Vent Gate Project: Data Preparation Run Book

M Candia & P Candia

August 20, 2022

## Objective

Determine whether there is a causal relationship between the vent gates openness and Seeing measurments. Measure the effect of vent gates openness on the Seeing, accounting for wind conditions and other relevant factors. Given wind conditions, determine the vengate openness for optimal Seeing (Minimum).

## Background

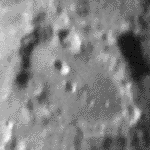
The telescope has two vent gates (East, West) on the side of the dome. Each vent gate can open about 160 degrees. The vent gates open in percentage settings tipically: 10%, 30% …100%. The telsecope operator decides how much to open each vent gate; each vent gate may be opened independently.



*Telescope dome vent gates, one of the vents is not vissible because it is in the oposite side.*

*“*[*Astronomical seeing*](https://www.innovationsforesight.com/education/astronomical-seeing-tutorial/) *refers to the blurring of astronomical objects caused by the Earth atmospheric turbulence. Thermal convection and winds produce turbulence cells having different optical refraction indexes, leading to perturbations and distortions of the incoming light wavefronts. It is worth noting that air motion and associated turbulence do not impact the light path itself, unless there is a change in refraction index, typically coming from temperature differences across the air mass involved in the process.”*

The seeing varies between 0.3 and 2.0 arc seconds approximately. A seeing of 0.3 is very good (stable and sharp image); in contrast, aseeing of 2.0 arc seconds is very bad (unstable and out of focus [blurry] image). For example this image of the Moon has high seeing measurement. [see example](https://upload.wikimedia.org/wikipedia/commons/e/ef/Seeing_Moon.gif)



Moon image deteriorated because of the atmospheric turbulence.

## Vent Gate Raw Data

Upload Vent Gate raw data (for April, 2022) into R after manually removing header text, then check the structure and print a few records, and then save the data frame as *df202204a.Rdata*. Note that we extracted the tab delimited records first, this is because of the format the data was provided.

df202204a = read.delim("D:/PROJECTS/R\_Projects/Vent\_Gates\_Proj/VentGateProject-main/Datos\_mes\_edit\_1/Vent\_Gate\_Project/apr1\_30\_bothVent\_1.txt", header=FALSE, sep="\t")  
  
str(df202204a)

## 'data.frame': 589256 obs. of 5 variables:  
## $ V1: chr "04/01/2022 16:10:16.768737641" "04/01/2022 17:59:59.362571548" "04/01/2022 18:00:04.762571548" "04/01/2022 18:00:05.762571548" ...  
## $ V2: chr "#N/A" "0.10638298" "0.12765957" "0.10638298" ...  
## $ V3: chr "<no data>" "" "" "" ...  
## $ V4: chr "0.23404255" "0.23404255" "0.23404255" "0.23404255" ...  
## $ V5: chr "" "" "" "" ...

head(df202204a)

## V1 V2 V3 V4 V5  
## 1 04/01/2022 16:10:16.768737641 #N/A <no data> 0.23404255   
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255   
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255   
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255   
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255   
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255

save(df202204a, file="df202204a.Rdata")

Keep only columns *V1, V2, V4* and then check structure, print a few records, and save as *df202204b.Rdata*.

df202204b = df202204a[, c('V1','V2','V4') ]  
  
str(df202204b)

## 'data.frame': 589256 obs. of 3 variables:  
## $ V1: chr "04/01/2022 16:10:16.768737641" "04/01/2022 17:59:59.362571548" "04/01/2022 18:00:04.762571548" "04/01/2022 18:00:05.762571548" ...  
## $ V2: chr "#N/A" "0.10638298" "0.12765957" "0.10638298" ...  
## $ V4: chr "0.23404255" "0.23404255" "0.23404255" "0.23404255" ...

head(df202204b)

## V1 V2 V4  
## 1 04/01/2022 16:10:16.768737641 #N/A 0.23404255  
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255  
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255  
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255  
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255  
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255

save(df202204b, file="df202204b.Rdata")

Note that all fields are character fields. Furthermore,inspection of the data shows some rows with *“#N/A”* values for *V2* and *V4*. The *“#N/A”* values were replaced by R missing values, and then a new data frame was produced and saved as *df202204c.RData*.

library(naniar) #package to assign missing values to given values  
  
df202204c <- df202204b %>% replace\_with\_na(replace = list( V2="#N/A", V4="#N/A"))   
  
str(df202204c)

## 'data.frame': 589256 obs. of 3 variables:  
## $ V1: chr "04/01/2022 16:10:16.768737641" "04/01/2022 17:59:59.362571548" "04/01/2022 18:00:04.762571548" "04/01/2022 18:00:05.762571548" ...  
## $ V2: chr NA "0.10638298" "0.12765957" "0.10638298" ...  
## $ V4: chr "0.23404255" "0.23404255" "0.23404255" "0.23404255" ...

head(df202204c)

## V1 V2 V4  
## 1 04/01/2022 16:10:16.768737641 <NA> 0.23404255  
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255  
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255  
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255  
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255  
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255

save(df202204c, file="df202204c.RData")

Convert *V2* and *V4* to numeric (*V2\_num* and *V4\_num*), and then round to nearest integer (*V2\_nrnd* and *V4\_nrnd*).

library(dplyr) #package to add mutate fuction  
  
df202204c\_num=mutate(df202204c,V2\_num=as.numeric(df202204c$V2), V4\_num=as.numeric(df202204c$V4))  
df202204c\_nrnd=mutate( df202204c\_num, V2\_rnd=round(df202204c\_num$V2\_num), V4\_rnd=round(df202204c\_num$V4\_num))  
str(df202204c\_nrnd)

## 'data.frame': 589256 obs. of 7 variables:  
## $ V1 : chr "04/01/2022 16:10:16.768737641" "04/01/2022 17:59:59.362571548" "04/01/2022 18:00:04.762571548" "04/01/2022 18:00:05.762571548" ...  
## $ V2 : chr NA "0.10638298" "0.12765957" "0.10638298" ...  
## $ V4 : chr "0.23404255" "0.23404255" "0.23404255" "0.23404255" ...  
## $ V2\_num: num NA 0.106 0.128 0.106 0.128 ...  
## $ V4\_num: num 0.234 0.234 0.234 0.234 0.234 ...  
## $ V2\_rnd: num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd: num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204c\_nrnd)

## V1 V2 V4 V2\_num V4\_num  
## 1 04/01/2022 16:10:16.768737641 <NA> 0.23404255 NA 0.2340425  
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## V2\_rnd V4\_rnd  
## 1 NA 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0

Split date-time field *V1* into *V1\_date* and *V1\_time* fields.

library("stringr")  
df202204c\_nrnd$V1\_date=substr(df202204c\_nrnd$V1,1,10)  
df202204c\_nrnd$V1\_time=substr(df202204c\_nrnd$V1,11,23)  
str(df202204c\_nrnd)

