater[, !names(Stormwater) %in% c("Integer ID", "Facility Identifier", "Integer Id", "Outfall ID", "Outfall Group ID", "From Asset ID", "From Asset Type", "To Asset ID","Pipe Type", "To Asset Type", "Year Lined", "Liner type", "Owned By", "Source Year", "Plan Created", "Plan Modified", "Last Update Date", "Notes", "Plan Discrepancy/Rehab", "Liner Type")]  
  
head(stormdata)

## # A tibble: 6 × 8  
## `Street ID` `Upstream Elevation` Downs…¹ Diame…² Length Slope Mater…³ Insta…⁴  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 21659 65.0 56.0 "24\"" 310 0.029 Reinfo… NA  
## 2 63377 NA NA <NA> NA NA Unknown 2003  
## 3 63491 NA NA <NA> NA NA Unknown 2003  
## 4 22338 78.9 77.8 "15\"" 195 0.006 Reinfo… 2006  
## 5 22801 102. 102. "24\"" 164 0.002 Reinfo… 2005  
## 6 22318 68 64 "12\"" 44 0.091 Reinfo… 1999  
## # … with abbreviated variable names ¹​`Downstream Elevation`, ²​Diameter,  
## # ³​Material, ⁴​`Install Year`

view(stormdata)

### Identifying if the dataset has missing variables

stormdata[!complete.cases(stormdata),]

## # A tibble: 97 × 8  
## `Street ID` Upstream Elevatio…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 21659 65.0 56.0 "24\"" 310 0.029 Reinfo… NA  
## 2 63377 NA NA <NA> NA NA Unknown 2003  
## 3 63491 NA NA <NA> NA NA Unknown 2003  
## 4 71705 51.2 50.5 "33\"" NA NA Reinfo… 1999  
## 5 26190 50.8 NA "27\"" 260 0.004 Reinfo… 2010  
## 6 20636 NA 52.8 "24\"" 7 NA Unknown 2010  
## 7 72704 NA 39.0 "33\"" 14 0.002 Reinfo… 2008  
## 8 21696 58.7 NA "21\"" 192 NA C 2007  
## 9 9991 199. 195. "36\"" NA 0.025 Reinfo… 2008  
## 10 9991 195. 190. "36\"" NA 0.025 Reinfo… 2008  
## # … with 87 more rows, and abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`

stormdata

## # A tibble: 352 × 8  
## `Street ID` Upstream Elevatio…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 21659 65.0 56.0 "24\"" 310 0.029 Reinfo… NA  
## 2 63377 NA NA <NA> NA NA Unknown 2003  
## 3 63491 NA NA <NA> NA NA Unknown 2003  
## 4 22338 78.9 77.8 "15\"" 195 0.006 Reinfo… 2006  
## 5 22801 102. 102. "24\"" 164 0.002 Reinfo… 2005  
## 6 22318 68 64 "12\"" 44 0.091 Reinfo… 1999  
## 7 71705 51.2 50.5 "33\"" NA NA Reinfo… 1999  
## 8 22106 46.6 45.6 "24\"" 347 0.003 Reinfo… 1999  
## 9 83065 46.9 46.6 "24\"" 80 0.003 Reinfo… 1999  
## 10 21680 43.8 43.4 "27\"" 43 0.009 Reinfo… 1999  
## # … with 342 more rows, and abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`

### Identifying the head and tail of the stormwater dataset

stormdataclean<-stormdata  
head(stormdataclean)

## # A tibble: 6 × 8  
## `Street ID` `Upstream Elevation` Downs…¹ Diame…² Length Slope Mater…³ Insta…⁴  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 21659 65.0 56.0 "24\"" 310 0.029 Reinfo… NA  
## 2 63377 NA NA <NA> NA NA Unknown 2003  
## 3 63491 NA NA <NA> NA NA Unknown 2003  
## 4 22338 78.9 77.8 "15\"" 195 0.006 Reinfo… 2006  
## 5 22801 102. 102. "24\"" 164 0.002 Reinfo… 2005  
## 6 22318 68 64 "12\"" 44 0.091 Reinfo… 1999  
## # … with abbreviated variable names ¹​`Downstream Elevation`, ²​Diameter,  
## # ³​Material, ⁴​`Install Year`

tail(stormdataclean)

## # A tibble: 6 × 8  
## `Street ID` `Upstream Elevation` Downst…¹ Diame…² Length Slope Mater…³ Insta…⁴  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 15820 112. 112. "18\"" 40 3e-3 High D… 2007  
## 2 15820 NA 114. "18\"" 9 NA High D… 2007  
## 3 15820 112. 112. <NA> 21 4e-3 High D… 2007  
## 4 106098 82.0 81.7 "15\"" 82 4e-3 Reinfo… 2007  
## 5 NA NA NA <NA> NA NA <NA> NA  
## 6 NA NA NA <NA> NA NA <NA> NA  
## # … with abbreviated variable names ¹​`Downstream Elevation`, ²​Diameter,  
## # ³​Material, ⁴​`Install Year`

str(stormdataclean)

## tibble [352 × 8] (S3: tbl\_df/tbl/data.frame)  
## $ Street ID : num [1:352] 21659 63377 63491 22338 22801 ...  
## $ Upstream Elevation : num [1:352] 65 NA NA 78.9 101.8 ...  
## $ Downstream Elevation: num [1:352] 56 NA NA 77.8 101.5 ...  
## $ Diameter : chr [1:352] "24\"" NA NA "15\"" ...  
## $ Length : num [1:352] 310 NA NA 195 164 44 NA 347 80 43 ...  
## $ Slope : num [1:352] 0.029 NA NA 0.006 0.002 0.091 NA 0.003 0.003 0.009 ...  
## $ Material : chr [1:352] "Reinforced Concrete" "Unknown" "Unknown" "Reinforced Concrete" ...  
## $ Install Year : num [1:352] NA 2003 2003 2006 2005 ...

