Part 1

All the early steps shown were to create the dataframe which is clean and tidy.

```
#This chunk of code is to load and library important packages.

library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts -------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(hyperSpec)
```

```
## Loading required package: lattice
```

```
## Loading required package: grid
```

```
## Package hyperSpec, version 0.99-20180627
##
## To get started, try
##    vignette ("hyperspec")
##    package?hyperSpec
##    vignette (package = "hyperSpec")
##
## If you use this package please cite it appropriately.
##    citation("hyperSpec")
## will give you the correct reference.
##
## The project homepage is http://hyperspec.r-forge.r-project.org
```

```
##
## Attaching package: 'hyperSpec'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
library(ggplot2)
library(baseline)
```

```
##
## Attaching package: 'baseline'
```

```
## The following object is masked from 'package:stats':
##
##     getCall
```

```
library(netSEM)
```

```
#This loads and saves important files into dataframes.

#create vector list of data files
keyFiles<- list.files(path = "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
```

```
                      pattern = ".csv")

#Loads the color measurements files
step0colordata <- list.files(path =
                               "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
                             pattern = ".csv")
step1colordata <- list.files(path =
                               "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
                             pattern = ".csv")
step2colordata <- list.files(path =
                               "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
                             pattern = ".csv")
step3colordata <- list.files(path =
                               "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
                             pattern = ".csv")
step4colordata <- list.files(path =
                               "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardoca
                             pattern = ".csv")

#Loads the FTIR measurements files
step0ftirdata <- list.files(path =
                              "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocat
                            pattern = ".csv")
step1ftirdata <- list.files(path =
                              "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocat
                            pattern = ".csv")
step2ftirdata <- list.files(path =
                              "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocat
                            pattern = ".csv")
step3ftirdata <- list.files(path =
                              "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocat
                            pattern = ".csv")
step4ftirdata <- list.files(path =
                              "H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocat
                            pattern = ".csv")
```

```
#This next chunk of code creates indivual exposure hour and sample key data frames.

samplekey <- read.csv("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocats/acry

exposurehours <- read.csv("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocats/
```

```
#This chunk of code creates the final data frame, fills it in with data and removes irrelevant data for

#Creates the final data frame to be used
finaldata <- data.frame(NULL)

#Creates empty color values
colordata <- NULL


# This fills in colordata with step 0 color measurements. It is done by reading each file and finally o
for (i in step0colordata) {
```

```
  readdata <- read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardo
    cbind("Step" = 0)

  colordata <- rbind(colordata, readdata)

}

# Same as before but with step 1 data.
for (i in step1colordata) {

  # read each file
  readdata <- read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardo
    cbind("Step" = 1)

  # rbind data to organize it
  colordata <- rbind(colordata, readdata)

}

# Same as before but with step 2 data.
for (i in step2colordata) {

  # read each file
  readdata <- read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardo
    cbind("Step" = 2)

  # rbind data to organize it
  colordata <- rbind(colordata, readdata)

}

# Same as before but with step 3 data.
for (i in step3colordata) {

  # read each file
  readdata <- read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardo
    cbind("Step" = 3)

  # rbind data to organize it
  colordata <- rbind(colordata, readdata)

}

# Same as before but with step 4 data.
for (i in step4colordata) {

  # read each file
  readdata <- read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardo
    cbind("Step" = 4)

  # rbind data to organize it
  colordata <- rbind(colordata, readdata)
```

```
}


#removes irrelevent data from colordata
colordata <- transmute(colordata, ID = ID, YI = YI.E313..D65.10., Haze = Haze...D65.10, Step = Step)
```

```
# This dataframe does for FTIR what we just did for color. The data values are offset, normalized, base
# Like before, we first create the dataframe
ftirdata <-  data.frame("ID" = factor(),
             "peak1250" = numeric(),
             "peak1700" = numeric(),
             "peak2900" = numeric(),
             "peak3350" = numeric(),
             "Step" = numeric())
#Next, we are creating a hyperspec object from the file. This will contain all spectral data for the sa
readdata <- read.csv("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHardocats/ftir/
spectraldata <- new("hyperSpec", wavelength = readdata[,1],
               spc=t(readdata[,-1]),
               data = data.frame(colnames(readdata[,-1])))
#Adjusting the wavelength range. Did this to reach appropriate range.