## 'data.frame': 589256 obs. of 9 variables:  
## $ V1 : chr "04/01/2022 16:10:16.768737641" "04/01/2022 17:59:59.362571548" "04/01/2022 18:00:04.762571548" "04/01/2022 18:00:05.762571548" ...  
## $ V2 : chr NA "0.10638298" "0.12765957" "0.10638298" ...  
## $ V4 : chr "0.23404255" "0.23404255" "0.23404255" "0.23404255" ...  
## $ V2\_num : num NA 0.106 0.128 0.106 0.128 ...  
## $ V4\_num : num 0.234 0.234 0.234 0.234 0.234 ...  
## $ V2\_rnd : num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date: chr "04/01/2022" "04/01/2022" "04/01/2022" "04/01/2022" ...  
## $ V1\_time: chr " 16:10:16.768" " 17:59:59.362" " 18:00:04.762" " 18:00:05.762" ...

head(df202204c\_nrnd)

## V1 V2 V4 V2\_num V4\_num  
## 1 04/01/2022 16:10:16.768737641 <NA> 0.23404255 NA 0.2340425  
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## V2\_rnd V4\_rnd V1\_date V1\_time  
## 1 NA 0 04/01/2022 16:10:16.768  
## 2 0 0 04/01/2022 17:59:59.362  
## 3 0 0 04/01/2022 18:00:04.762  
## 4 0 0 04/01/2022 18:00:05.762  
## 5 0 0 04/01/2022 18:00:09.762  
## 6 0 0 04/01/2022 18:00:12.362

Convert date-time field *V1* into a date-time field *V1\_date\_time* with POSIXct format. The numeric value of *V1\_date\_time* corresponds to the number of seconds since 1970. Then save data frame as *df202204c\_nrnd.RData*.

op <- options(digits.sec=9)  
df202204c\_nrnd$V1\_date\_time=as.POSIXct(df202204c\_nrnd$V1, format="%m/%d/%Y %H:%M:%OS")  
class(df202204c\_nrnd$V1\_date\_time)

## [1] "POSIXct" "POSIXt"

str(df202204c\_nrnd$V1\_date\_time)

## POSIXct[1:589256], format: "2022-04-01 16:10:16" "2022-04-01 17:59:59" "2022-04-01 18:00:04" ...

head(unclass(df202204c\_nrnd$V1\_date\_time)) # Number of seconds since 1970 for first 6 records

## [1] 1648854617 1648861199 1648861205 1648861206 1648861210 1648861212

head(df202204c\_nrnd$V1\_date\_time) # POSIXct format for first 6 records

## [1] "2022-04-01 16:10:16 PDT" "2022-04-01 17:59:59 PDT"  
## [3] "2022-04-01 18:00:04 PDT" "2022-04-01 18:00:05 PDT"  
## [5] "2022-04-01 18:00:09 PDT" "2022-04-01 18:00:12 PDT"

head(df202204c\_nrnd)

## V1 V2 V4 V2\_num V4\_num  
## 1 04/01/2022 16:10:16.768737641 <NA> 0.23404255 NA 0.2340425  
## 2 04/01/2022 17:59:59.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 3 04/01/2022 18:00:04.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 4 04/01/2022 18:00:05.762571548 0.10638298 0.23404255 0.1063830 0.2340425  
## 5 04/01/2022 18:00:09.762571548 0.12765957 0.23404255 0.1276596 0.2340425  
## 6 04/01/2022 18:00:12.362571548 0.10638298 0.23404255 0.1063830 0.2340425  
## V2\_rnd V4\_rnd V1\_date V1\_time V1\_date\_time  
## 1 NA 0 04/01/2022 16:10:16.768 2022-04-01 16:10:16  
## 2 0 0 04/01/2022 17:59:59.362 2022-04-01 17:59:59  
## 3 0 0 04/01/2022 18:00:04.762 2022-04-01 18:00:04  
## 4 0 0 04/01/2022 18:00:05.762 2022-04-01 18:00:05  
## 5 0 0 04/01/2022 18:00:09.762 2022-04-01 18:00:09  
## 6 0 0 04/01/2022 18:00:12.362 2022-04-01 18:00:12

save(df202204c\_nrnd, file="df202204c\_nrnd.RData")  
options(op)

Prepare the data for concatenation with the seeing data by adding a column to align with the seeing measured by the P2 detector: a) Create Seeing indicator and set to zero; b) order columns.

#Prepare Vent-Gate data for concatenation  
df202204d=subset(df202204c\_nrnd, select=c(V2\_rnd,V4\_rnd, V1\_date,V1\_date\_time))  
df202204e=mutate(df202204d, P2\_ind=0)  
  
#Reorder columns  
df202204f=df202204e[, c("P2\_ind","V1\_date", "V1\_date\_time", "V2\_rnd", "V4\_rnd")]  
str(df202204f)

## 'data.frame': 589256 obs. of 5 variables:  
## $ P2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date : chr "04/01/2022" "04/01/2022" "04/01/2022" "04/01/2022" ...  
## $ V1\_date\_time: POSIXct, format: "2022-04-01 16:10:16" "2022-04-01 17:59:59" ...  
## $ V2\_rnd : num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd : num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204f)

## P2\_ind V1\_date V1\_date\_time V2\_rnd V4\_rnd  
## 1 0 04/01/2022 2022-04-01 16:10:16 NA 0  
## 2 0 04/01/2022 2022-04-01 17:59:59 0 0  
## 3 0 04/01/2022 2022-04-01 18:00:04 0 0  
## 4 0 04/01/2022 2022-04-01 18:00:05 0 0  
## 5 0 04/01/2022 2022-04-01 18:00:09 0 0  
## 6 0 04/01/2022 2022-04-01 18:00:12 0 0

save(df202204f, file="df202204f.RData")  
  
df202204f\_VG <- df202204f  
  
head(df202204f\_VG)

## P2\_ind V1\_date V1\_date\_time V2\_rnd V4\_rnd  
## 1 0 04/01/2022 2022-04-01 16:10:16 NA 0  
## 2 0 04/01/2022 2022-04-01 17:59:59 0 0  
## 3 0 04/01/2022 2022-04-01 18:00:04 0 0  
## 4 0 04/01/2022 2022-04-01 18:00:05 0 0  
## 5 0 04/01/2022 2022-04-01 18:00:09 0 0  
## 6 0 04/01/2022 2022-04-01 18:00:12 0 0

Rename V2\_rnd to V2\_rnd\_VG\_East and V4\_rnd to V4\_rnd\_VG\_West.

#Rename V2\_rnd to V2\_rnd\_VG\_East and V4\_rnd to V4\_rnd\_VG\_West  
  
library("tidyverse")  
df202204g\_VG = rename(df202204f\_VG, c("V2\_rnd\_VG\_East" = "V2\_rnd", "V4\_rnd\_VG\_West" = "V4\_rnd"))

str(df202204g\_VG)

## 'data.frame': 589256 obs. of 5 variables:  
## $ P2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date : chr "04/01/2022" "04/01/2022" "04/01/2022" "04/01/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-01 16:10:16" "2022-04-01 17:59:59" ...  
## $ V2\_rnd\_VG\_East: num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd\_VG\_West: num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204g\_VG, n = 6L)

## P2\_ind V1\_date V1\_date\_time V2\_rnd\_VG\_East V4\_rnd\_VG\_West  
## 1 0 04/01/2022 2022-04-01 16:10:16 NA 0  
## 2 0 04/01/2022 2022-04-01 17:59:59 0 0  
## 3 0 04/01/2022 2022-04-01 18:00:04 0 0  
## 4 0 04/01/2022 2022-04-01 18:00:05 0 0  
## 5 0 04/01/2022 2022-04-01 18:00:09 0 0  
## 6 0 04/01/2022 2022-04-01 18:00:12 0 0

save(df202204g\_VG, file = "df202204g\_VG.RData")

## Seeing Raw Data

Upload seeing raw data measured by the detector P2 (for April, 2022) into R after manually removing header text, then check the structure and print a few records, and then save the data frame as *df202204P2a.Rdata*. Note that we extracted the tab delimited records first, this is because of the format the data was provided.

