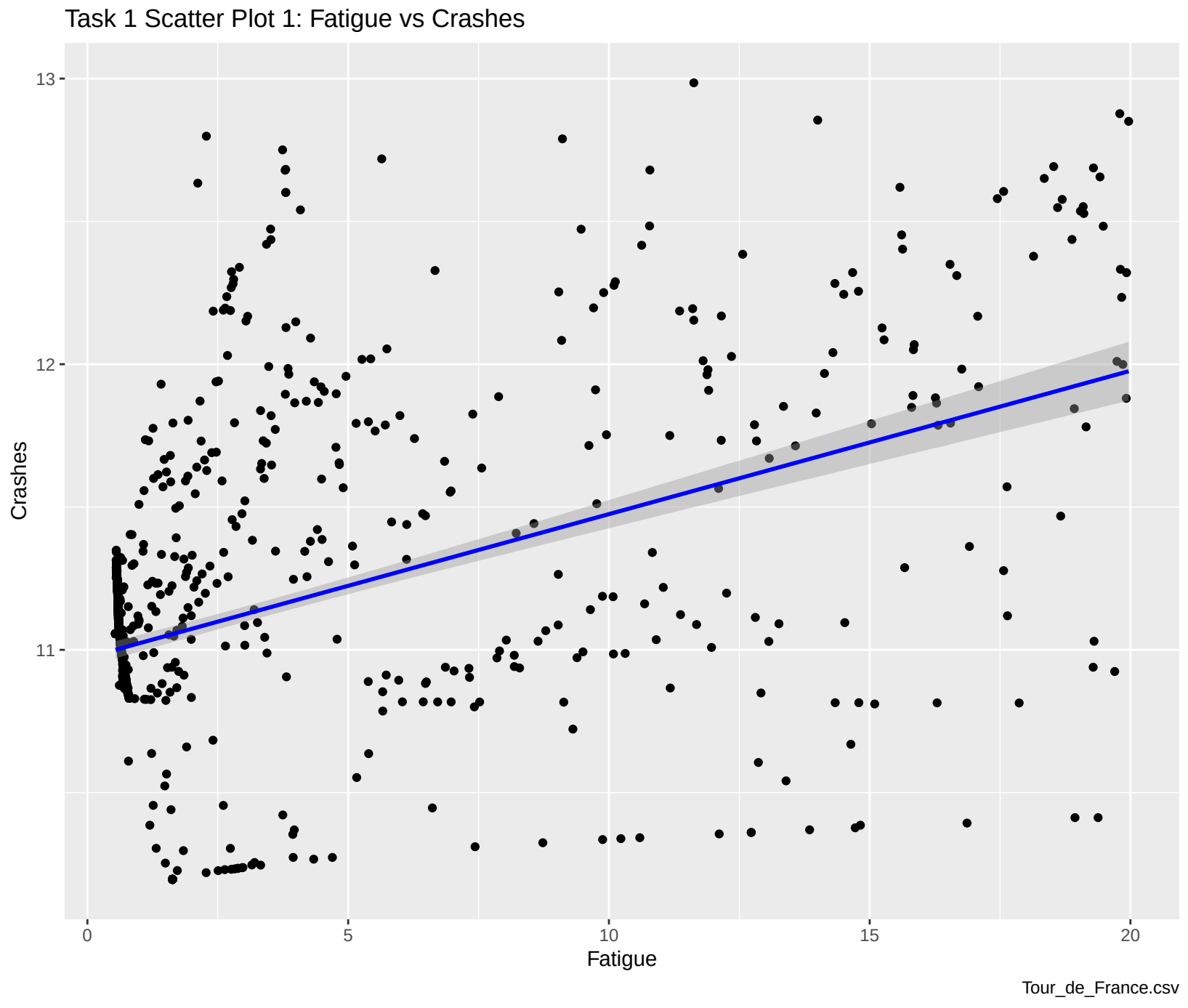
Yeo Ying Xuan 1003835

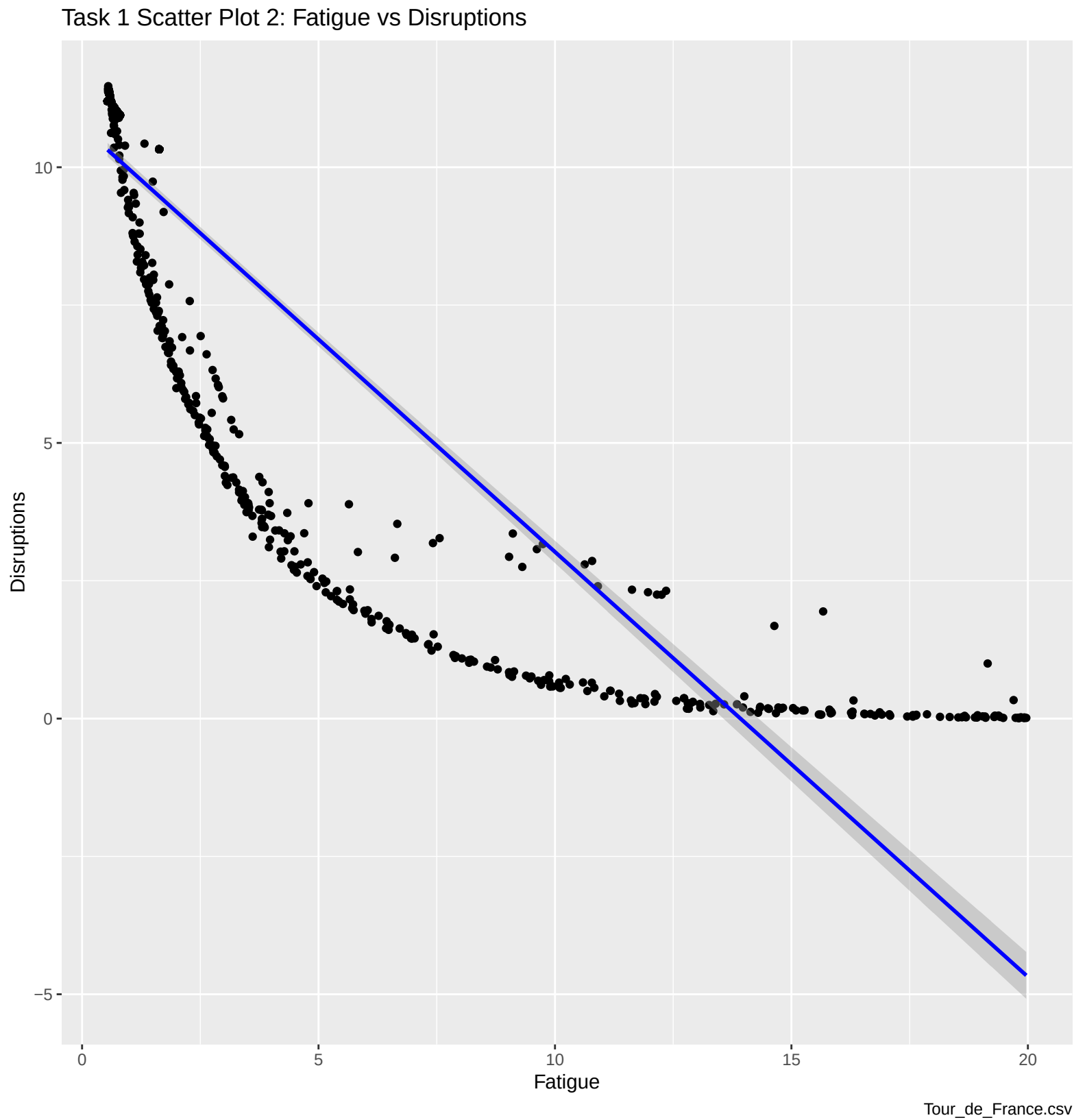
40.014 Engineering Systems Architecture

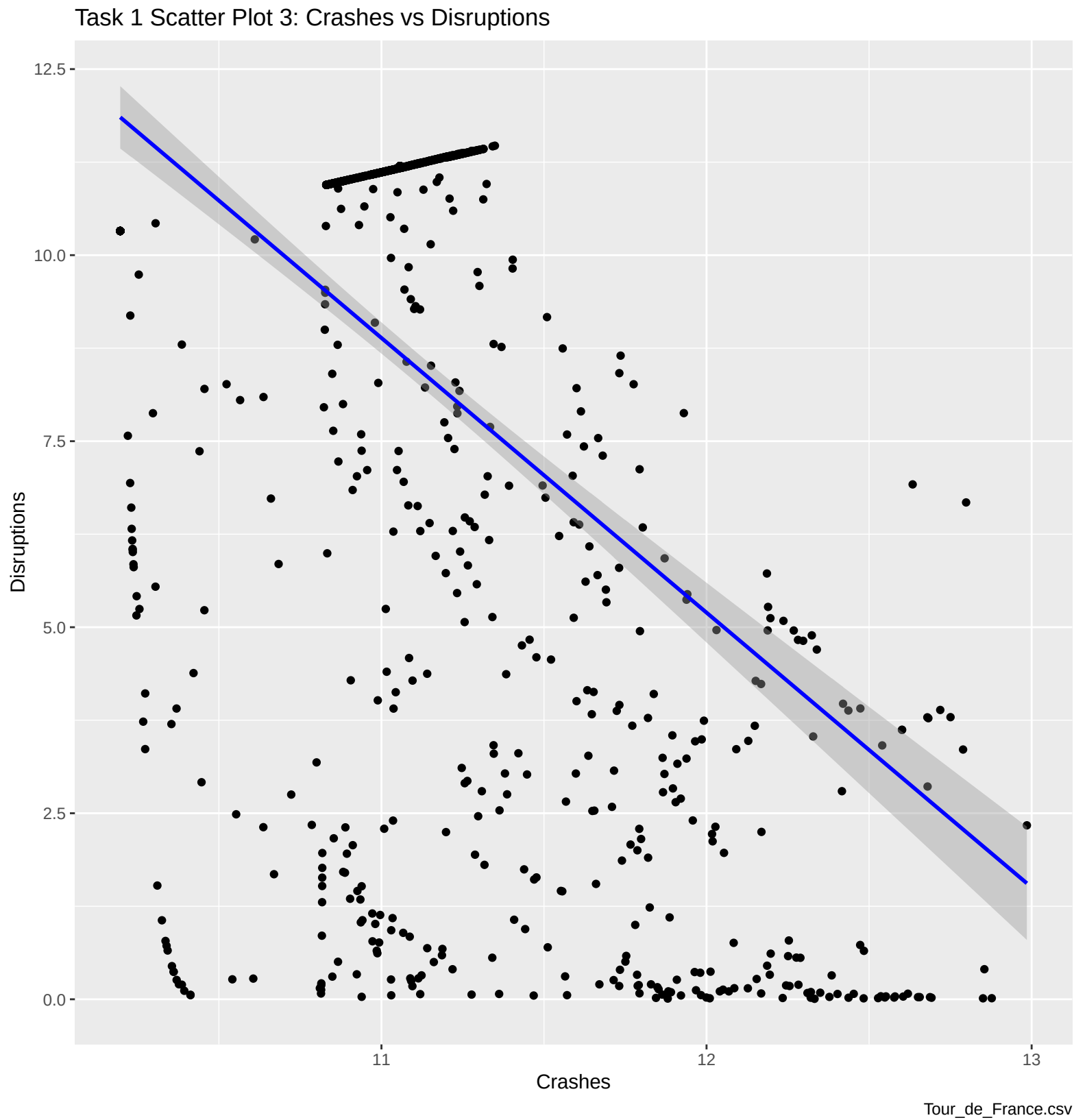
Term 5 2020

Homework 1

Task 1


Task 1 Scatter Plot 1: Fatigue vs Crashes
Crashes
13
12
11
0
5
10
15
20
Fatigue
Tour_de_France.csv


Task 1 Scatter Plot 2: Fatigue vs Disruptions
Disruptions
10
5
0
−5
0
5
10
15
20
Fatigue
Tour_de_France.csv


Task 1 Scatter Plot 3: Crashes vs Disruptions
12.5
10.0
7.5
5.0
2.5
0.0
Disruptions
11
12
13
Crashes
Tour_de_France.csv

Task 2

Plot 1: Fatigue vs Crashes

Based on linear regression model, increase in racing fatigue correlates to increase in expected crashes.