spectraldata <-  spectraldata[,, 1200 ~ 3500]
#Corrected the offsets and baseline and wrote down the normalization factor
offsets <- apply(spectraldata, 1, min)
spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
spectraldata.corrected[[]] <- getCorrected(correctbaseline)
normalizationfactor <- 1/max(spectraldata.corrected[,,1680~1720])

# This fills in ftir data with step 0 ftir measurements. It does this by creating a hyperspecobject
for (i in step0ftirdata) {

  #Creating a hyperspec object like before.
  readdata <-    read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHa
  spectraldata <- new("hyperSpec", wavelength = readdata[,1],
                  spc=t(readdata[,-1]),
                  data = data.frame(colnames(readdata[,-1])))
  #adjust wavelength range to relevant range. Correcting offsets, writing down normalizing the peak to
  spectraldata <-  spectraldata[,, 1200 ~ 3500]
  offsets <- apply(spectraldata, 1, min)
  spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
  correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
  spectraldata.corrected[[]] <- getCorrected(correctbaseline)
  spectraldata.corrected <- sweep(spectraldata.corrected,1,normalizationfactor,"*")

  #Adding peaks and sample ID to ftirdata
  for (j in 1:nrow(spectraldata)){
    ftirdata <- add_row(ftirdata,
            ID = spectraldata.corrected[[j,1]][,1],
            peak1250 = max(spectraldata.corrected[j,,1230~1270]),
            peak1700 = max(spectraldata.corrected[j,,1680~1720]),
            peak2900 = max(spectraldata.corrected[j,,2880~2920]),
            peak3350 = max(spectraldata.corrected[j,,3330~3370]),
            Step = 0
    )
```

```
  }
}

# This fills in ftir data with step 1 ftir measurements using the same method as for step 0 ftir measur
for (i in step1ftirdata) {

  #Creating a hyperspec object like before.
  readdata <-    read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHa
  spectraldata <- new("hyperSpec", wavelength = readdata[,1],
                spc=t(readdata[,-1]),
                data = data.frame(colnames(readdata[,-1])))
  #adjusting wavelength range to relevant range. Correcting offsets, writing down normalizing the peak
  spectraldata <-  spectraldata[,, 1200 ~ 3500]
  offsets <- apply(spectraldata, 1, min)
  spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
  correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
  spectraldata.corrected[[]] <- getCorrected(correctbaseline)
  spectraldata.corrected <- sweep(spectraldata.corrected,1,normalizationfactor,"*")

  #Adding peaks and sample ID to ftirdata
  for (j in 1:nrow(spectraldata)){
    ftirdata <- add_row(ftirdata,
            ID = spectraldata.corrected[[j,1]][,1],
            peak1250 = max(spectraldata.corrected[j,,1230~1270]),
            peak1700 = max(spectraldata.corrected[j,,1680~1720]),
            peak2900 = max(spectraldata.corrected[j,,2880~2920]),
            peak3350 = max(spectraldata.corrected[j,,3330~3370]),
            Step = 1
    )
  }
}


# This fills in ftir data with step 2 ftir measurements using the same method as for step 0 and step 1
for (i in step2ftirdata) {

  #Creating a hyperspec object like before.
  readdata <-    read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHa
  spectraldata <- new("hyperSpec", wavelength = readdata[,1],
                spc=t(readdata[,-1]),
                data = data.frame(colnames(readdata[,-1])))
  #adjusting wavelength range to relevant range. Correcting offsets, writing down normalizing the peak
  spectraldata <-  spectraldata[,, 1200 ~ 3500]
  offsets <- apply(spectraldata, 1, min)
  spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
  correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
  spectraldata.corrected[[]] <- getCorrected(correctbaseline)
  spectraldata.corrected <- sweep(spectraldata.corrected,1,normalizationfactor,"*")

  #Adding peaks and sample ID to ftirdata
  for (j in 1:nrow(spectraldata)){
    ftirdata <- add_row(ftirdata,
            ID = spectraldata.corrected[[j,1]][,1],
```

```
            peak1250 = max(spectraldata.corrected[j,,1230~1270]),
            peak1700 = max(spectraldata.corrected[j,,1680~1720]),
            peak2900 = max(spectraldata.corrected[j,,2880~2920]),
            peak3350 = max(spectraldata.corrected[j,,3330~3370]),
            Step = 2
    )
  }
}


# This fills in ftir data with step 3 ftir measurements using the same method as the previous steps.
for (i in step3ftirdata) {

  #Creating a hyperspec object like before.