#read source text data  
df202204P2a = read.delim("D:/PROJECTS/R\_Projects/Vent\_Gates\_Proj/VentGateProject-main/Datos\_mes\_edit\_1/Vent\_Gate\_Project/P2seeing\_apr1\_31\_1.txt", header=FALSE, sep="\t")  
  
#Check structure and View  
str(df202204P2a)

## 'data.frame': 8467 obs. of 3 variables:  
## $ V1: chr "04/01/2022 12:02:40.106355905" "04/01/2022 20:21:13.980065107" "04/01/2022 20:21:44.143955945" "04/02/2022 07:06:32.931230560" ...  
## $ V2: chr "6.8260944" "0.48289563" "0.49089873" "#N/A" ...  
## $ V3: chr "" "" "" "Disconnected" ...

head(df202204P2a)

## V1 V2 V3  
## 1 04/01/2022 12:02:40.106355905 6.8260944   
## 2 04/01/2022 20:21:13.980065107 0.48289563   
## 3 04/01/2022 20:21:44.143955945 0.49089873   
## 4 04/02/2022 07:06:32.931230560 #N/A Disconnected  
## 5 04/02/2022 07:06:32.931239067 #N/A Archive\_Off  
## 6 04/02/2022 07:06:41.583477096 0.49089873

#Save a copy of dataframe  
save(df202204P2a, file="df202204P2a.Rdata")

Keep the first two columns.

#Keep first 2 columns  
df202204P2b = df202204P2a[, c('V1','V2') ]  
str(df202204P2b)

## 'data.frame': 8467 obs. of 2 variables:  
## $ V1: chr "04/01/2022 12:02:40.106355905" "04/01/2022 20:21:13.980065107" "04/01/2022 20:21:44.143955945" "04/02/2022 07:06:32.931230560" ...  
## $ V2: chr "6.8260944" "0.48289563" "0.49089873" "#N/A" ...

head(df202204P2b)

## V1 V2  
## 1 04/01/2022 12:02:40.106355905 6.8260944  
## 2 04/01/2022 20:21:13.980065107 0.48289563  
## 3 04/01/2022 20:21:44.143955945 0.49089873  
## 4 04/02/2022 07:06:32.931230560 #N/A  
## 5 04/02/2022 07:06:32.931239067 #N/A  
## 6 04/02/2022 07:06:41.583477096 0.49089873

Note that all fields are character fields. Furthermore, inspection of the data shows some rows with *“#N/A”* values for *V2*. The *“#N/A”* values were replaced by R missing values, and then a new data frame was produced and saved as *df202204P2c.RData*.

df202204P2c <- df202204P2b %>% replace\_with\_na(replace = list( V2="#N/A"))  
   
str(df202204P2c)

## 'data.frame': 8467 obs. of 2 variables:  
## $ V1: chr "04/01/2022 12:02:40.106355905" "04/01/2022 20:21:13.980065107" "04/01/2022 20:21:44.143955945" "04/02/2022 07:06:32.931230560" ...  
## $ V2: chr "6.8260944" "0.48289563" "0.49089873" NA ...

head(df202204P2c)

## V1 V2  
## 1 04/01/2022 12:02:40.106355905 6.8260944  
## 2 04/01/2022 20:21:13.980065107 0.48289563  
## 3 04/01/2022 20:21:44.143955945 0.49089873  
## 4 04/02/2022 07:06:32.931230560 <NA>  
## 5 04/02/2022 07:06:32.931239067 <NA>  
## 6 04/02/2022 07:06:41.583477096 0.49089873

save(df202204P2c, file="df202204P2c.RData")

Convert *V2* to numeric (*V2\_seeing\_num*), and then round to nearest integer (*V2\_seeing\_nrnd*).

df202204P2d=mutate(df202204P2c,V2\_seeing\_num=as.numeric(df202204P2c$V2))  
df202204P2e=mutate(df202204P2d, V2\_seeing\_rnd=round(df202204P2d$V2\_seeing\_num, digits=3))  
str(df202204P2e)

## 'data.frame': 8467 obs. of 4 variables:  
## $ V1 : chr "04/01/2022 12:02:40.106355905" "04/01/2022 20:21:13.980065107" "04/01/2022 20:21:44.143955945" "04/02/2022 07:06:32.931230560" ...  
## $ V2 : chr "6.8260944" "0.48289563" "0.49089873" NA ...  
## $ V2\_seeing\_num: num 6.826 0.483 0.491 NA NA ...  
## $ V2\_seeing\_rnd: num 6.826 0.483 0.491 NA NA ...

head(df202204P2e)

## V1 V2 V2\_seeing\_num V2\_seeing\_rnd  
## 1 04/01/2022 12:02:40.106355905 6.8260944 6.8260944 6.826  
## 2 04/01/2022 20:21:13.980065107 0.48289563 0.4828956 0.483  
## 3 04/01/2022 20:21:44.143955945 0.49089873 0.4908987 0.491  
## 4 04/02/2022 07:06:32.931230560 <NA> NA NA  
## 5 04/02/2022 07:06:32.931239067 <NA> NA NA  
## 6 04/02/2022 07:06:41.583477096 0.49089873 0.4908987 0.491

Convert date-time field *V1* into a date-time field *V1\_date\_time* with POSIXct format. The numeric value of *V1\_date\_time* corresponds to the number of seconds since 1970. Then save data frame as *df202204c\_nrnd.RData*.

df202204P2e$V1\_date=substr(df202204P2e$V1,1,10)  
df202204P2e$V1\_time=substr(df202204P2e$V1,11,23)   
df202204P2e$V1\_date\_time=as.POSIXct(df202204P2e$V1, format="%m/%d/%Y %H:%M:%OS")  
  
str(df202204P2e$V1\_date\_time)

## POSIXct[1:8467], format: "2022-04-01 12:02:40" "2022-04-01 20:21:13" "2022-04-01 20:21:44" ...

head(unclass(df202204P2e$V1\_date\_time))

## [1] 1648839760 1648869674 1648869704 1648908393 1648908393 1648908402

head(df202204P2e$V1\_date\_time)

## [1] "2022-04-01 12:02:40 PDT" "2022-04-01 20:21:13 PDT"  
## [3] "2022-04-01 20:21:44 PDT" "2022-04-02 07:06:32 PDT"  
## [5] "2022-04-02 07:06:32 PDT" "2022-04-02 07:06:41 PDT"

Flag dates and date-times when the detector P2 was active, and then keep those records for which P2 was active.