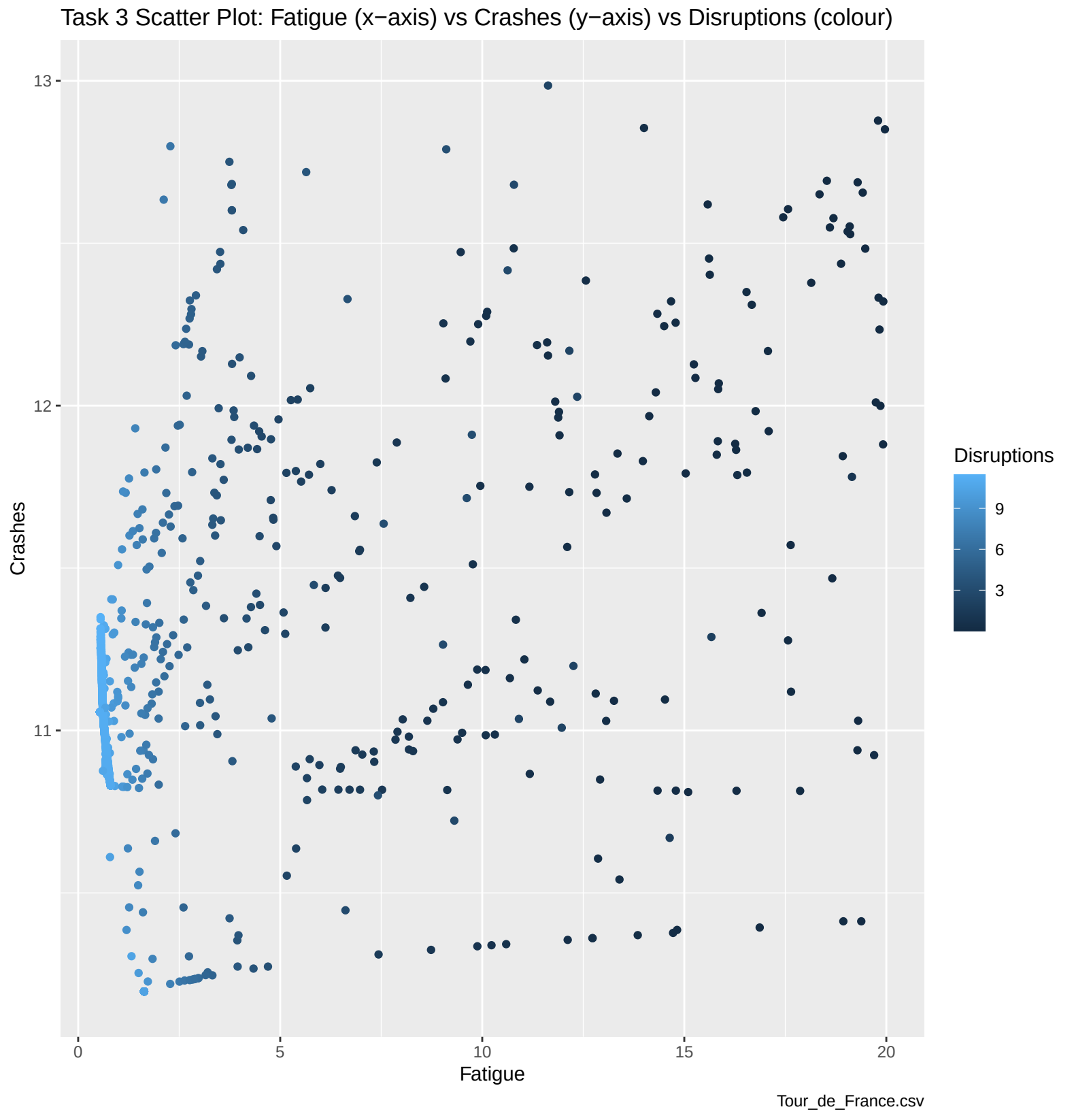
Plot 2: Fatigue vs Disruptions

Based on linear regression model, increase in racing fatigue correlates to decrease in disruptions caused to the regular traffic.

Plot 3: Crashes vs Disruptions

Based on linear regression model, increase in expected crashes correlates to decrease in disruptions caused to the regular traffic.

Task 3


Task 3 Scatter Plot: Fatigue (x−axis) vs Crashes (y−axis) vs Disruptions (colour)
13
12
11
Crashes
0
5
10
15
20
Fatigue
Disruptions
9
6
3
Tour_de_France.csv

Task 4

The objective is to minimise racing fatigue, expected crashes and disruptions caused to the regular traffic.

Based on the scatter plot produced in Task 3 above, majority of the dominated solutions are clustered around the middle left of the plot, as indicated by the data points with lighter shades of blue. Despite fatigue values being minimised and values of expected crashes are mid-range (around 11), disruption levels are much higher relative to other portions of plot.

The plot below shows whether each solution is dominated or non-dominated.