  readdata <-    read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHa
  spectraldata <- new("hyperSpec", wavelength = readdata[,1],
                spc=t(readdata[,-1]),
                data = data.frame(colnames(readdata[,-1])))
  #adjusting wavelength range to relevant range. Correcting offsets, writing down normalizing the peak
  spectraldata <-  spectraldata[,, 1200 ~ 3500]
  offsets <- apply(spectraldata, 1, min)
  spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
  correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
  spectraldata.corrected[[]] <- getCorrected(correctbaseline)
  spectraldata.corrected <- sweep(spectraldata.corrected,1,normalizationfactor,"*")

  #Adding peaks and sample ID to ftirdata
  for (j in 1:nrow(spectraldata)){
    ftirdata <- add_row(ftirdata,
            ID = spectraldata.corrected[[j,1]][,1],
            peak1250 = max(spectraldata.corrected[j,,1230~1270]),
            peak1700 = max(spectraldata.corrected[j,,1680~1720]),
            peak2900 = max(spectraldata.corrected[j,,2880~2920]),
            peak3350 = max(spectraldata.corrected[j,,3330~3370]),
            Step = 3
    )
  }
}

# This fills in ftir data with step 4 ftir measurements using the same method as the previous steps.
for (i in step4ftirdata) {

  #Creating a hyperspec object like before.
  readdata <-    read.csv(paste0("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHa
  spectraldata <- new("hyperSpec", wavelength = readdata[,1],
                spc=t(readdata[,-1]),
                data = data.frame(colnames(readdata[,-1])))
  #adjusting wavelength range to relevant range. Correcting offsets, writing down normalizing the peak
  spectraldata <-  spectraldata[,, 1200 ~ 3500]
  offsets <- apply(spectraldata, 1, min)
  spectraldata.corrected <- sweep(spectraldata, 1, offsets, "-")
  correctbaseline <- baseline(spectraldata.corrected[[]], method = "modpolyfit", degree = 4)
  spectraldata.corrected[[]] <- getCorrected(correctbaseline)
```

```
  spectraldata.corrected <- sweep(spectraldata.corrected,1,normalizationfactor,"*")

  #Adding peaks and sample ID to ftirdata
  for (j in 1:nrow(spectraldata)){
    ftirdata <- add_row(ftirdata,
            ID = spectraldata.corrected[[j,1]][,1],
            peak1250 = max(spectraldata.corrected[j,,1230~1270]),
            peak1700 = max(spectraldata.corrected[j,,1680~1720]),
            peak2900 = max(spectraldata.corrected[j,,2880~2920]),