#P2-active date selection indicator  
df202204P2e$Date\_in1=(df202204P2e$V1\_date %in% c("04/02/2022",  
 "04/04/2022",  
 "04/05/2022",  
 "04/06/2022",  
 "04/07/2022",  
 "04/08/2022",  
 "04/10/2022",  
 "04/11/2022",  
 "04/16/2022",  
 "04/18/2022",  
 "04/19/2022",  
 "04/20/2022",  
 "04/21/2022",  
 "04/22/2022",  
 "04/23/2022",  
 "04/24/2022",  
 "04/25/2022",  
 "04/26/2022",  
 "04/27/2022",  
 "04/28/2022",  
 "04/29/2022",  
 "04/30/2022"))  
  
#P2-active date-time selection indicator  
#First time period for which P2 was active  
  
df202204P2e$Datetime\_in1 = (as.numeric(as.POSIXct("2022-04-02 23:01:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-02 23:34:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-04 23:51:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-04 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-05 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-05 2:56:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-06 03:38:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-06 05:05:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-07 02:44:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-07 03:31:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-08 03:32:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-08 05:04:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-10 05:25:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-10 06:01:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-11 00:16:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-11 00:56:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-16 02:07:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-16 02:25:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-18 02:31:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-18 02:42:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-19 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-19 03:58:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-20 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-20 06:23:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-21 00:25:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-21 06:17:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-22 19:30:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-22 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-23 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-23 06:41:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-24 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-24 01:57:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-25 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-25 05:31:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-26 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-26 05:58:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-27 01:11:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-27 02:53:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-28 00:00:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-28 06:19:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-29 00:04:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-29 03:21:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-30 00:15:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-30 04:42:59.999"))) +  
   
 #Second time period for which P2 was active  
   
 (as.numeric(as.POSIXct("2022-04-05 04:32:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-05 04:50:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-07 05:15:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-07 06:07:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-18 19:57:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-18 20:13:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-19 19:30:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-19 19:46:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-20 19:30:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-20 20:54:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-23 19:30:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-23 21:09:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-24 03:42:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-24 06:12:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-25 20:29:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-25 21:14:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-27 21:19:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-27 23:59:59.999"))) +   
 (as.numeric(as.POSIXct("2022-04-30 19:30:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-30 23:59:59.999"))) +   
   
 #Third time period for which P2 was active  
   
 (as.numeric(as.POSIXct("2022-04-18 23:10:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-18 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-19 21:27:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-19 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-23 22:57:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-23 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-24 20:41:00")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-24 23:59:59.999"))) +  
 (as.numeric(as.POSIXct("2022-04-25 23:12:59")) <= df202204P2e$V1\_date\_time & df202204P2e$V1\_date\_time < as.numeric(as.POSIXct("2022-04-25 999-23:59:59")))   
   
  
#Filter valid P2 records  
df202204P2f =df202204P2e[df202204P2e$Datetime\_in1 > 0,]  
  
str(df202204P2f)

## 'data.frame': 7755 obs. of 9 variables:  
## $ V1 : chr "04/02/2022 23:01:14.130250930" "04/02/2022 23:01:44.294039011" "04/02/2022 23:02:14.458241939" "04/02/2022 23:02:44.622322082" ...  
## $ V2 : chr "0.39821874" "0.44567433" "0.42914129" "0.41808554" ...  
## $ V2\_seeing\_num: num 0.398 0.446 0.429 0.418 0.339 ...  
## $ V2\_seeing\_rnd: num 0.398 0.446 0.429 0.418 0.339 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_time : chr " 23:01:14.130" " 23:01:44.294" " 23:02:14.458" " 23:02:44.622" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ Date\_in1 : logi TRUE TRUE TRUE TRUE TRUE TRUE ...  
## $ Datetime\_in1 : int 1 1 1 1 1 1 1 1 1 1 ...

head(df202204P2f)

## V1 V2 V2\_seeing\_num V2\_seeing\_rnd  
## 17 04/02/2022 23:01:14.130250930 0.39821874 0.3982187 0.398  
## 18 04/02/2022 23:01:44.294039011 0.44567433 0.4456743 0.446  
## 19 04/02/2022 23:02:14.458241939 0.42914129 0.4291413 0.429  
## 20 04/02/2022 23:02:44.622322082 0.41808554 0.4180855 0.418  
## 21 04/02/2022 23:03:14.786847114 0.33854191 0.3385419 0.339  
## 22 04/02/2022 23:03:44.955780029 0.58897957 0.5889796 0.589  
## V1\_date V1\_time V1\_date\_time Date\_in1 Datetime\_in1  
## 17 04/02/2022 23:01:14.130 2022-04-02 23:01:14 TRUE 1  
## 18 04/02/2022 23:01:44.294 2022-04-02 23:01:44 TRUE 1  
## 19 04/02/2022 23:02:14.458 2022-04-02 23:02:14 TRUE 1  
## 20 04/02/2022 23:02:44.622 2022-04-02 23:02:44 TRUE 1  
## 21 04/02/2022 23:03:14.786 2022-04-02 23:03:14 TRUE 1  
## 22 04/02/2022 23:03:44.955 2022-04-02 23:03:44 TRUE 1

#Save a copy of dataframe  
save(df202204P2f, file="df202204P2f.Rdata")

Prepare the data for concatenation with the vent gate data: a) Create Seeing indicator and set to one; b) select and order columns.

df202204P2g=subset(df202204P2f, select=c(V2\_seeing\_rnd, V1\_date,V1\_date\_time))  
df202204P2h=mutate(df202204P2g, P2\_ind=1)  
df202204P2i=df202204P2h[, c("P2\_ind","V1\_date", "V1\_date\_time", "V2\_seeing\_rnd")]  
str(df202204P2i)

## 'data.frame': 7755 obs. of 4 variables:  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V2\_seeing\_rnd: num 0.398 0.446 0.429 0.418 0.339 ...

head(df202204P2i)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd  
## 1 1 04/02/2022 2022-04-02 23:01:14 0.398  
## 2 1 04/02/2022 2022-04-02 23:01:44 0.446  
## 3 1 04/02/2022 2022-04-02 23:02:14 0.429  
## 4 1 04/02/2022 2022-04-02 23:02:44 0.418  
## 5 1 04/02/2022 2022-04-02 23:03:14 0.339  
## 6 1 04/02/2022 2022-04-02 23:03:44 0.589

#Save a copy of dataframe  
save(df202204P2i, file="df202204P2i.Rdata")

## Merging Vent Gate and Seeing Data

Concatenate *df202204g\_VG* and *df202204P2i* data frames and sort by V1\_date\_time\_sorted.

#Concatenate april2022P2\_sub2 and april2022VG\_sub2  
df202204\_VG\_P2\_a <-bind\_rows(df202204P2i,df202204g\_VG)  
df202204\_VG\_P2\_b =mutate(df202204\_VG\_P2\_a,V1\_date\_time\_num=as.numeric(df202204\_VG\_P2\_a$V1\_date\_time))  
V1\_date\_time\_sorted <- order(df202204\_VG\_P2\_b$V1\_date\_time\_num)  
df202204\_VG\_P2\_c <- df202204\_VG\_P2\_b[V1\_date\_time\_sorted, ]  
str(df202204\_VG\_P2\_c)

## 'data.frame': 597011 obs. of 7 variables:  
## $ P2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date : chr "04/01/2022" "04/01/2022" "04/01/2022" "04/01/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-01 16:10:16" "2022-04-01 17:59:59" ...  
## $ V2\_seeing\_rnd : num NA NA NA NA NA NA NA NA NA NA ...  
## $ V2\_rnd\_VG\_East : num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd\_VG\_West : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...