### Calculating the number of rows and columns in the Data set.

nrow(stormdataclean)

## [1] 352

ncol(stormdataclean)

## [1] 8

There are 355 rows and 8 columns in the dataset

### Dropping Non-Available Values.

stormdata<-drop\_na(stormdataclean)  
stormdata

## # A tibble: 255 × 8  
## `Street ID` `Upstream Elevation` Downs…¹ Diame…² Length Slope Mater…³ Insta…⁴  
## <dbl> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl>  
## 1 22338 78.9 77.8 "15\"" 195 0.006 Reinfo… 2006  
## 2 22801 102. 102. "24\"" 164 0.002 Reinfo… 2005  
## 3 22318 68 64 "12\"" 44 0.091 Reinfo… 1999  
## 4 22106 46.6 45.6 "24\"" 347 0.003 Reinfo… 1999  
## 5 83065 46.9 46.6 "24\"" 80 0.003 Reinfo… 1999  
## 6 21680 43.8 43.4 "27\"" 43 0.009 Reinfo… 1999  
## 7 22639 85.3 85.2 "12\"" 32 0.005 Reinfo… 1999  
## 8 22639 87.1 85.3 "12\"" 382 0.005 Reinfo… 1999  
## 9 22624 82.5 80.0 "15\"" 550 0.005 Reinfo… 1999  
## 10 11812 240. 238. "33\"" 200 0.012 Reinfo… 1999  
## # … with 245 more rows, and abbreviated variable names ¹​`Downstream Elevation`,  
## # ²​Diameter, ³​Material, ⁴​`Install Year`

### Determining the type of variable in each column of the dataset is a character, numeric, or factor and finding the summary statistics of each variable. We can observe that the diameter column has a character string. We intend to convert it into a categorical column to achieve the objective of identifying the mean diameters of pipes corresponding to certain pipe slopes.

summary(stormdata$Diameter)

## Length Class Mode   
## 255 character character

summary(stormdata$Slope)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.08900 0.00300 0.00500 0.01192 0.01000 0.22900

summary(stormdata$Length)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 9.0 85.5 186.0 219.5 310.5 890.0

summary(stormdata$"Upstream Elevation")

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -6.31 66.29 110.40 117.57 133.12 542.69

summary(stormdata$"Downstream Elevation")

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -7.03 64.67 110.01 115.42 131.82 524.06

summary(stormdata$Material)

## Length Class Mode   
## 255 character character

summary(stormdata$"Install Year")

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1961 2002 2007 2005 2008 2015

stormdata$Diameter

## [1] "15\"" "24\"" "12\"" "24\"" "24\"" "27\"" "12\"" "12\"" "15\"" "33\""  
## [11] "15\"" "27\"" "36\"" "36\"" "36\"" "36\"" "18\"" "33\"" "36\"" "36\""  
## [21] "18\"" "12\"" "15\"" "15\"" "21\"" "18\"" "15\"" "15\"" "24\"" "24\""  
## [31] "10\"" "8\"" "15\"" "15\"" "24\"" "24\"" "18\"" "15\"" "15\"" "18\""  
## [41] "18\"" "24\"" "24\"" "12\"" "18\"" "18\"" "18\"" "18\"" "12\"" "8\""

names(stormdata)

## [1] "Street ID" "Upstream Elevation" "Downstream Elevation"  
## [4] "Diameter" "Length" "Slope"   
## [7] "Material" "Install Year"

gfg<- stormdata$Diameter  
answer<-strsplit(gfg,split = "\"")  
num <- as.numeric(unlist(answer))  
stormdata$Diameter <- num  
stormdata$Diameter

## [1] 15 24 12 24 24 27 12 12 15 33 15 27 36 36 36 36 18 33 36 36 18 12 15 15 21  
## [26] 18 15 15 24 24 10 8 15 15 24 24 18 15 15 18 18 24 24 12 18 18 18 18 12 8  
## [51] 12 12 42 42 15 10 10 10 15 15 15 18 15 18 15 15 15 24 27 15 15 18 18 15 15  
## [76] 18 60 21 12 12 54 54 72 72 12 36 15 27 30 15 12 18 12 12 15 21 10 21 21 24  
## [101] 15 48 48 48 48 10 24 30 24 24 24 60 60 48 48 27 18 66 27 18 27 27 27 15 12  
## [126] 10 10 8 16 18 12 12 15 18 15 12 15 18 66 66 10 15 21 10 12 12 12 15 15 36  
## [151] 16 14 15 15 15 12 12 12 12 18 18 18 18 18 18 66 66 60 12 12 12 12 66 66 66  
## [176] 60 14 24 27 30 24 30 15 12 24 21 60 18 10 30 30 30 30 30 30 30 18 27 27 27  
## [201] 24 24 24 24 24 24 24 12 12 12 12 12 12 6 18 10 33 12 15 60 60 18 15 10 8  
## [226] 18 36 18 18 60 60 18 24 10 14 16 12 12 18 10 10 18 12 15 15 15 15 15 15 15  
## [251] 18 12 18 18 15

stormdata

## # A tibble: 255 × 8  
## `Street ID` `Upstream Elevation` Downs…¹ Diame…² Length Slope Mater…³ Insta…⁴  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999  
## 7 22639 85.3 85.2 12 32 0.005 Reinfo… 1999  
## 8 22639 87.1 85.3 12 382 0.005 Reinfo… 1999  
## 9 22624 82.5 80.0 15 550 0.005 Reinfo… 1999  
## 10 11812 240. 238. 33 200 0.012 Reinfo… 1999  
## # … with 245 more rows, and abbreviated variable names ¹​`Downstream Elevation`,  
## # ²​Diameter, ³​Material, ⁴​`Install Year`

head(stormdata$Diameter)

## [1] 15 24 12 24 24 27

### We used the strsplit command to split the character string. And coneverted it into a numeric form

### Creating ranges for the Diameter category column

Diameter\_cat <- stormdata$Diameter  
Diameter\_cat[stormdata$Diameter <= 10] <- "0-10"   
Diameter\_cat[stormdata$Diameter > 10 & stormdata$Diameter <= 20] <- "10-20"   
Diameter\_cat[stormdata$Diameter > 20 & stormdata$Diameter <= 30] <- "20-30"  
Diameter\_cat[stormdata$Diameter > 30 & stormdata$Diameter <= 40] <- "30-40"  
Diameter\_cat[stormdata$Diameter > 40 & stormdata$Diameter <= 50] <- "40-50"  
Diameter\_cat[stormdata$Diameter > 50 & stormdata$Diameter <= 60] <- "50-60"  
Diameter\_cat[stormdata$Diameter > 60 & stormdata$Diameter <= 70] <- "60-70"  
Diameter\_cat[stormdata$Diameter > 70] <- "70+"  
Diameter\_cat<- as.factor(Diameter\_cat)   
print(Diameter\_cat)

## [1] 10-20 20-30 10-20 20-30 20-30 20-30 10-20 10-20 10-20 30-40 10-20 20-30  
## [13] 30-40 30-40 30-40 30-40 10-20 30-40 30-40 30-40 10-20 10-20 10-20 10-20  
## [25] 20-30 10-20 10-20 10-20 20-30 20-30 0-10 0-10 10-20 10-20 20-30 20-30  
## [37] 10-20 10-20 10-20 10-20 10-20 20-30 20-30 10-20 10-20 10-20 10-20 10-20  
## [49] 10-20 0-10 10-20 10-20 40-50 40-50 10-20 0-10 0-10 0-10 10-20 10-20  
## [61] 10-20 10-20 10-20 10-20 10-20 10-20 10-20 20-30 20-30 10-20 10-20 10-20  
## [73] 10-20 10-20 10-20 10-20 50-60 20-30 10-20 10-20 50-60 50-60 70+ 70+   
## [85] 10-20 30-40 10-20 20-30 20-30 10-20 10-20 10-20 10-20 10-20 10-20 20-30  
## [97] 0-10 20-30 20-30 20-30 10-20 40-50 40-50 40-50 40-50 0-10 20-30 20-30  
## Levels: 0-10 10-20 20-30 30-40 40-50 50-60 60-70 70+

stormdata[31, c("Diameter")]

