simulate_gas

Benjamin Chu

Code description

For the computing project I chose to implement part of the code needed for my final project. For my final project, I need to simulate the process of refueling under gas price fluctuations. Ultimately I am going to compare the total expenditure of control group vs test group, where they employ different refueling strategies.

Here I implemented the control group. Everyday they will lose x% of fuel, where $x \sim N(\mu, \sigma^2)$ and they will blindly refuel whenever they drop below y% of gas. Currently all numbers (mean, SD, tank size...etc. are all arbitrarily chosen). I have not made the graphs pretty yet, but I figured it is sufficient to illustrate my point. Below are descriptions of major functions:

`control`: Simulates the refueling process of the control group. Keeps track of expense in a list.

`lose-fuel`: Everyday people will lose x% of fuels, where $x \sim N(\mu, \sigma^2)$.

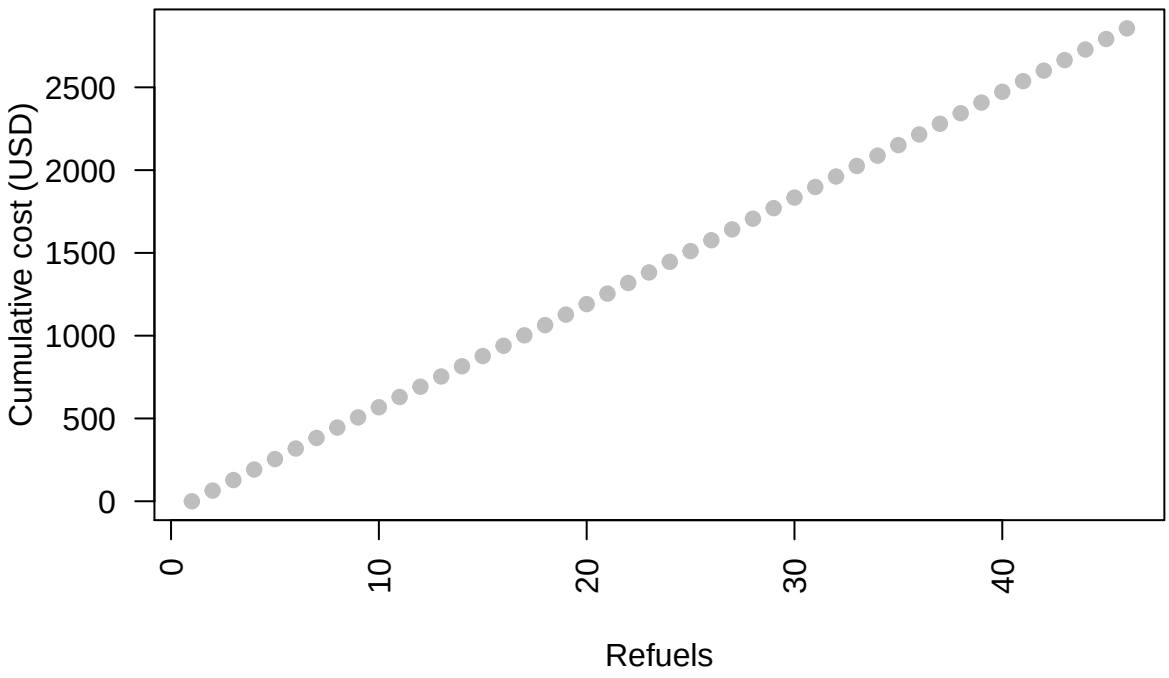
```
library(readxl) # parsing excel
library(lubridate) # dates/months/years addition made easy
```

```
##
## Attaching package: 'lubridate'
```