head(df202204\_VG\_P2\_c[df202204\_VG\_P2\_c$V1\_date\_time\_num > 1651308780,]) #Print a few records with active

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V2\_rnd\_VG\_East  
## 592124 0 04/30/2022 2022-04-30 01:53:00 NA 80  
## 7473 1 04/30/2022 2022-04-30 01:53:13 0.841 NA  
## 7474 1 04/30/2022 2022-04-30 01:53:43 0.872 NA  
## 7475 1 04/30/2022 2022-04-30 01:54:13 1.349 NA  
## 592125 0 04/30/2022 2022-04-30 01:54:25 NA 80  
## 592126 0 04/30/2022 2022-04-30 01:54:25 NA 80  
## V4\_rnd\_VG\_West V1\_date\_time\_num  
## 592124 5 1651308780  
## 7473 NA 1651308793  
## 7474 NA 1651308823  
## 7475 NA 1651308854  
## 592125 5 1651308865  
## 592126 5 1651308866

head(df202204\_VG\_P2\_c)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V2\_rnd\_VG\_East  
## 7756 0 04/01/2022 2022-04-01 16:10:16 NA NA  
## 7757 0 04/01/2022 2022-04-01 17:59:59 NA 0  
## 7758 0 04/01/2022 2022-04-01 18:00:04 NA 0  
## 7759 0 04/01/2022 2022-04-01 18:00:05 NA 0  
## 7760 0 04/01/2022 2022-04-01 18:00:09 NA 0  
## 7761 0 04/01/2022 2022-04-01 18:00:12 NA 0  
## V4\_rnd\_VG\_West V1\_date\_time\_num  
## 7756 0 1648854617  
## 7757 0 1648861199  
## 7758 0 1648861205  
## 7759 0 1648861206  
## 7760 0 1648861210  
## 7761 0 1648861212

save(df202204\_VG\_P2\_c, file = "df202204\_VG\_P2\_c.RData")

Add lag variables for the vent gate fields and P2\_ind.

df202204\_VG\_P2\_d = mutate(df202204\_VG\_P2\_c, lagvg\_east=lag(V2\_rnd\_VG\_East), lagvg\_west=lag(V4\_rnd\_VG\_West), lagp2\_ind=lag(P2\_ind))  
str(df202204\_VG\_P2\_d)

## 'data.frame': 597011 obs. of 10 variables:  
## $ P2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date : chr "04/01/2022" "04/01/2022" "04/01/2022" "04/01/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-01 16:10:16" "2022-04-01 17:59:59" ...  
## $ V2\_seeing\_rnd : num NA NA NA NA NA NA NA NA NA NA ...  
## $ V2\_rnd\_VG\_East : num NA 0 0 0 0 0 0 0 0 0 ...  
## $ V4\_rnd\_VG\_West : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num NA NA 0 0 0 0 0 0 0 0 ...  
## $ lagvg\_west : num NA 0 0 0 0 0 0 0 0 0 ...  
## $ lagp2\_ind : num NA 0 0 0 0 0 0 0 0 0 ...

head(df202204\_VG\_P2\_d[df202204\_VG\_P2\_d$V1\_date\_time\_num > 1651308780,]) #Print a few records which include P2 active

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V2\_rnd\_VG\_East  
## 591841 0 04/30/2022 2022-04-30 01:53:00 NA 80  
## 591842 1 04/30/2022 2022-04-30 01:53:13 0.841 NA  
## 591843 1 04/30/2022 2022-04-30 01:53:43 0.872 NA  
## 591844 1 04/30/2022 2022-04-30 01:54:13 1.349 NA  
## 591845 0 04/30/2022 2022-04-30 01:54:25 NA 80  
## 591846 0 04/30/2022 2022-04-30 01:54:25 NA 80  
## V4\_rnd\_VG\_West V1\_date\_time\_num lagvg\_east lagvg\_west lagp2\_ind  
## 591841 5 1651308780 80 5 0  
## 591842 NA 1651308793 80 5 0  
## 591843 NA 1651308823 NA NA 1  
## 591844 NA 1651308854 NA NA 1  
## 591845 5 1651308865 NA NA 1  
## 591846 5 1651308866 80 5 0

save(df202204\_VG\_P2\_d, file = "df202204\_VG\_P2\_d.RData")

Keep only valid P2 records

#Filter valid P2 records  
df202204\_VG\_P2\_e =df202204\_VG\_P2\_d[df202204\_VG\_P2\_d$P2\_ind > 0,]  
str(df202204\_VG\_P2\_e)

## 'data.frame': 7755 obs. of 10 variables:  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V2\_seeing\_rnd : num 0.398 0.446 0.429 0.418 0.339 ...  
## $ V2\_rnd\_VG\_East : num NA NA NA NA NA NA NA NA NA NA ...  
## $ V4\_rnd\_VG\_West : num NA NA NA NA NA NA NA NA NA NA ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagp2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204\_VG\_P2\_e)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V2\_rnd\_VG\_East  
## 40409 1 04/02/2022 2022-04-02 23:01:14 0.398 NA  
## 40418 1 04/02/2022 2022-04-02 23:01:44 0.446 NA  
## 40431 1 04/02/2022 2022-04-02 23:02:14 0.429 NA  
## 40446 1 04/02/2022 2022-04-02 23:02:44 0.418 NA  
## 40463 1 04/02/2022 2022-04-02 23:03:14 0.339 NA  
## 40472 1 04/02/2022 2022-04-02 23:03:44 0.589 NA  
## V4\_rnd\_VG\_West V1\_date\_time\_num lagvg\_east lagvg\_west lagp2\_ind  
## 40409 NA 1648965674 30 30 0  
## 40418 NA 1648965704 30 30 0  
## 40431 NA 1648965734 30 30 0  
## 40446 NA 1648965765 30 30 0  
## 40463 NA 1648965795 30 30 0  
## 40472 NA 1648965825 30 30 0

Keep selected fields.

df202204\_VG\_P2\_f=subset(df202204\_VG\_P2\_e, select=c(P2\_ind, V1\_date, V1\_date\_time, V2\_seeing\_rnd, V1\_date\_time\_num,lagvg\_east, lagvg\_west, lagp2\_ind))  
head(df202204\_VG\_P2\_f)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V1\_date\_time\_num  
## 40409 1 04/02/2022 2022-04-02 23:01:14 0.398 1648965674  
## 40418 1 04/02/2022 2022-04-02 23:01:44 0.446 1648965704  
## 40431 1 04/02/2022 2022-04-02 23:02:14 0.429 1648965734  
## 40446 1 04/02/2022 2022-04-02 23:02:44 0.418 1648965765  
## 40463 1 04/02/2022 2022-04-02 23:03:14 0.339 1648965795  
## 40472 1 04/02/2022 2022-04-02 23:03:44 0.589 1648965825  
## lagvg\_east lagvg\_west lagp2\_ind  
## 40409 30 30 0  
## 40418 30 30 0  
## 40431 30 30 0  
## 40446 30 30 0  
## 40463 30 30 0  
## 40472 30 30 0

Create additional flags.

df202204\_VG\_P2\_g=mutate(df202204\_VG\_P2\_f,lead1=lead(lagp2\_ind), lpl\_flag1=as.numeric(as.logical(lagp2\_ind + lead1)) )  
df202204\_VG\_P2\_h=mutate(df202204\_VG\_P2\_g, upchange=ifelse((lagp2\_ind==0 & lead1==1),1,0))  
str(df202204\_VG\_P2\_g)