            peak3350 = max(spectraldata.corrected[j,,3330~3370]),
            Step = 4
    )
  }
}
```

```
#We now turn to cleaning and tidying the dataframe.

#clean dat_color:
colordata <- colordata%>%
  filter(nchar(as.character(ID)) == 10)

#clean dat_FTIR
ftirdata <- ftirdata%>%
  filter(nchar(as.character(ID)) == 10)
```

```
#Finally, we will be assembling our data starting frm the key file with appropriate observations being

finaldata <- samplekey%>%
  mutate(Haze = as.numeric(NA),
         YI = as.numeric(NA),
         peak1250 = as.numeric(NA),
         peak1700 = as.numeric(NA),
         peak2900 = as.numeric(NA),
         peak3550 = as.numeric(NA),
         hours = as.numeric(NA),
         dose = as.numeric(NA))

#Next we will make a dose calculator function. It will take exposure and step number as the arguments a

dosecalculator <- function(dataexposurehours, exposure, step){
  dose <- as.numeric(NA)
  hours <- as.numeric(NA)
  dataexposurehours <- read.csv("H:/Git/18f-dsci351-351m-451-axm949/1-assignments/proj/proj1/AcrylicHar
  #This is for baseline exposure
  if (exposure == "baseline"){
    dose = 0
  }
  else{#This is for mASTG154 exposure
    if (exposure == "mASTMG154"){
      hours = sum(dataexposurehours$mASTG154[c(1:(step+1))])
      dose = 0.21835*hours
    }
    else{#This is for ASTG154 exposure
      if (exposure == "ASTMG154"){
```

```
        hours = sum(dataexposurehours$ASTMG154[c(1:(step+1))])
        dose = 0.21835*hours*(2/3)
      }
      else{#This is for ASTG155 exposure
        if (exposure == "ASTMG155"){
          hours = sum(dataexposurehours$ASTMG155[c(1:(step+1))])
          dose = 0.09382*hours
        }
        else{#This is for HF exposure
          if (exposure == "HF"){
            hours = sum(dataexposurehours$HF[c(1:(step+1))])
            dose = 0
          }
          else{#This is for 1x exposure
            if (exposure == "1x"){
              hours = sum(dataexposurehours$X1x[c(1:(step+1))])
              dose = 0.04599*hours
            }
            else{#This is for 5x exposure
              if (exposure == "5x"){
                hours = sum(dataexposurehours$X5x[c(1:(step+1))])
              }else{dose = NA}}}}}}
  return(dose)
}

#We can now assemble finaldata from colordata and ftirdata. We do this then add the dose column to our
finaldata <- colordata%>%
  left_join(ftirdata, by = c("ID","Step"))%>%