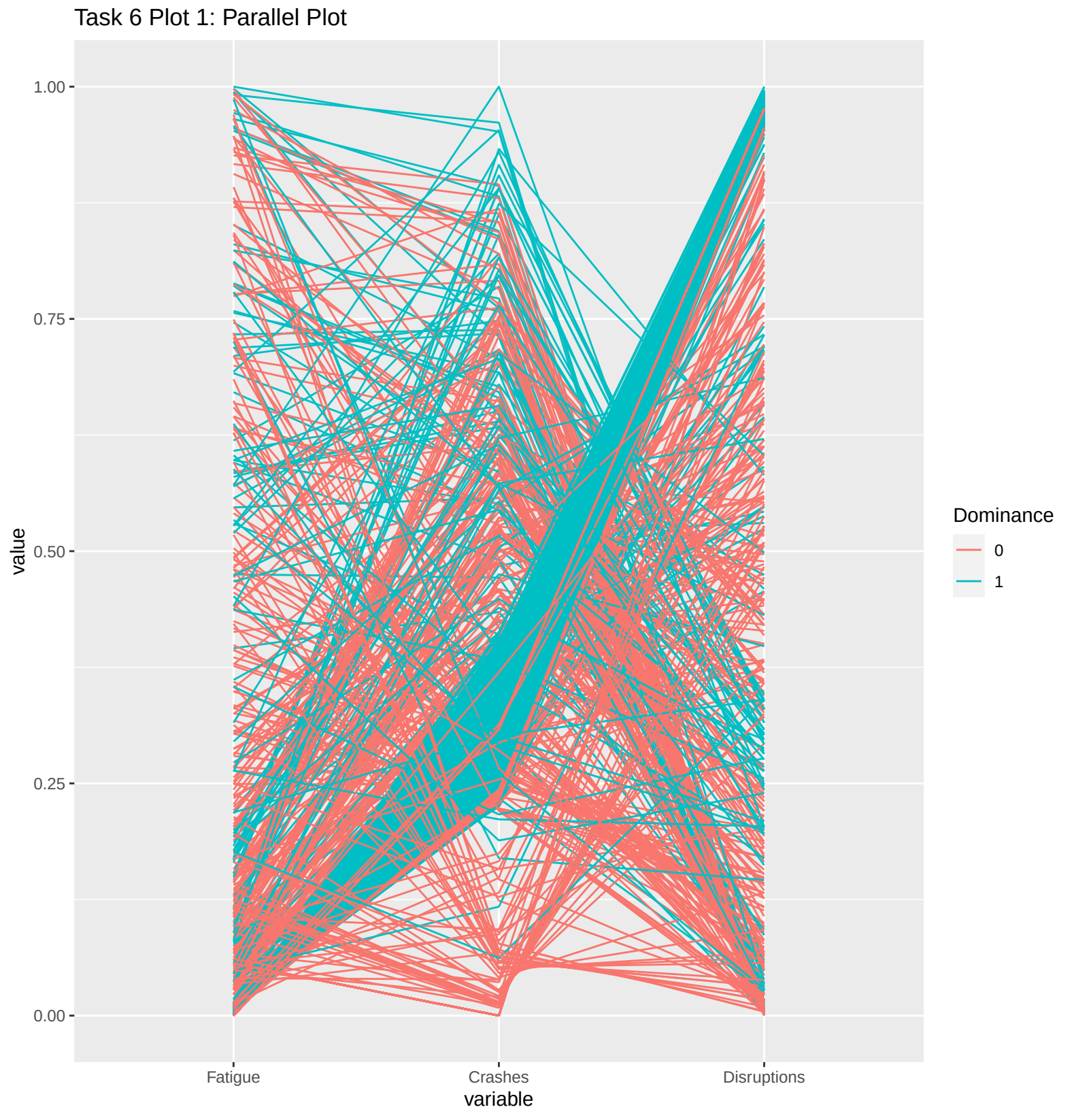

13
12
11
Crashes
0
5
10
15
20
Fatigue
Dominance
Dominated
Non−dominated

Task 5

Non-dominated alternatives = 475

Dominated alternatives = 678

Task 6


Task 6 Plot 1: Parallel Plot
1.00
0.75
0.50
0.25
0.00
value
Fatigue
Crashes
Disruptions
variable
Dominance
0
1