## 'data.frame': 7755 obs. of 10 variables:  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V2\_seeing\_rnd : num 0.398 0.446 0.429 0.418 0.339 ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagp2\_ind : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ lead1 : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ lpl\_flag1 : num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204\_VG\_P2\_h[df202204\_VG\_P2\_g$V1\_date\_time\_num > 1648966227,], n=15L)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V1\_date\_time\_num  
## 19 1 04/02/2022 2022-04-02 23:10:27 0.336 1648966227  
## 20 1 04/02/2022 2022-04-02 23:10:57 0.363 1648966257  
## 21 1 04/02/2022 2022-04-02 23:11:27 0.459 1648966287  
## 22 1 04/02/2022 2022-04-02 23:11:57 0.344 1648966318  
## 23 1 04/02/2022 2022-04-02 23:12:27 0.404 1648966348  
## 24 1 04/02/2022 2022-04-02 23:12:57 0.395 1648966378  
## 25 1 04/02/2022 2022-04-02 23:13:28 0.369 1648966408  
## 26 1 04/02/2022 2022-04-02 23:13:58 0.919 1648966438  
## 27 1 04/02/2022 2022-04-02 23:14:28 0.448 1648966468  
## 28 1 04/02/2022 2022-04-02 23:14:58 0.537 1648966499  
## 29 1 04/02/2022 2022-04-02 23:15:28 0.451 1648966529  
## 30 1 04/02/2022 2022-04-02 23:15:58 0.353 1648966559  
## 31 1 04/02/2022 2022-04-02 23:16:29 0.476 1648966589  
## 32 1 04/02/2022 2022-04-02 23:16:59 0.363 1648966619  
## 33 1 04/02/2022 2022-04-02 23:17:29 0.382 1648966649  
## lagvg\_east lagvg\_west lagp2\_ind lead1 lpl\_flag1 upchange  
## 19 30 30 0 0 0 0  
## 20 30 30 0 0 0 0  
## 21 30 30 0 1 1 1  
## 22 NA NA 1 1 1 0  
## 23 NA NA 1 1 1 0  
## 24 NA NA 1 0 1 0  
## 25 30 30 0 0 0 0  
## 26 30 30 0 0 0 0  
## 27 30 30 0 1 1 1  
## 28 NA NA 1 0 1 0  
## 29 30 30 0 1 1 1  
## 30 NA NA 1 0 1 0  
## 31 30 30 0 1 1 1  
## 32 NA NA 1 0 1 0  
## 33 30 30 0 0 0 0

save(df202204\_VG\_P2\_h, file = "df202204\_VG\_P2\_h.RData")

Split data frame *df202204\_VG\_P2\_h* into two data frames *df202204\_VG\_P2\_i* and *df202204\_VG\_P2\_j*. The data frame *df202204\_VG\_P2\_i* contains all records with *lpl\_flag1 > 0* that together with the *upchange* field will help to identify the groups of records for which the seeing needs to be averaged.

df202204\_VG\_P2\_i = df202204\_VG\_P2\_h[df202204\_VG\_P2\_h$lpl\_flag1 > 0,]  
df202204\_VG\_P2\_j = df202204\_VG\_P2\_h[df202204\_VG\_P2\_h$lpl\_flag1 == 0,]   
head(df202204\_VG\_P2\_i)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V1\_date\_time\_num  
## 21 1 04/02/2022 2022-04-02 23:11:27 0.459 1648966287  
## 22 1 04/02/2022 2022-04-02 23:11:57 0.344 1648966318  
## 23 1 04/02/2022 2022-04-02 23:12:27 0.404 1648966348  
## 24 1 04/02/2022 2022-04-02 23:12:57 0.395 1648966378  
## 27 1 04/02/2022 2022-04-02 23:14:28 0.448 1648966468  
## 28 1 04/02/2022 2022-04-02 23:14:58 0.537 1648966499  
## lagvg\_east lagvg\_west lagp2\_ind lead1 lpl\_flag1 upchange  
## 21 30 30 0 1 1 1  
## 22 NA NA 1 1 1 0  
## 23 NA NA 1 1 1 0  
## 24 NA NA 1 0 1 0  
## 27 30 30 0 1 1 1  
## 28 NA NA 1 0 1 0

head(df202204\_VG\_P2\_j)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V1\_date\_time\_num  
## 1 1 04/02/2022 2022-04-02 23:01:14 0.398 1648965674  
## 2 1 04/02/2022 2022-04-02 23:01:44 0.446 1648965704  
## 3 1 04/02/2022 2022-04-02 23:02:14 0.429 1648965734  
## 4 1 04/02/2022 2022-04-02 23:02:44 0.418 1648965765  
## 5 1 04/02/2022 2022-04-02 23:03:14 0.339 1648965795  
## 6 1 04/02/2022 2022-04-02 23:03:44 0.589 1648965825  
## lagvg\_east lagvg\_west lagp2\_ind lead1 lpl\_flag1 upchange  
## 1 30 30 0 0 0 0  
## 2 30 30 0 0 0 0  
## 3 30 30 0 0 0 0  
## 4 30 30 0 0 0 0  
## 5 30 30 0 0 0 0  
## 6 30 30 0 0 0 0

In the data frame *df202204\_VG\_P2\_i* flag the various groups of records for which the seeing will be averaged. This is done calculating the cumulative sum for the field *upchange*.

df202204\_VG\_P2\_i$upchg\_counter <- cumsum(df202204\_VG\_P2\_i$upchange)   
head(df202204\_VG\_P2\_i, n=20L)

## P2\_ind V1\_date V1\_date\_time V2\_seeing\_rnd V1\_date\_time\_num  
## 21 1 04/02/2022 2022-04-02 23:11:27 0.459 1648966287  
## 22 1 04/02/2022 2022-04-02 23:11:57 0.344 1648966318  
## 23 1 04/02/2022 2022-04-02 23:12:27 0.404 1648966348  
## 24 1 04/02/2022 2022-04-02 23:12:57 0.395 1648966378  
## 27 1 04/02/2022 2022-04-02 23:14:28 0.448 1648966468  
## 28 1 04/02/2022 2022-04-02 23:14:58 0.537 1648966499  
## 29 1 04/02/2022 2022-04-02 23:15:28 0.451 1648966529  
## 30 1 04/02/2022 2022-04-02 23:15:58 0.353 1648966559  
## 31 1 04/02/2022 2022-04-02 23:16:29 0.476 1648966589  
## 32 1 04/02/2022 2022-04-02 23:16:59 0.363 1648966619  
## 34 1 04/02/2022 2022-04-02 23:17:59 0.427 1648966680  
## 35 1 04/02/2022 2022-04-02 23:18:29 0.532 1648966710  
## 36 1 04/02/2022 2022-04-02 23:18:59 0.948 1648966740  
## 37 1 04/02/2022 2022-04-02 23:19:30 0.490 1648966770  
## 38 1 04/02/2022 2022-04-02 23:20:00 0.364 1648966800  
## 39 1 04/02/2022 2022-04-02 23:20:30 0.361 1648966830  
## 42 1 04/02/2022 2022-04-02 23:22:00 0.472 1648966921  
## 43 1 04/02/2022 2022-04-02 23:22:31 0.432 1648966951  
## 47 1 04/02/2022 2022-04-02 23:24:31 0.429 1648967072  
## 48 1 04/02/2022 2022-04-02 23:25:01 0.471 1648967102  
## lagvg\_east lagvg\_west lagp2\_ind lead1 lpl\_flag1 upchange upchg\_counter  
## 21 30 30 0 1 1 1 1  
## 22 NA NA 1 1 1 0 1  
## 23 NA NA 1 1 1 0 1  
## 24 NA NA 1 0 1 0 1  
## 27 30 30 0 1 1 1 2  
## 28 NA NA 1 0 1 0 2  
## 29 30 30 0 1 1 1 3  
## 30 NA NA 1 0 1 0 3  
## 31 30 30 0 1 1 1 4  
## 32 NA NA 1 0 1 0 4  
## 34 30 30 0 1 1 1 5  
## 35 NA NA 1 0 1 0 5  
## 36 30 30 0 1 1 1 6  
## 37 NA NA 1 0 1 0 6  
## 38 30 30 0 1 1 1 7  
## 39 NA NA 1 0 1 0 7  
## 42 30 30 0 1 1 1 8  
## 43 NA NA 1 0 1 0 8  
## 47 30 30 0 1 1 1 9  
## 48 NA NA 1 1 1 0 9