## # A tibble: 1 × 1  
## Diameter  
## <dbl>  
## 1 10

### 12. Adding a new column or Diameter category that is “Diameter\_cat” to the stormwater data frame

stormdata$Diameter\_cat <- Diameter\_cat  
head(stormdata)

## # A tibble: 6 × 9  
## `Street ID` Upstream El…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵ Diame…⁶  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct>   
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20   
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30   
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20   
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30   
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30   
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30   
## # … with abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`,  
## # ⁶​Diameter\_cat

**Before plotting this data set we would be interested to estimate the velocity and discharge through each pipe. This data was estimated with the current data provided to us. Calculating the velocity requires a few parameters such as hydraulic radius which is the area of the pipe/ wetted perimeter of pipe.**

**We have to keep this equation in mind Velocity= kCR^0.5\*S^0.5**

### 13. Estimating the area of the pipe; Area= pi\*d^2/4

names(stormdata)

## [1] "Street ID" "Upstream Elevation" "Downstream Elevation"  
## [4] "Diameter" "Length" "Slope"   
## [7] "Material" "Install Year" "Diameter\_cat"

diam = stormdata$Diameter  
stormdata['Area'] = diam^2\*pi /4  
head(stormdata)

## # A tibble: 6 × 10  
## `Street ID` Upstr…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵ Diame…⁶ Area  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20 177.  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30 452.  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20 113.  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30 452.  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30 452.  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30 573.  
## # … with abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`,  
## # ⁶​Diameter\_cat

### 14. Computing Wetted Perimeter of pipe = D/8 and adding it to the original stormwater dataset

diam1 = stormdata$Diameter  
stormdata['Wetted Perimeter'] = diam/8  
head(stormdata)

## # A tibble: 6 × 11  
## `Street ID` Upstr…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵ Diame…⁶ Area  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20 177.  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30 452.  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20 113.  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30 452.  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30 452.  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30 573.  
## # … with 1 more variable: `Wetted Perimeter` <dbl>, and abbreviated variable  
## # names ¹​`Upstream Elevation`, ²​`Downstream Elevation`, ³​Diameter, ⁴​Material,  
## # ⁵​`Install Year`, ⁶​Diameter\_cat

### 15. Calculating the Hydraulic radius R= Area/Wetted perimeter of the pipe

H\_Radius = stormdata$Area/stormdata$`Wetted Perimeter`  
stormdata['Hydraulic Radius'] = H\_Radius  
head(stormdata)

## # A tibble: 6 × 12  
## `Street ID` Upstr…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵ Diame…⁶ Area  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20 177.  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30 452.  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20 113.  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30 452.  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30 452.  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30 573.  
## # … with 2 more variables: `Wetted Perimeter` <dbl>, `Hydraulic Radius` <dbl>,  
## # and abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`,  
## # ⁶​Diameter\_cat

### 16. Estimating velocity through the pipe using Hazen Williams pipe equation where V= K x CR^0.63 x S^0.54, where K is the conversion factor, Assuming, C is the roughness coefficient of concrete = 100, R is the hydraulic radius and S is the slope of the pipe. Where K= 0.849

A <- stormdata$`Hydraulic Radius`  
head(A)

## [1] 94.24778 150.79645 75.39822 150.79645 150.79645 169.64600

B <- stormdata$Slope  
head(B)

## [1] 0.006 0.002 0.091 0.003 0.003 0.009

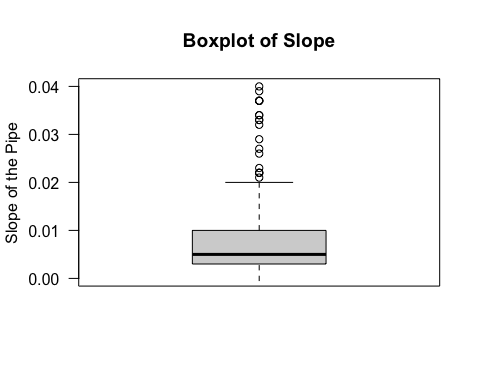
C<-A^0.63  
D<-B^0.54  
velocity=0.849\*100\*C\*D  
stormdata['Velocity'] = velocity  
head(stormdata)

## # A tibble: 6 × 13  
## `Street ID` Upstr…¹ Downs…² Diame…³ Length Slope Mater…⁴ Insta…⁵ Diame…⁶ Area  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20 177.  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30 452.  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20 113.  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30 452.  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30 452.  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30 573.  
## # … with 3 more variables: `Wetted Perimeter` <dbl>, `Hydraulic Radius` <dbl>,  
## # Velocity <dbl>, and abbreviated variable names ¹​`Upstream Elevation`,  
## # ²​`Downstream Elevation`, ³​Diameter, ⁴​Material, ⁵​`Install Year`,  
## # ⁶​Diameter\_cat

#### **17. Answering the important questions in this dataset**

#### **a) Plotting a boxplot on the slope of the pipe to represent outliers**

boxplot(stormdata$Slope, main="Boxplot of Slope", ylab="Slope of the Pipe", ylim=c(0,0.04), las=1, color= "red")



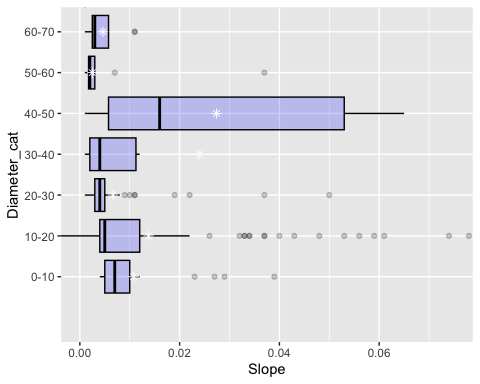
quantile(stormdata$Slope, probs=c(0,0.25,0.5,0.75,1), na.rm=TRUE)

## 0% 25% 50% 75% 100%   
## -0.089 0.003 0.005 0.010 0.229

To plot the relation between diameter and boxplot we are grouping the diameters by range into a new column.