  left_join(select(samplekey, -Step.Number.Retained), by = c("ID" = "Sample.Number"))%>%
  filter(!is.na(peak1250), !is.na(Product.Name), !is.na(Exposure), !is.na(Step))
```

```
## Warning: Column `ID` joining factors with different levels, coercing to
## character vector
```

```
## Warning: Column `ID`/`Sample.Number` joining character vector and factor,
## coercing into character vector
```

```
finaldata <- finaldata%>%
  mutate(Dose = mapply(dosecalculator, exp = Exposure, step = Step))

finaldata <- distinct(finaldata)

write.csv(finaldata, file = "finaldataset.csv")
```

#Part 1 - Answers

1) The dimensions are shown below

```
dim(finaldata)
```

```
## [1] 250  11
```

2) The heads and tails of the data are shown below

```
head(finaldata)
```

```
##           ID   YI Haze Step   peak1250  peak1700   peak2900   peak3350
## 1 sa22068.00 1.68  1.4    0 0.08780498 0.2487969 0.04547991 0.01245608
```

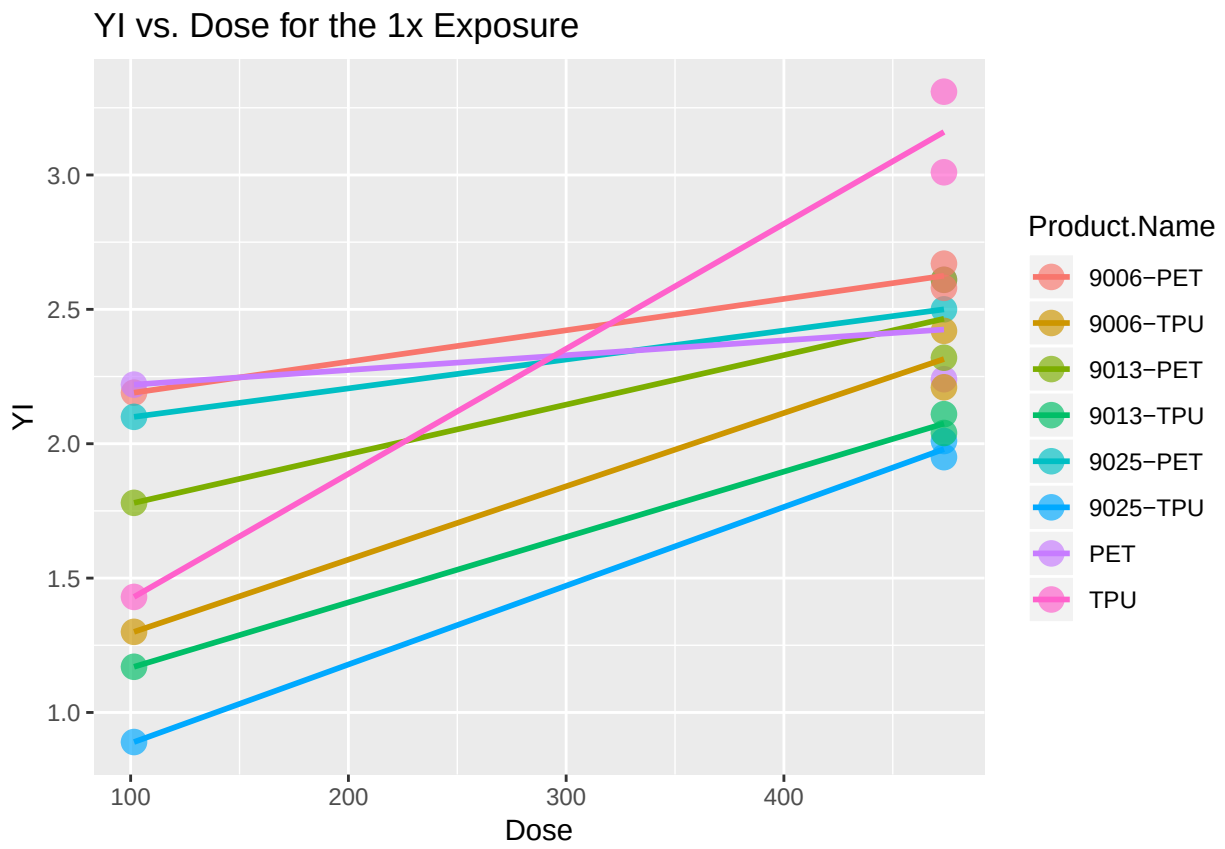
```
## 2 sa22070.09 1.62  1.6    0 0.13348515 0.2921250 0.04603217 0.01571835
## 3 sa22071.00 0.57  2.2    0 0.29153440 1.0000000 0.15800139 0.04448329
## 4 sa22073.09 0.62  4.4    0 0.33962229 0.7862463 0.25817840 0.06764771
## 5 sa22074.00 1.69  1.9    0 0.15977944 0.3340767 0.05841827 0.01838441
## 6 sa22076.09 1.68  2.0    0 0.14297743 0.2771517 0.05850802 0.01822855
##   Product.Name Exposure Dose
## 1     9006-PET baseline    0
## 2     9006-PET baseline    0
## 3     9006-TPU baseline    0
## 4     9006-TPU baseline    0
## 5     9013-PET baseline    0
## 6     9013-PET baseline    0
```

```
tail(finaldata)
```

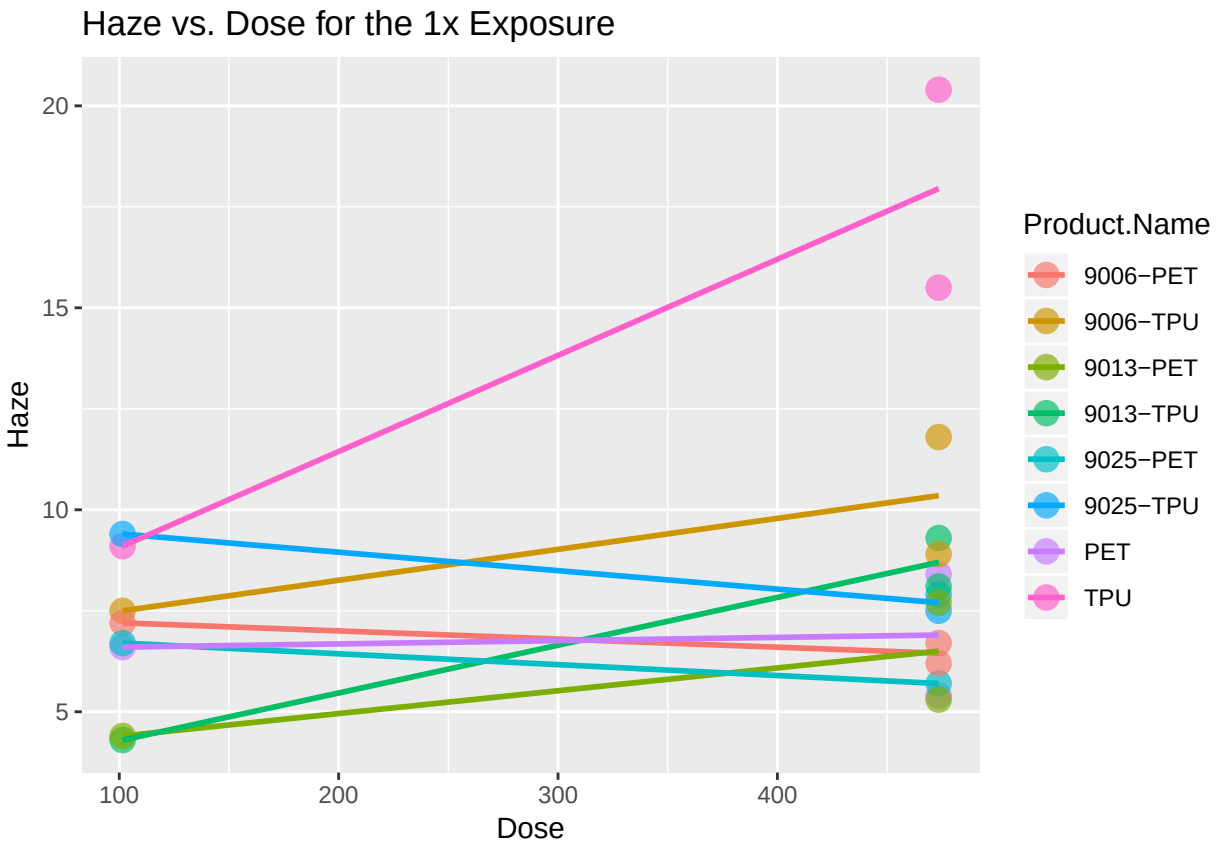
```
##             ID   YI Haze Step   peak1250  peak1700   peak2900    peak3350
## 245 sa22075.04 2.73  4.9    4 0.05447078 0.1008560 0.01678111 0.008075400
## 246 sa22080.15 2.47  7.3    4 0.08053469 0.1568958 0.03119338 0.010186100
## 247 sa22080.14 3.25  6.2    4 0.04948241 0.1104132 0.02067752 0.008409085
## 248 sa22086.14 1.96  2.9    4 0.52833679 0.5591786 0.02674627 0.012094311
## 249 sa22087.00 2.05  4.2    4 0.47992823 0.5061842 0.02279625 0.009594556
## 250 sa22069.05 3.24  5.9    4 0.08727721 0.2784521 0.05399525 0.018422963
##     Product.Name Exposure Dose
## 245     9013-PET       HF    0
## 246     9025-PET       HF    0
## 247     9025-PET       HF    0
## 248          PET       HF    0
## 249          PET       HF    0
## 250     9006-PET       HF    0
```