Task 7

Utopia vector = `[0.539508000 10.196563000 0.006677781 ]`

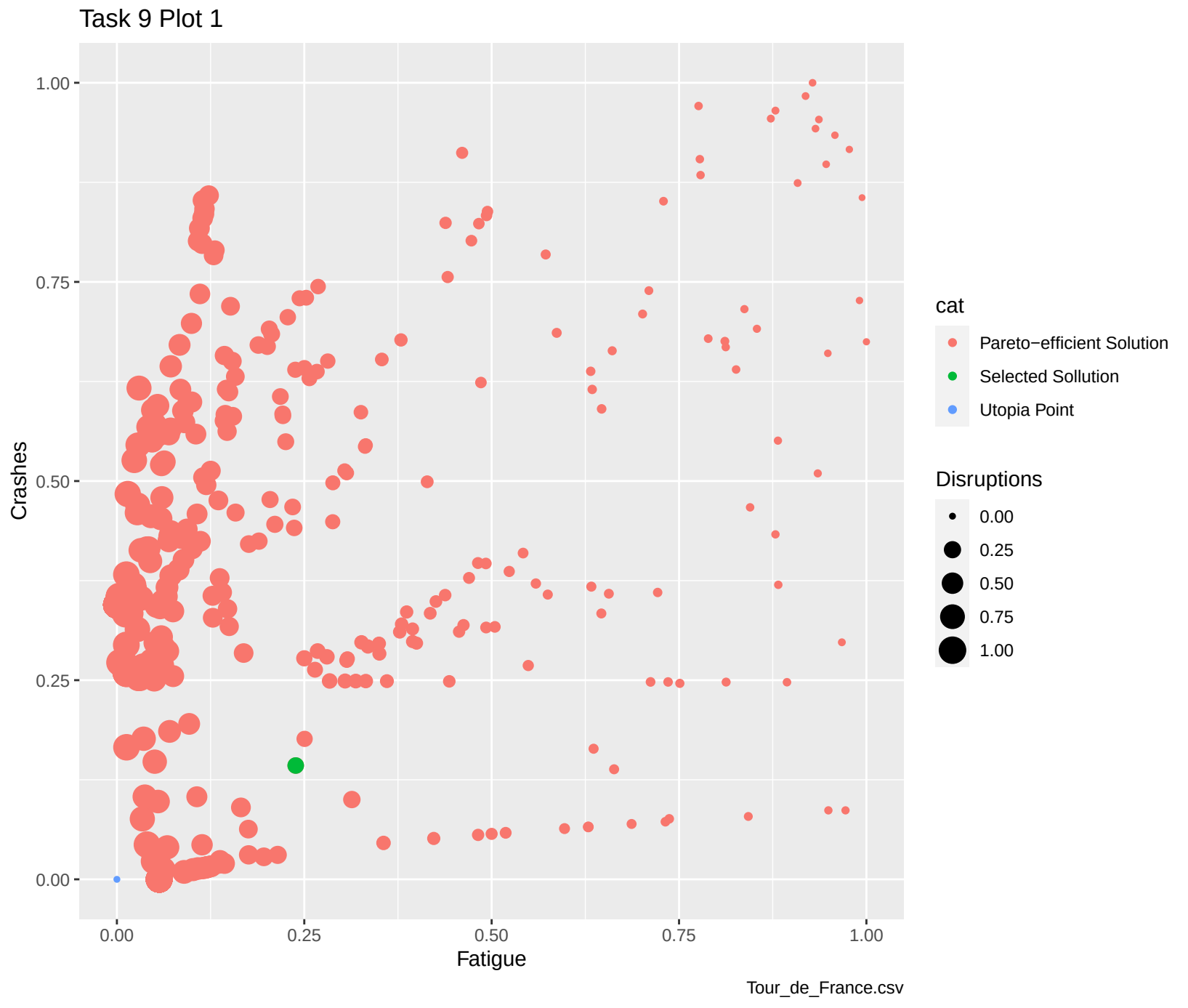
Nadir vector = `[19.92032 12.69208 11.19780]`