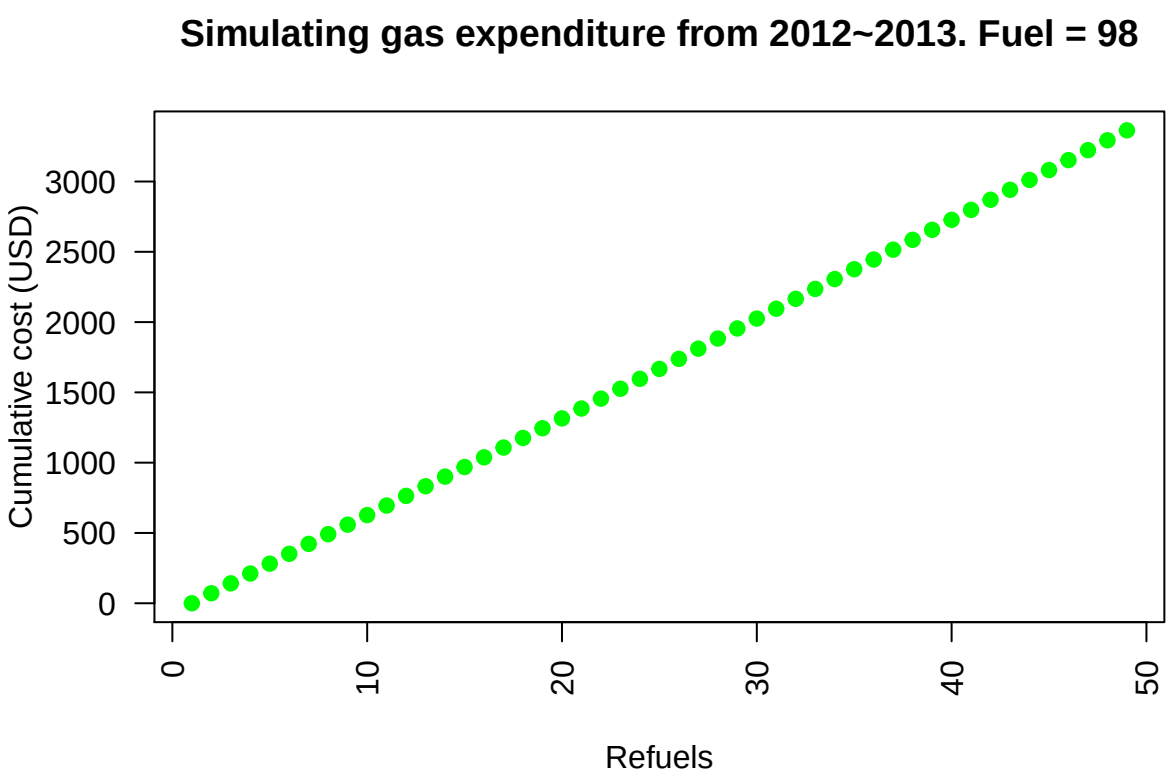
```
## The following object is masked from 'package:base':
##
##     date
```

```
mydata <- read_excel("CPC_data_short.xlsx")

X_VALUE = 10
Y_VALUE = 30
Z_VALUE = 40

assign_fuel_type <- function(fuel_type) { # identity fuel type (92/95/98)
  dummy_var = 0
  if (fuel_type == 92) {dummy_var = 2}
  else if (fuel_type == 95) {dummy_var = 3}
  else if (fuel_type == 98) {dummy_var = 4}
  else {
    print("please enter either 92, 95, or 98")
    return()
  }
  return(dummy_var)
}

lose_fuel <- function() { # everyday lose fuel ~ normal
  SD = 5
  lost = rnorm(1, X_VALUE, SD) #generate x ~ N(mean, SD^2)

  if (lost < 0) {
    lost = 0
  }
  return(lost)
```

```
}

control <- function(fuel_type, table) {
  # fuel_type = 92, 95, or 98

  which_column = assign_fuel_type(fuel_type)
  my_price_list = table[[which_column]]
  my_date_list = table[[1]]

  current_date = ymd(my_date_list[1])
  current_price = my_price_list[1]
  my_price_list <- my_price_list[-1] #delete first element
  my_date_list <- my_date_list[-1] #delete first element
  last_day = ymd(my_date_list[length(my_date_list)])

  total_expense = 0 # keep track of total money spent
  expense_list <- list(total_expense) # keeps track of marginal expense
  current_fuel = 100

  while (current_date != last_day) {
    if(current_fuel < Y_VALUE) { # refuel when necessary
      total_expense = total_expense + 57.0 * current_price #57 liters ~ 15 gallons ~ approx gas tank si
      expense_list <- append(expense_list, total_expense / 30.0)
      current_fuel = 100
    }

    current_fuel = current_fuel - lose_fuel()
    current_date = current_date + ddays(1)

    #check if today needs price adjustments. Remember to watch out for NA.
    if (current_date == ymd(my_date_list[1])) {
      my_date_list <- my_date_list[-1]
      if (!is.na(my_price_list[1])) {
        current_price = my_price_list[1]
      }
      my_price_list <- my_price_list[-1]
    }
  }
  return(expense_list)
}


test_1 = control(92, mydata)
number_1 = length(test_1)
count_1 = 1:number_1
plot(count_1, test_1, col="grey", las = 2, main = "Simulating gas expenditure from 2012~2013. Fuel = 92
```




```
test_2 = control(98, mydata)
number_2 = length(test_2)
count_2 = 1:number_2
plot(count_2, test_2, col="green", las = 2, main = "Simulating gas expenditure from 2012~2013. Fuel = 98
```


Simulating gas expenditure from 2012~2013. Fuel = 98
Cumulative cost (USD)
3000
2500
2000
1500
1000
500
0
0
10
20
30
40
50
Refuels