### b) Plotting a box-plot between a numeric and categorical variable showing how slope of the pipe varies across different diameters.

ggplot(data = stormdata, aes(Slope, Diameter\_cat))+   
 geom\_boxplot(width=0.8, color= "Black", fill= "blue", alpha=0.2)+  
 coord\_cartesian(xlim = c(0,0.075), ylim = c(0,7))+  
 theme(legend.position="none")+  
 #scale\_x\_continuous(n.breaks=0.1)+  
 stat\_summary(fun = "mean", geom = "point", shape = 8, size = 2, color = "white")



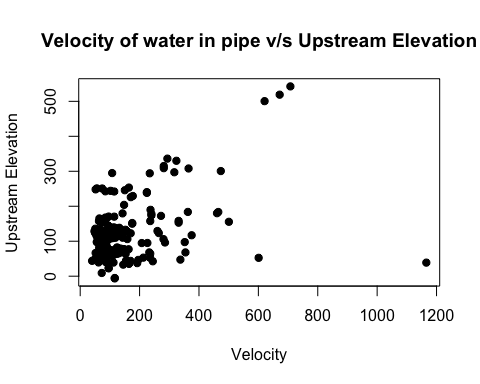
#pglot + coord\_cartesian(ylim = c(0,0.2), xlim = c(0,10))

**The boxplot above gives the reader information that the largest range of pipe diameters common to this data set fall in the range of 40-50. It The range 10-20 has the highest number of outliers compared to other ranges.**

View(stormdata)

### c) A scatter plot between a numeric and categorical variable showing how slope of the pipe varying across different diameters.

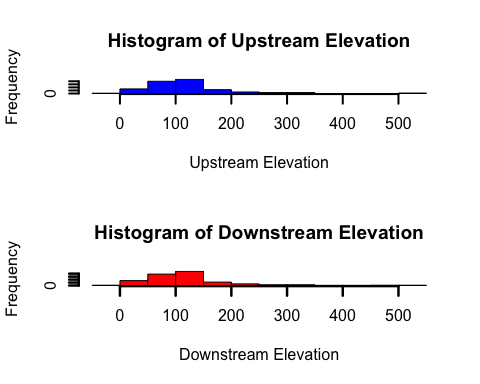
plot(stormdata$Velocity, stormdata$'Upstream Elevation',   
 col = "Black", pch = 19,  
 main = "Velocity of water in pipe v/s Upstream Elevation",   
 xlab = "Velocity", ylab = "Upstream Elevation", lwd = 1)



**This scatter plot conveys that the Velocity of water is directly proportional to the Upstream flow of water in the pipe and shows a strong positive relationship. It means that if the velocity of water in the pipe is high, the elevation of water upstream is also high. The data is concentrated in the bottom left which means that. At the data set has maximum water velocities to be between 10-90 miles/hour. The upstream elevation heads of water are also below 200 mm in the pipe.**

### c) Plotting an enhanced histogram showing the frequency of upstream and downstream elevation

par(mfrow = c(2,1))  
  
hist(stormdata$'Upstream Elevation', xlab = "Upstream Elevation", lwd = 2,   
main = "Histogram of Upstream Elevation", col= 'Blue')  
  
hist(stormdata$'Downstream Elevation', col = 'Red',   
 xlab = "Downstream Elevation", lwd = 2.5, main = "Histogram of Downstream Elevation")



### 80% of the upstream and downstream elevation mean lies between 50-150, it represents that of a pareto chart where maximum values contribute to 80% of the data. Upstream and downstream elevations are maximum at values between 100 to 150. It means that, for a mean pipe length of 200 (represented by the density plot in the latter of this report) the corresponding appropriate water Upstream and downtream elevations are between 50-150mm. Form the density plot further in the report it attributes to a mean slope of 0.002 with diamteres ranging from 40-50 (from the bar plot). We can also ifer than an upstream mean elevation from 50-150mm correcponds to a water velocity in the range of 80-100 miles/hour. To match to the flux of velocity most pipe materials are chosen as concrete since it is free from corossion and can withstand the forces and densities of water at the corresponding pipe diameters. If the corresponding parameters are interpretted incorrectly the stormwater pipe can break and pose a major threat of escalating stormwater flood velocity instead of counteracting it. We can infer from the first plot (scatter plot) for high flood velocities the upstream flow is high. A higher upstream flow corresponds to a larger pipe diameter. If the diamter is insufficient for the water to pass freely the pipe will break.

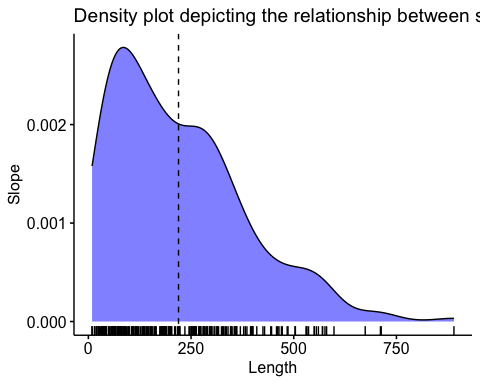
### d) Adding a density plot to show the relatinship between length and slope of the pipe which satisfies obective number 5. Changing pipe slopes across different lengths of the pipe.

stormdata %>%  
 ggdensity(x = "Length",  
 add = "mean",   
 rug = TRUE,  
 color = "Black",   
 fill = "Blue",  
 title = "Density plot depicting the relationship between shope and length of pipe",  
 xlab = "Length",  
 ylab = "Slope",  
 palette = c("#2e00fa", "#a000bc", "#ca0086", "#e40058"))

## Warning: `geom\_vline()`: Ignoring `mapping` because `xintercept` was provided.

## Warning: `geom\_vline()`: Ignoring `data` because `xintercept` was provided.

## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.  
## ℹ Please use `after\_stat(density)` instead.  
## ℹ The deprecated feature was likely used in the ggpubr package.  
## Please report the issue at <]8;;https://github.com/kassambara/ggpubr/issueshttps://github.com/kassambara/ggpubr/issues]8;;>.



### This density plot indicated that the mean of pipe length is close to 200ft. Therefore, this gives us information that mean pipe length of 200 ft corresponds to a pipe slope of 0.002. It shows a negative relationship between slope and pipe length. It is practical that pipes of larger lengths will have a smaller slope or inclination. This is true theoretically, and practically. A pipe of larger length must maintain a lower degree of slope, to smoothen the upstream and downstream elevations of water and decrease its velocity. This will allow the stormwater from exiting a pipe with a minimized velocity. A higher slope can aggregate the upstream elevations and water velocity, imposing a pressure and force above the capacity of pipe limit can rupture the pipe.

**Performing a Regression analysis on the dataset**

#### **1. Recording Observations: The scatterplot shows a linear relationship between the x and y variables, where X is the independent variable and y is the dependent variable.**

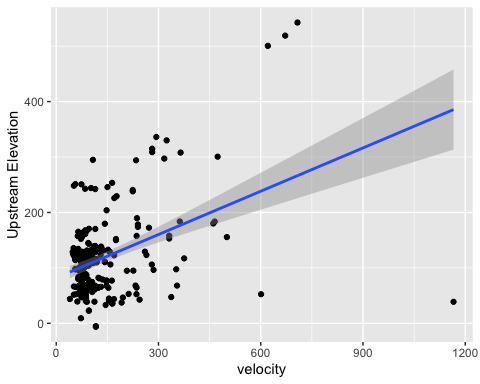
**b) Fitting a simple Linear Regression Model**

library(tidyverse)  
library(ggplot2)  
ggplot(stormdata, aes(velocity,`Upstream Elevation`))+ geom\_point()+ geom\_smooth(method= 'lm')

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 5 rows containing non-finite values (`stat\_smooth()`).