For each group calculate the average seeing: *V2\_seeing\_mean*

#aggregate  
df202204\_VG\_P2\_k = aggregate(df202204\_VG\_P2\_i[, c("V2\_seeing\_rnd")], list(df202204\_VG\_P2\_i$upchg\_counter), mean)  
str(df202204\_VG\_P2\_k)

## 'data.frame': 765 obs. of 2 variables:  
## $ Group.1: num 1 2 3 4 5 6 7 8 9 10 ...  
## $ x : num 0.401 0.493 0.402 0.419 0.48 ...

colnames(df202204\_VG\_P2\_k)

## [1] "Group.1" "x"

names(df202204\_VG\_P2\_k)[names(df202204\_VG\_P2\_k) == "Group.1"] <- "upchg\_counter"  
names(df202204\_VG\_P2\_k)[names(df202204\_VG\_P2\_k) == "x"] <- "V2\_seeing\_mean"  
colnames(df202204\_VG\_P2\_k)

## [1] "upchg\_counter" "V2\_seeing\_mean"

head(df202204\_VG\_P2\_k, n=20L)

## upchg\_counter V2\_seeing\_mean  
## 1 1 0.4005000  
## 2 2 0.4925000  
## 3 3 0.4020000  
## 4 4 0.4195000  
## 5 5 0.4795000  
## 6 6 0.7190000  
## 7 7 0.3625000  
## 8 8 0.4520000  
## 9 9 0.4226667  
## 10 10 0.5890000  
## 11 11 0.4040000  
## 12 12 0.4480000  
## 13 13 0.3590000  
## 14 14 0.8630000  
## 15 15 0.5330000  
## 16 16 0.6745000  
## 17 17 1.1595000  
## 18 18 0.8953333  
## 19 19 0.8070000  
## 20 20 0.5560000

Keep only rows with *lagvg\_east* non missing (NA) and selected columns from *df202204\_VG\_P2\_i* and produce the data frame *df202204\_VG\_P2\_m*.

#Keep only non NA rows and subset columns  
df202204\_VG\_P2\_l = df202204\_VG\_P2\_i[!is.na(df202204\_VG\_P2\_i$lagvg\_east)=="TRUE",]  
df202204\_VG\_P2\_m = df202204\_VG\_P2\_l[, c("P2\_ind", "V1\_date", "V1\_date\_time", "V1\_date\_time\_num", "lagvg\_east", "lagvg\_west", "upchg\_counter")]  
str(df202204\_VG\_P2\_m)

## 'data.frame': 765 obs. of 7 variables:  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:11:27" "2022-04-02 23:14:28" ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ upchg\_counter : num 1 2 3 4 5 6 7 8 9 10 ...

head(df202204\_VG\_P2\_m)

## P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east lagvg\_west  
## 21 1 04/02/2022 2022-04-02 23:11:27 1648966287 30 30  
## 27 1 04/02/2022 2022-04-02 23:14:28 1648966468 30 30  
## 29 1 04/02/2022 2022-04-02 23:15:28 1648966529 30 30  
## 31 1 04/02/2022 2022-04-02 23:16:29 1648966589 30 30  
## 34 1 04/02/2022 2022-04-02 23:17:59 1648966680 30 30  
## 36 1 04/02/2022 2022-04-02 23:18:59 1648966740 30 30  
## upchg\_counter  
## 21 1  
## 27 2  
## 29 3  
## 31 4  
## 34 5  
## 36 6

Merge *df202204\_VG\_P2\_m* and *df202204\_VG\_P2\_k* by *upchg\_counter* to append the seeing mean *V2\_seeing\_mean* to *df202204\_VG\_P2\_m*.

df202204\_VG\_P2\_n = merge(df202204\_VG\_P2\_m, df202204\_VG\_P2\_k, by = "upchg\_counter")  
str(df202204\_VG\_P2\_n)

## 'data.frame': 765 obs. of 8 variables:  
## $ upchg\_counter : num 1 2 3 4 5 6 7 8 9 10 ...  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:11:27" "2022-04-02 23:14:28" ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ V2\_seeing\_mean : num 0.401 0.493 0.402 0.419 0.48 ...

head(df202204\_VG\_P2\_n)

## upchg\_counter P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num  
## 1 1 1 04/02/2022 2022-04-02 23:11:27 1648966287  
## 2 2 1 04/02/2022 2022-04-02 23:14:28 1648966468  
## 3 3 1 04/02/2022 2022-04-02 23:15:28 1648966529  
## 4 4 1 04/02/2022 2022-04-02 23:16:29 1648966589  
## 5 5 1 04/02/2022 2022-04-02 23:17:59 1648966680  
## 6 6 1 04/02/2022 2022-04-02 23:18:59 1648966740  
## lagvg\_east lagvg\_west V2\_seeing\_mean  
## 1 30 30 0.4005  
## 2 30 30 0.4925  
## 3 30 30 0.4020  
## 4 30 30 0.4195  
## 5 30 30 0.4795  
## 6 30 30 0.7190

Change the name of group variable *upchg\_counter* to *group*.