Task 8

Index of best alternative across all solutions = 1065

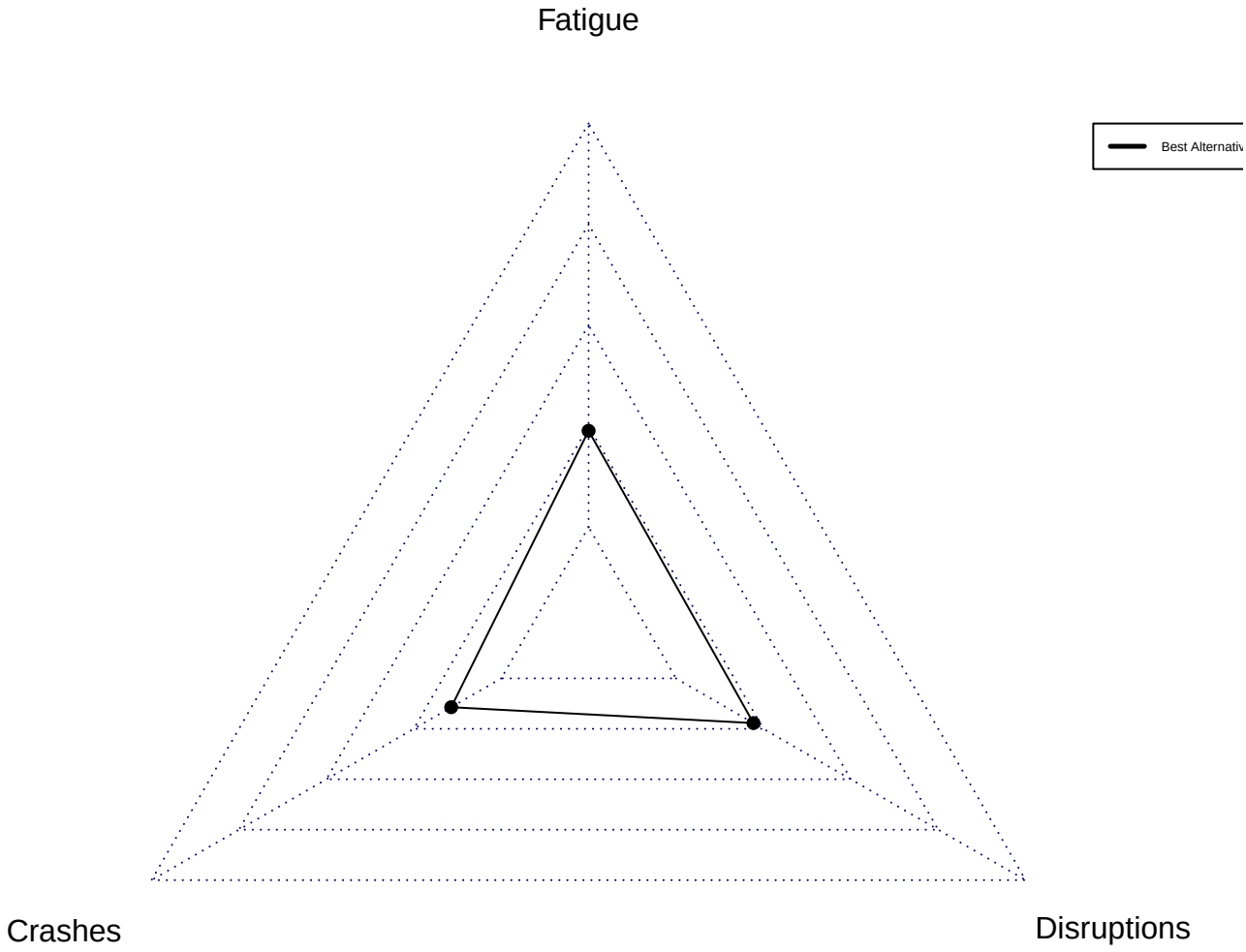
minimum distance = 0.3555774

Task 9


Task 9 Plot 1
1.00
0.75
0.50
0.25
0.00
Crashes
0.00
0.25
0.50
0.75
1.00
Fatigue
Tour_de_France.csv
cat
Pareto−efficient Solution
Selected Sollution
Utopia Point
Disruptions
0.00
0.25
0.50
0.75
1.00