## Warning: Removed 5 rows containing missing values (`geom\_point()`).



The linear regression line depicts that the x variable is directly proportional to the y variable, that is an increase in the x value will cause the Y variable to increase.

Testing the significance of regression using α = 0.05 and 0.01 and recording observations.

**Step 1.** Fit regression Model for which we are required to check if the dependent variable “Upstream Elevation” follows noraml distribution.

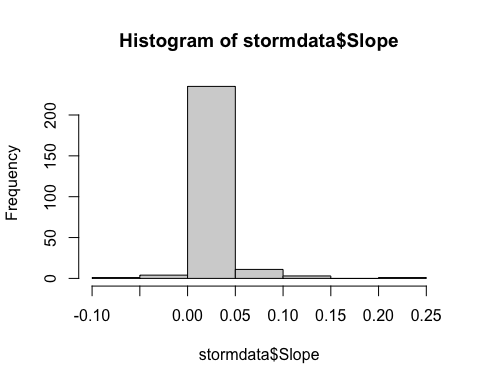
head(stormdata, 10)

## # A tibble: 10 × 13  
## Street I…¹ Upstr…² Downs…³ Diame…⁴ Length Slope Mater…⁵ Insta…⁶ Diame…⁷ Area  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <fct> <dbl>  
## 1 22338 78.9 77.8 15 195 0.006 Reinfo… 2006 10-20 177.  
## 2 22801 102. 102. 24 164 0.002 Reinfo… 2005 20-30 452.  
## 3 22318 68 64 12 44 0.091 Reinfo… 1999 10-20 113.  
## 4 22106 46.6 45.6 24 347 0.003 Reinfo… 1999 20-30 452.  
## 5 83065 46.9 46.6 24 80 0.003 Reinfo… 1999 20-30 452.  
## 6 21680 43.8 43.4 27 43 0.009 Reinfo… 1999 20-30 573.  
## 7 22639 85.3 85.2 12 32 0.005 Reinfo… 1999 10-20 113.  
## 8 22639 87.1 85.3 12 382 0.005 Reinfo… 1999 10-20 113.  
## 9 22624 82.5 80.0 15 550 0.005 Reinfo… 1999 10-20 177.  
## 10 11812 240. 238. 33 200 0.012 Reinfo… 1999 30-40 855.  
## # … with 3 more variables: `Wetted Perimeter` <dbl>, `Hydraulic Radius` <dbl>,  
## # Velocity <dbl>, and abbreviated variable names ¹​`Street ID`,  
## # ²​`Upstream Elevation`, ³​`Downstream Elevation`, ⁴​Diameter, ⁵​Material,  
## # ⁶​`Install Year`, ⁷​Diameter\_cat

storm <-data.frame(stormdata$Slope, stormdata$Velocity)   
head(storm)

## stormdata.Slope stormdata.Velocity  
## 1 0.006 93.95073  
## 2 0.002 69.79917  
## 3 0.091 354.42761  
## 4 0.003 86.88395  
## 5 0.003 86.88395  
## 6 0.009 169.36011

hist(stormdata$Slope)



model<-storm  
model<- lm (stormdata$Slope ~ velocity, data= storm)  
summary(model)

##   
## Call:  
## lm(formula = stormdata$Slope ~ velocity, data = storm)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.052295 -0.002629 0.000802 0.003112 0.060331   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.038e-02 1.117e-03 -9.293 <2e-16 \*\*\*  
## velocity 1.659e-04 5.968e-06 27.791 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.01172 on 248 degrees of freedom  
##(5 observations deleted due to missingness)  
## Multiple R-squared: 0.7569, Adjusted R-squared: 0.756   
## F-statistic: 772.4 on 1 and 248 DF, p-value: < 2.2e-16

Residual is defined as the difference between the observed value and the predicted value. Since the model observations have a small residual it means, the model is a good fit. For every unit increase in velocity the slope increases by 0.0001659. The R square indicates 0.7569 (75.69%) indicates the extent of y explained by x which means that there is 75.69% variability in the data. A value of R>0.66 means x and y have a strong inter-dependent relatiionship. For the slope to be 0.00384 the probability is extremely small which is indicated by the p-value= 8.81e-15. and since it is less than 0.05 we reject the null hypothesis that beta= 0. Hence there is a significant relationship between the variables in the linear regression model of dataset.

Hence the slope of pipe for mean velocity of 100miles per hour is calculate below

coef(model)

##(Intercept) velocity   
## -0.0103822485 0.0001658706

slopeofpipe<- -0.001038+0.0001659\*(100)  
slopeofpipe

## [1] 0.015552

### As we can see the pipe slope from the grpah is in the mean range of 0.00 to 0.05, A positive slope refers to an inclinde of positive elevation of pipe from the horizontal, and a negative slope represents a downward shift from the normal.

**2. Estimating the correlation coefficient and computing a 95% confidence interval for the correlation coefficient.**

head(storm, 5)

## stormdata.Slope stormdata.Velocity  
## 1 0.006 93.95073  
## 2 0.002 69.79917  
## 3 0.091 354.42761  
## 4 0.003 86.88395  
## 5 0.003 86.88395

as.numeric(storm$Velocity)

## numeric(0)

class(velocity)

## [1] "numeric"

class(stormdata$Slope)

## [1] "numeric"

cor(storm[1:255,])

## stormdata.Slope stormdata.Velocity  
## stormdata.Slope 1 NA  
## stormdata.Velocity NA 1

A positive correlation value indicates that the y values increase as a function of x. Correlation coefficient of 1 implies that all the data points fall exactly on a straight line. In this case there is some scatter of data, so the correlation coefficient is 1 and is considered strong.

**Estimating the regression values of a 95% confidence interval using l.model (linear regression model)**

l.model <- lm(velocity ~1, storm)  
l.model

##   
## Call:  
## lm(formula = velocity ~ 1, data = storm)  
##   
## Coefficients:  
## (Intercept)   
## 140.1

confint(l.model, level=0.95)

## 2.5 % 97.5 %  
## (Intercept) 124.5781 155.571

### e) Plotting the residuals versus y^i and comment on the underlying regression assumptions. Specifically, does it seem that the equality of variance assumption is satisfied.