3) The plots are shown below

```
ggplot(finaldata%>%filter(Exposure == "1x")) +
  geom_point(aes(x=Dose,y=YI, colour = Product.Name), size = 4, alpha = 2/3) +
  geom_smooth(method = lm, aes(x = Dose, y = YI, colour = Product.Name), se = FALSE) +
  ggtitle("YI vs. Dose for the 1x Exposure")
```




```
ggplot(finaldata%>%filter(Exposure == "1x")) +
  geom_point(aes(x=Dose,y=Haze, colour = Product.Name), size = 4, alpha = 2/3) +
  geom_smooth(method = lm, aes(x = Dose, y = Haze, colour = Product.Name), se = FALSE) +
  ggtitle("Haze vs. Dose for the 1x Exposure")
```



#Part 2-netSEM(did not have time to make code work on this part) 1. In structural equation modelining (SEM), a combination of factor analysis and regression is used to analyze how multiple factors correlate. SEM follows a path. Path diagrams like the one shown in the figure explain how measured variables as well as latent factors can have arrows pointing to different factors indicating causal relationships and so on. In figure 1 and in this project, our stressor was irradiance. There were multiple modes of that which irradiance followed. HF and the baseline exposures providedno irradiance exposure to their samples, but ASTMG155, ASTMG154, mASTMG154, 1x, and 5x all consisted of either labroatory conditions or natural conditions for irradiance. The responses were YI or the yellowness index. 2. I thought mASTMG154 would be the closest to 1x degradation. I thought this because mASTMG154 follows a cyclical period of both light and night periods that resemble a typical day a lot more. mASTMG154 has 8 hours of light and 4 hours of darkness and also the intensity of the mASTMG154 is not very powerful and this makes the ASTMG155 even more natural.

```
model <- finaldata %>%
dplyr::filter(Exposure == '1x' ) %>%
  dplyr::select((c( Dose , YI , peak1250 , peak1700 , peak2900 , peak3350 ))) %>%
netSEMm()
```

```
## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated
```

```
## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated
```

```
model2 <- finaldata %>%
dplyr::filter(Exposure == 'mASTMG154' ) %>%
  dplyr::select(((c( Dose , YI , peak1250 , peak1700 , peak2900 ,  peak3350 )))) %>%
netSEMm()
```

```
## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: max number of iterations attained

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated

## Warning: No breakpoint estimated
```