Task 10

Task 10 Plot 1: Best Alternative



R-Script

```
#Homework 1
rm(list=ls())
setwd("~/Documents/SUTD/Term 5/ESA/Homework/Homework 1")

if(!require(ggplot2)){
  install.packages("ggplot2")
  library(ggplot2)
}

if(!require(GGally)) {
  install.packages("GGally")
  library(GGally)
}

if(!require(fmsb)) {
  install.packages("fmsb")
  library(fmsb)
}

dataset <- read.csv(file="Tour_de_France.csv")

# Task 1: three 2D scatter plots

# Scatter Plot 1: Fatigue against Crashes
t1_plot1 <- ggplot(data = dataset, mapping = aes(x = Fatigue, y = Crashes))
+
  geom_point() +
  labs(title = "Task 1 Scatter Plot 1: Fatigue vs Crashes", x = "Fatigue",
y = "Crashes", caption = "Tour_de_France.csv") +
  # geom_smooth(method = 'lm', color = 'blue') +
  geom_smooth(method = 'lm', color = "blue")
t1_plot1
ggsave("T1 Plot1.pdf", t1_plot1)

# Scatter Plot 2: Fatigue against Disruptions
t1_plot2 <- ggplot(data = dataset, mapping = aes(x = Fatigue, y =
Disruptions)) +
  geom_point() +
```

```r
  labs(title = "Task 1 Scatter Plot 2: Fatigue vs Disruptions", x =
"Fatigue", y = "Disruptions", caption = "Tour_de_France.csv") +
  geom_smooth(method = 'lm', color = "blue")
t1_plot2
ggsave("T1 Plot2.pdf", t1_plot2)

# Scatter Plot 3: Crashes against Disruptions
t1_plot3 <- ggplot(data = dataset, mapping = aes(x = Crashes, y =
Disruptions)) +
  geom_point() +
  labs(title = "Task 1 Scatter Plot 3: Crashes vs Disruptions", x =
"Crashes", y = "Disruptions", caption = "Tour_de_France.csv") +
  geom_smooth(method = 'lm', color = "blue")
t1_plot3
ggsave("T1 Plot3.pdf", t1_plot3)



# Task 3
# Plot: Fatigue vs Crashes, Disruptions (colour)
t3_plot1 <- ggplot(data = dataset, mapping = aes(x = Fatigue, y = Crashes,
colour = Disruptions)) +
  geom_point() +
  labs(title = "Task 3 Scatter Plot: Fatigue (x-axis) vs Crashes (y-axis)
vs Disruptions (colour)", x = "Fatigue", y = "Crashes", caption =
"Tour_de_France.csv")
t3_plot1
ggsave("T3 Plot1.pdf", t3_plot1)



#Task 5
source("ParetoSorting_adv.R")

# data in matrix form to input into pareto sorting function
# N x M matrix with N alternatives (rows) and M objectives (columns)
matrix_input <- matrix(c(dataset$Fatigue, dataset$Crashes,
dataset$Disruptions), nrow = nrow(dataset), ncol = ncol(dataset))

result <- ParetoSorting_adv(matrix_input, "MIN")

result

# 1: dominated
# 0: not dominated
```

```r
# dominated count
dominated_count <- length(which(result == 1))
dominated_count
# dominated count: 678

# not dominated count
notdominated_count <- length(which(result == 0))
notdominated_count
# non-dominated count: 475

# function to create a list stating string "dominated" or "non-dominated"
dom_func <- function(list_input) {
  result_list <- c()
  for (i in 1:length(list_input)) {
    if (list_input[i] == 1) {
      name <- "Dominated"
    } else {
      name <- "Non-dominated"
    }
    result_list <- c(result_list, name)
  }
  return(result_list)
}

dom_list <- dom_func(result)
dataset$Dominance <- dom_list
domplot <- ggplot(data = dataset, mapping = aes(x=Fatigue, y = Crashes)) +
  geom_point(aes(color = Dominance)) +
  labs(title = "Domination Plot", caption ="Tour_de_France.csv")
ggsave("T4 Plot1.pdf", domplot)



# Task 6
t6_plot1 <- ggparcoord(data = data.frame(Fatigue = dataset[,1], Crashes =
dataset[,2], Disruptions = dataset[,3], Domination = as.factor(result)),
                       columns = 1:3,
                       title = "Task 6 Plot 1: Parallel Plot",
                       scale = "uniminmax",
                       groupColumn = "Domination")
t6_plot1
ggsave("T6 Plot1.pdf", t6_plot1)
```

```r
# Task 7
# list of non-dominated solutions
index <- which(result==0)

# Utopia vector
utopia_fatigue <- min(dataset[index, 1])
utopia_crashes <- min(dataset[index, 2])
utopia_disruptions <- min(dataset[index, 3])
utopia_vector <- c(utopia_fatigue, utopia_crashes, utopia_disruptions)

# Nadir vector
nadir_fatigue <- max(dataset[index, 1])
nadir_crashes <- max(dataset[index, 2])
nadir_disruptions <- max(dataset[index, 3])
nadir_vector <- c(nadir_fatigue, nadir_crashes, nadir_disruptions)



# Task 8
# normalise non-dominated set of solutions
fatigue_norm <- (dataset[index, 1] - utopia_fatigue) / (nadir_fatigue -
utopia_fatigue)
crashes_norm <- (dataset[index, 2] - utopia_crashes) / (nadir_crashes -
utopia_crashes)
disruptions_norm <- (dataset[index, 3] - utopia_disruptions) /
(nadir_disruptions - utopia_disruptions)

# check normality
utopia_norm <- c(min(fatigue_norm), min(crashes_norm),
min(disruptions_norm))

distance <- rep(0, times = notdominated_count)
for (i in 1:notdominated_count) {
  distance[i] <- sqrt((fatigue_norm[i] - utopia_norm[1])^2 +
(crashes_norm[i] - utopia_norm[2])^2 + (disruptions_norm[i] -
utopia_norm[3])^2)
}
min_dist <- min(distance)
# min_dist 0.3555774
index_min_dist <- which.min(distance)
# 403th row value in the set of non-dominated solutions
index[403]
# 1065th solution in the entire dataset
```

```r
# Task 9
new_norm_data <- data.frame(Fatigue = fatigue_norm, Crashes = crashes_norm,
Disruptions = disruptions_norm)
utopia <- data.frame(Fatigue = 0, Crashes = 0, Disruptions = 0)
selected <- data.frame(Fatigue = fatigue_norm[403], Crashes =
crashes_norm[403], Disruptions = disruptions_norm[403])

new_norm_data$cat <- "Pareto-efficient Solution"
utopia$cat <- "Utopia Point"
selected$cat <- "Selected Solution"

df <- rbind(new_norm_data, utopia, selected)

t9_plot1 <- ggplot(data = df, mapping = aes(x = Fatigue, y = Crashes, size
= Disruptions)) +
  geom_point(aes(color=cat)) +
  labs(title = "Task 9 Plot 1", x = "Fatigue", y = "Crashes")
t9_plot1
ggsave("T9 Plot1.pdf", t9_plot1)



# Task 10
max_val <- c(1, 1, 1)
min_val <- c(0, 0, 0)
selected_point <- selected[, 1:3]

radar_data <- rbind(max_val, min_val, selected[, 1:3])
t10_plot1 <- radarchart(df = radar_data, title = "Task 10 Plot 1: Best
Alternative")
legend(1, 1, lty = c(1, 2), lwd = c(2.5, 2.5), col = c("black"), c("Best
Alternative"), cex = 0.4)
```