Step 1: Calculating the standard residuals

standard\_res <- rstandard(model)  
head(standard\_res)

## 1 2 3 4 5 6   
## 0.06831472 0.06885607 3.66460634 -0.08805437 -0.08805437 -0.74493779

tail(standard\_res)

## 250 251 252 253 254 255   
## 4.15799972 -0.06707908 1.66901082 -0.03928445 0.11634975 0.15940728

#### **Step 2: Adding the standardized residuals to the original data frame, Here y represents the standard residual**

final\_data<- merge(storm, standard\_res)  
head(final\_data)

## stormdata.Slope stormdata.Velocity y  
## 1 0.006 93.95073 0.06831472  
## 2 0.002 69.79917 0.06831472  
## 3 0.091 354.42761 0.06831472  
## 4 0.003 86.88395 0.06831472  
## 5 0.003 86.88395 0.06831472  
## 6 0.009 169.36011 0.06831472

We can then sort each observation in descending order according to its standardized residual to get an idea of which observations are closest to being outliers.

**Step 3- Sorting the residuals:**

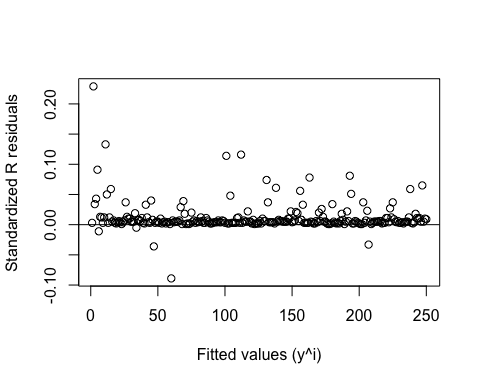
final\_1 <- final\_data[order(-standard\_res),]  
head(final\_1)

## stormdata.Slope stormdata.Velocity y  
## 242 0.003 72.48167 0.06831472  
## 15 0.229 1165.58176 0.06831472  
## 244 0.034 239.71605 0.06831472  
## 245 0.043 272.12690 0.06831472  
## 3 0.091 354.42761 0.06831472  
## 43 -0.011 NaN 0.06831472

**From the above results, we can observe that all the residuals are less than an absolute value of 3 therefore none of them are outliers.**

**Step 4: Visualizing the standard residuals**

plot(final\_1$stormdata.Slope, final\_1$stormdata.y, ylab= 'Standardized R residuals', xlab='Fitted values (y^i)')  
abline(0,0)



**The above figure indicates that the error variance is constant since it is along the horizontal line it means that there is linear relationship between the regressors. In this case a variance stabilization transformation is not required.**

## 2. Performing a Principal Component Analysis

library(datasets)  
library(readr)  
library(dplyr)  
library(base)  
library(stats)  
library(ggplot2)  
library(Matrix)

##   
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack

#library(RcppEigen)  
#install.packages("tidyverse")  
#install.packages("devtools")  
#install.packages("stats19")  
#install.packages("matrixcalc")

**a) Generating the covariance and correlation matrix and dicussing on the variables in the dataset. Creating a dataframe and removing column names from the data**

a<-stormdata$Diameter  
b<-stormdata$Length  
c<-stormdata$Slope  
d<-stormdata$Area  
df<-data.frame(a=a,b=b,c=c,d=d)  
view(df)  
data<- df  
data1<- data[,!names(data) %in% c('law')]  
head(data1)

## a b c d  
## 1 15 195 0.006 176.7146  
## 2 24 164 0.002 452.3893  
## 3 12 44 0.091 113.0973  
## 4 24 347 0.003 452.3893  
## 5 24 80 0.003 452.3893  
## 6 27 43 0.009 572.5553

view(data1)

#### **Removing non-numeric data columns**

data\_new <-data1[,unlist(lapply(data1,is.numeric))]  
data\_new

## a b c d  
## 1 15 195 0.006 176.71459  
## 2 24 164 0.002 452.38934  
## 3 12 44 0.091 113.09734  
## 4 24 347 0.003 452.38934  
## 5 24 80 0.003 452.38934  
## 6 27 43 0.009 572.55526  
## 7 12 32 0.005 113.09734  
## 8 12 382 0.005 113.09734  
## 9 15 550 0.005 176.71459  
## 10 33 200 0.012 855.29860  
...

#### **Scaling the dataset**

scale(data\_new)

## a b c d  
## [1,] -0.53587428 -0.151404840 -0.239090400 -0.47617077  
## [2,] 0.07235098 -0.343270063 -0.400702202 -0.15404615  
## [3,] -0.73861604 -1.085974151 3.195160380 -0.55050722  
## [4,] 0.07235098 0.789353672 -0.360299251 -0.15404615  
## [5,] 0.07235098 -0.863162925 -0.360299251 -0.15404615  
## [6,] 0.27509273 -1.092163352 -0.117881549 -0.01363285  
## [7,] -0.73861604 -1.160244560 -0.279493351 -0.55050722  
## [8,] -0.73861604 1.005975698 -0.279493351 -0.55050722  
## [9,] -0.53587428 2.045761422 -0.279493351 -0.47617077  
## [10,] 0.68057624 -0.120458836 0.003327302 0.31675138  
...  
  
## attr(,"scaled:center")  
## a b c d   
## 22.92941176 219.46274510 0.01191765 584.22227383   
## attr(,"scaled:scale")  
## a b c d   
## 14.79714931 161.57175091 0.02475067 855.80155544

scaledata <- scale(data\_new)  
head(scaledata)

## a b c d  
## [1,] -0.53587428 -0.1514048 -0.2390904 -0.47617077  
## [2,] 0.07235098 -0.3432701 -0.4007022 -0.15404615  
## [3,] -0.73861604 -1.0859742 3.1951604 -0.55050722  
## [4,] 0.07235098 0.7893537 -0.3602993 -0.15404615  
## [5,] 0.07235098 -0.8631629 -0.3602993 -0.15404615  
## [6,] 0.27509273 -1.0921634 -0.1178815 -0.01363285

#### **Assigning a matrix, identifying the head, class and structure of matrix**

my\_matrix<-as.matrix(scaledata)  
head(my\_matrix)

## a b c d  
## [1,] -0.53587428 -0.1514048 -0.2390904 -0.47617077  
## [2,] 0.07235098 -0.3432701 -0.4007022 -0.15404615  
## [3,] -0.73861604 -1.0859742 3.1951604 -0.55050722  
## [4,] 0.07235098 0.7893537 -0.3602993 -0.15404615  
## [5,] 0.07235098 -0.8631629 -0.3602993 -0.15404615  
## [6,] 0.27509273 -1.0921634 -0.1178815 -0.01363285

class(my\_matrix)

## [1] "matrix" "array"

str(my\_matrix)

## num [1:255, 1:4] -0.5359 0.0724 -0.7386 0.0724 0.0724 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:4] "a" "b" "c" "d"  
## - attr(\*, "scaled:center")= Named num [1:4] 22.9294 219.4627 0.0119 584.2223  
## ..- attr(\*, "names")= chr [1:4] "a" "b" "c" "d"  
## - attr(\*, "scaled:scale")= Named num [1:4] 14.7971 161.5718 0.0248 855.8016  
## ..- attr(\*, "names")= chr [1:4] "a" "b" "c" "d"

#### **Estimating the covariance of matrix**

cov(my\_matrix)

## a b c d  
## a 1.00000000 0.1945223 -0.06720233 0.97513550  
## b 0.19452235 1.0000000 -0.19050517 0.20514128  
## c -0.06720233 -0.1905052 1.00000000 -0.07069595  
## d 0.97513550 0.2051413 -0.07069595 1.00000000

covMat <- cov(my\_matrix)  
covMat

## a b c d  
## a 1.00000000 0.1945223 -0.06720233 0.97513550  
## b 0.19452235 1.0000000 -0.19050517 0.20514128  
## c -0.06720233 -0.1905052 1.00000000 -0.07069595  
## d 0.97513550 0.2051413 -0.07069595 1.00000000

#### **Under the covariance, the 1 along the diagonal of the matrix indicates a strong positive correlation between diameter(a), length(b), area(d) and also tells us that the points on the graph will lie on a straight line. Additionally, slope(c) is negatively correlated to the (a), (b), and (c) variables. A smaller slope corresponds to a larger diameter, length and area of pipe.**

#### **Estimating the correlation of matrix**

cor(my\_matrix)

## a b c d  
## a 1.00000000 0.1945223 -0.06720233 0.97513550  
## b 0.19452235 1.0000000 -0.19050517 0.20514128  
## c -0.06720233 -0.1905052 1.00000000 -0.07069595  
## d 0.97513550 0.2051413 -0.07069595 1.00000000

corMat <-cor(my\_matrix)  
corMat

## a b c d  
## a 1.00000000 0.1945223 -0.06720233 0.97513550  
## b 0.19452235 1.0000000 -0.19050517 0.20514128  
## c -0.06720233 -0.1905052 1.00000000 -0.07069595  
## d 0.97513550 0.2051413 -0.07069595 1.00000000

#### **If the covariance values are negative the correlation between variables is also negative.**

#### **Checking if the covariance matrix is orthognal. An orthogonal matrix is a square matrix and must satisfy the condition A\*A(transpose)= Identity matrix**

head(covMat)

## a b c d  
## a 1.00000000 0.1945223 -0.06720233 0.97513550  
## b 0.19452235 1.0000000 -0.19050517 0.20514128  
## c -0.06720233 -0.1905052 1.00000000 -0.07069595  
## d 0.97513550 0.2051413 -0.07069595 1.00000000

A<-as.vector(covMat)  
B<-(t(covMat))

### A is product of the matrix and B is the transpose of matrix

(A\*B)

## a b c d  
## a 1.000000000 0.03783894 0.004516153 0.950889242  
## b 0.037838944 1.00000000 0.036292218 0.042082945  
## c 0.004516153 0.03629222 1.000000000 0.004997918  
## d 0.950889242 0.04208295 0.004997918 1.000000000

**Since the matrix multiplication with its transpose gives us an Identity matrix, it means that the matrix is orthogonal**

### b) Computing the eigenvalues and eigenvectors for the covariance matrix

eigen(covMat)

## eigen() decomposition  
## $values  
## [1] 2.07078827 1.11199991 0.79240885 0.02480297  
##   
## $vectors  
## [,1] [,2] [,3] [,4]  
## [1,] -0.6721454 -0.2009952 0.09509313 0.706242733  
## [2,] -0.2753256 0.5886248 -0.76003645 0.007824820  
## [3,] 0.1356534 -0.7589119 -0.63690632 -0.001123519  
## [4,] -0.6738060 -0.1928068 0.08747707 -0.707925640

eigen1 <- eigen(covMat)  
eigen1$values

## [1] 2.07078827 1.11199991 0.79240885 0.02480297

eigen1$vectors

## [,1] [,2] [,3] [,4]  
## [1,] -0.6721454 -0.2009952 0.09509313 0.706242733  
## [2,] -0.2753256 0.5886248 -0.76003645 0.007824820  
## [3,] 0.1356534 -0.7589119 -0.63690632 -0.001123519  
## [4,] -0.6738060 -0.1928068 0.08747707 -0.707925640

**The above eigenvalues belong to a positive definite since every eigenvalue is greater than zero**

**Computing eigenvalue and eigenvectors of the correlation matrix**

eigen(corMat)

## eigen() decomposition  
## $values  
## [1] 2.07078827 1.11199991 0.79240885 0.02480297  
##   
## $vectors  
## [,1] [,2] [,3] [,4]  
## [1,] 0.6721454 -0.2009952 0.09509313 0.706242733  
## [2,] 0.2753256 0.5886248 -0.76003645 0.007824820  
## [3,] -0.1356534 -0.7589119 -0.63690632 -0.001123519  
## [4,] 0.6738060 -0.1928068 0.08747707 -0.707925640

eigen2 <- eigen(corMat)  
eigen2$values

## [1] 2.07078827 1.11199991 0.79240885 0.02480297

eigen2$vectors

## [,1] [,2] [,3] [,4]  
## [1,] 0.6721454 -0.2009952 0.09509313 0.706242733  
## [2,] 0.2753256 0.5886248 -0.76003645 0.007824820  
## [3,] -0.1356534 -0.7589119 -0.63690632 -0.001123519  
## [4,] 0.6738060 -0.1928068 0.08747707 -0.707925640

#### **Finding the square root of the covariance matrix using the spectral decomposition method. Spectral decomposition is the factorization of a matrix into a canonical form where the matrix is presented in terms of Eigen values and eigenvectors. Each eigen value is associated to an equation. The eigen vectors (direction and magnitude) elongate or shrink by its eigen value.**

lamda1<-eigen1$values[1]  
lamda1

## [1] 2.070788

lamda2<-eigen1$values[2]  
lamda2

## [1] 1.112

e1<-eigen1$vectors[,1]  
e1

## [1] -0.6721454 -0.2753256 0.1356534 -0.6738060

e2<-eigen1$vectors[,2]  
e2

## [1] -0.2009952 0.5886248 -0.7589119 -0.1928068

matrix1<-(lamda1\*matrix(e1)\*e1)+(lamda2\*matrix(e2)\*e2)  
matrix1

## [,1]  
## [1,] 0.9804633  
## [2,] 0.5422592  
## [3,] 0.6785596  
## [4,] 0.9815061

sqrt(matrix1)

## [,1]  
## [1,] 0.9901835  
## [2,] 0.7363825  
## [3,] 0.8237473  
## [4,] 0.9907099

**The square root matrix above is a positive semi definite matrix since eigen values are greater than 0**

**Based on the eigen decomposition we will now determine, how many principal components to select to reduce feature dimensions such that we can capture 85% of variability in the data and this analysis is performed using the correlation matrix.**

**Estimating the total variance explained by each principal component.**

P\_var <- (eigen1$values/sum(eigen1$values))  
P\_var

## [1] 0.517697068 0.277999977 0.198102212 0.006200742

cumsum(P\_var)

## [1] 0.5176971 0.7956970 0.9937993 1.0000000

**We shall select the first three principle component bcause they add upto >85% that is**

PC1 <- 0.517697068  
 PC2 <- 0.277999977  
 PC3 <- 0.198102212  
Total = (PC1+PC2+PC3)   
Total

## [1] 0.9937993

**Choosing the first 3 components explains 99% variability > 85%**

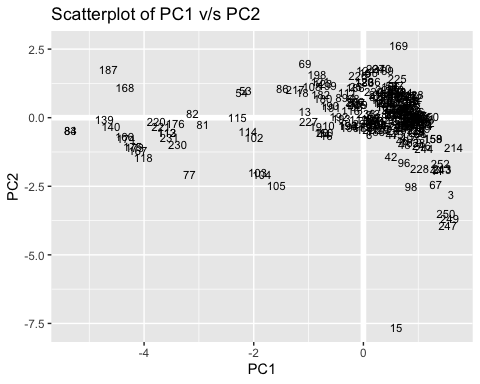
### Task 6: Computing the principal component vectors based and commenting on the interpretation of the PCs

evectors <- eigen1$vectors[, 1:4]  
colnames(evectors) <- c('a', 'b', 'c', 'd')  
row.names(evectors) <- colnames(data1)  
  
PC1 <- as.matrix(scaledata) %\*% evectors[,1]  
PC2 <- as.matrix(scaledata) %\*% evectors[,2]  
PC3 <- as.matrix(scaledata) %\*% evectors[,3]  
PC <- data.frame(pipe\_parameters=row.names(data1), PC1, PC2, PC3)  
head(PC)

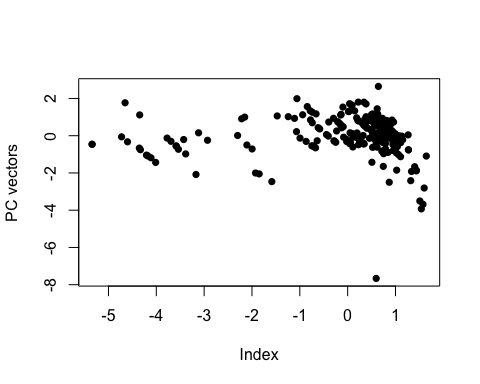
## pipe\_parameters PC1 PC2 PC3  
## 1 1 0.69028437 0.2918451 0.1747394  
## 2 2 0.09532127 0.1171993 0.5095121  
## 3 3 1.59982318 -2.8094767 -1.3280320  
## 4 4 -0.21103822 0.7532275 -0.3770561  
## 5 5 0.24394187 -0.2194848 0.8789167  
## 6 6 0.10899306 -0.6060766 0.9301303

**A scatterplot is created below to show relationship between individual principle components.**

ggplot(PC, aes(PC1, PC2)) + modelr::geom\_ref\_line(h=0) + modelr::geom\_ref\_line(v=0) + geom\_text(aes(label=pipe\_parameters), size=3) + ggtitle ("Scatterplot of PC1 v/s PC2")+ xlab("PC1")+ylab("PC2")



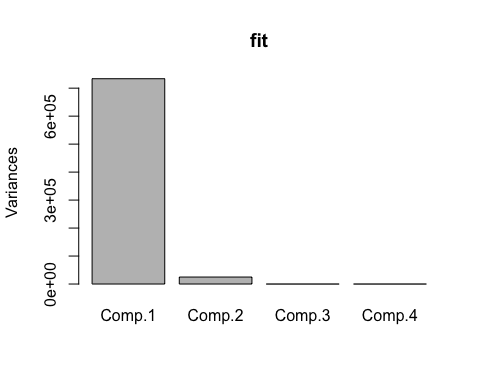
PC1<- as.matrix(scaledata) %\*% eigen1$vectors  
plot(PC1, pch=16, xlab="Index", ylab="PC vectors")

****

**There is a coagulation of vectors to the left corner**

**Checking variation in the PCs using the princomp package to check if the results are accurate. The graph below shows the variances v/s the principal components**

fit<-princomp(covmat= cov(data1))  
screeplot(fit)



#### **This graph shows that the maximum variance in the data is contributed by the principle component 1**

**Computing the data using the correlation matrix and comparing it with the data obtained from the covariance matrix**

P\_cor <- (eigen2$values/sum(eigen2$values))  
P\_cor

## [1] 0.517697068 0.277999977 0.198102212 0.006200742

cumsum(P\_cor)

## [1] 0.5176971 0.7956970 0.9937993 1.0000000

**The principal component values for both covariance and correlation are fairly similar hence in this case also we will choose 3 principle components. In conclusion, even the correlation matrix required 3 PCs to account for variation >80%**

**Conclusion:**

Summarizing the data presented earlier, firstly using the EDA analysis we can conclude from this report the mean concrete pipe diameters range from 40”-50” across a slope of 0-0.06. Moreover, diameter ranges from 10-20 had the highest outliers, which meant that their slopes were different. This helps us learn that smaller pipes have a greater probability of fluctuating slopes which is the reason for maximum outliers. Secondly, the upstream and downstream elevations of water have fairly the same distribution of profiles, normally distributed. Moreover, the mean pipe length is 200 feet which correspond to a mean slope of 0.02 as shown by the density plot. We can also infer that pipe lengths >200 contribute to 80% of the data. Thirdly, the principal component analysis gives us 3 dimensions of maximum importance which capture 99% of the variance in the data and clustering in the upper right of the PCI and PC2 plot. We can reduce the dimensionality to 3 variables. (Peter Nistrup, 2019). Lastly, the linear regression reciprocates that water velocity is directly proportional to the slope of the pipe and the error variance of the regressors is fairly constant along the horizontal and was found to be less than 3 which means that there were very few outliers. The scope of this project is limited since it doesn’t take an account of time series and sound data moreover it estimates results based only on a Concrete pipe with a roughness coefficient of 100 to be on the conservative side. There is scope for further research on how additional variables like water discharges (Q) and head losses affect pipe parameters.

**References:**

* Nistrup, January, (2019), Principle Component Analysis, *Towards Data Science,* <https://towardsdatascience.com/principal-component-analysis-pca-101-using-r-361f4c53a9ff>, Medium.
* City of San Jose, November, (2022), Stormwater Gravity Mains, *SanJoseCA.gov,* <https://data.sanjoseca.gov/dataset/storm-gravity-mains1>
* Hazen Williams equation, October (2022), *Wikipedia*, <https://en.wikipedia.org/wiki/Hazen–Williams_equation>
* Yihui Zhe, July, (2014), Unicode Characters in Shiny Apps, *Shiny*, <https://shiny.rstudio.com/articles/unicode.html>, Posit.
* Logan Kelly, November, (2020), R Practices for learning statistics, *Bookdown*, <https://bookdown.org/logan_kelly/r_practice/p10.html>
* Wickham, Grolemund, January (2017), R for Data Science, <https://r4ds.had.co.nz>, *Bookdown*