#Change name of group variable  
names(df202204\_VG\_P2\_n)[names(df202204\_VG\_P2\_n)=="upchg\_counter"] <- "group"  
head(df202204\_VG\_P2\_n)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 1 1 04/02/2022 2022-04-02 23:11:27 1648966287 30  
## 2 2 1 04/02/2022 2022-04-02 23:14:28 1648966468 30  
## 3 3 1 04/02/2022 2022-04-02 23:15:28 1648966529 30  
## 4 4 1 04/02/2022 2022-04-02 23:16:29 1648966589 30  
## 5 5 1 04/02/2022 2022-04-02 23:17:59 1648966680 30  
## 6 6 1 04/02/2022 2022-04-02 23:18:59 1648966740 30  
## lagvg\_west V2\_seeing\_mean  
## 1 30 0.4005  
## 2 30 0.4925  
## 3 30 0.4020  
## 4 30 0.4195  
## 5 30 0.4795  
## 6 30 0.7190

save(df202204\_VG\_P2\_n, file = "df202204\_VG\_P2\_n.RData")

Select and order coulmns for *df202204\_VG\_P2\_j*.

df202204\_VG\_P2\_o = df202204\_VG\_P2\_j[, c("upchange", "P2\_ind", "V1\_date", "V1\_date\_time", "V1\_date\_time\_num", "lagvg\_east", "lagvg\_west","V2\_seeing\_rnd" )]  
head(df202204\_VG\_P2\_o)

## upchange P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 0 1 04/02/2022 2022-04-02 23:01:14 1648965674 30  
## 2 0 1 04/02/2022 2022-04-02 23:01:44 1648965704 30  
## 3 0 1 04/02/2022 2022-04-02 23:02:14 1648965734 30  
## 4 0 1 04/02/2022 2022-04-02 23:02:44 1648965765 30  
## 5 0 1 04/02/2022 2022-04-02 23:03:14 1648965795 30  
## 6 0 1 04/02/2022 2022-04-02 23:03:44 1648965825 30  
## lagvg\_west V2\_seeing\_rnd  
## 1 30 0.398  
## 2 30 0.446  
## 3 30 0.429  
## 4 30 0.418  
## 5 30 0.339  
## 6 30 0.589

Rename “upchange” to “group” and “V2\_seeing\_rnd” to “V2\_seeing\_mean”

#Rename "upchange" to "group" and "V2\_seeing\_rnd" to "V2\_seeing\_mean"  
names(df202204\_VG\_P2\_o)[names(df202204\_VG\_P2\_o)=="upchange"] <- "group"  
names(df202204\_VG\_P2\_o)[names(df202204\_VG\_P2\_o)=="V2\_seeing\_rnd"] <- "V2\_seeing\_mean"  
str(df202204\_VG\_P2\_o)

## 'data.frame': 4960 obs. of 8 variables:  
## $ group : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ V2\_seeing\_mean : num 0.398 0.446 0.429 0.418 0.339 ...

head(df202204\_VG\_P2\_o)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 0 1 04/02/2022 2022-04-02 23:01:14 1648965674 30  
## 2 0 1 04/02/2022 2022-04-02 23:01:44 1648965704 30  
## 3 0 1 04/02/2022 2022-04-02 23:02:14 1648965734 30  
## 4 0 1 04/02/2022 2022-04-02 23:02:44 1648965765 30  
## 5 0 1 04/02/2022 2022-04-02 23:03:14 1648965795 30  
## 6 0 1 04/02/2022 2022-04-02 23:03:44 1648965825 30  
## lagvg\_west V2\_seeing\_mean  
## 1 30 0.398  
## 2 30 0.446  
## 3 30 0.429  
## 4 30 0.418  
## 5 30 0.339  
## 6 30 0.589

save(df202204\_VG\_P2\_o, file = "df202204\_VG\_P2\_o.RData")

Concatenate *df202204\_VG\_P2\_n* (open Vent Gate measurment with multiple Seeing measurements) and *df202204\_VG\_P2\_o* (open Vent Gate measurment with unique seeing [P2] measurement)

#Concatenate df202204\_VG\_P2\_n (open Vent Gate measurment with multiple Seeing measurements) and  
#df202204\_VG\_P2\_o (open Vent Gate measurment with unique seeing measurement)  
df202204\_VG\_P2\_p=bind\_rows(df202204\_VG\_P2\_o, df202204\_VG\_P2\_n )  
head(df202204\_VG\_P2\_p)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 0 1 04/02/2022 2022-04-02 23:01:14 1648965674 30  
## 2 0 1 04/02/2022 2022-04-02 23:01:44 1648965704 30  
## 3 0 1 04/02/2022 2022-04-02 23:02:14 1648965734 30  
## 4 0 1 04/02/2022 2022-04-02 23:02:44 1648965765 30  
## 5 0 1 04/02/2022 2022-04-02 23:03:14 1648965795 30  
## 6 0 1 04/02/2022 2022-04-02 23:03:44 1648965825 30  
## lagvg\_west V2\_seeing\_mean  
## 1 30 0.398  
## 2 30 0.446  
## 3 30 0.429  
## 4 30 0.418  
## 5 30 0.339  
## 6 30 0.589

Sort and save the resulting data frame

# sort data  
V1\_date\_time\_sorted <- order(df202204\_VG\_P2\_p$V1\_date\_time\_num)  
df202204\_VG\_P2\_q <- df202204\_VG\_P2\_p[V1\_date\_time\_sorted, ]  
str(df202204\_VG\_P2\_q)

## 'data.frame': 5725 obs. of 8 variables:  
## $ group : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ V2\_seeing\_mean : num 0.398 0.446 0.429 0.418 0.339 ...

head(df202204\_VG\_P2\_q[df202204\_VG\_P2\_q$V1\_date\_time\_num > 1648966197,], n=10L)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 19 0 1 04/02/2022 2022-04-02 23:10:27 1648966227 30  
## 20 0 1 04/02/2022 2022-04-02 23:10:57 1648966257 30  
## ...4961 1 1 04/02/2022 2022-04-02 23:11:27 1648966287 30  
## 25 0 1 04/02/2022 2022-04-02 23:13:28 1648966408 30  
## 26 0 1 04/02/2022 2022-04-02 23:13:58 1648966438 30  
## ...4962 2 1 04/02/2022 2022-04-02 23:14:28 1648966468 30  
## ...4963 3 1 04/02/2022 2022-04-02 23:15:28 1648966529 30  
## ...4964 4 1 04/02/2022 2022-04-02 23:16:29 1648966589 30  
## 33 0 1 04/02/2022 2022-04-02 23:17:29 1648966649 30  
## ...4965 5 1 04/02/2022 2022-04-02 23:17:59 1648966680 30  
## lagvg\_west V2\_seeing\_mean  
## 19 30 0.3360  
## 20 30 0.3630  
## ...4961 30 0.4005  
## 25 30 0.3690  
## 26 30 0.9190  
## ...4962 30 0.4925  
## ...4963 30 0.4020  
## ...4964 30 0.4195  
## 33 30 0.3820  
## ...4965 30 0.4795

tail(df202204\_VG\_P2\_q)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 7750 0 1 04/30/2022 2022-04-30 04:39:14 1651318754 5  
## 7751 0 1 04/30/2022 2022-04-30 04:39:44 1651318784 5  
## 7752 0 1 04/30/2022 2022-04-30 04:40:28 1651318829 5  
## 7753 0 1 04/30/2022 2022-04-30 04:41:04 1651318865 5  
## 7754 0 1 04/30/2022 2022-04-30 04:41:41 1651318901 5  
## NA NA NA <NA> <NA> NA NA  
## lagvg\_west V2\_seeing\_mean  
## 7750 80 3.741  
## 7751 80 6.001  
## 7752 80 4.896  
## 7753 80 4.892  
## 7754 80 3.577  
## NA NA NA

#Save a copy of dataframe  
save(df202204\_VG\_P2\_q, file = "df202204\_VG\_P2\_q.RData")

Drop last row with all NAs produced by lag functions, and then save data frame with vent gate and seeing data.

#Drop row with all NA (Last row)  
df202204\_VG\_P2\_r = df202204\_VG\_P2\_q[rowSums(is.na(df202204\_VG\_P2\_q)) != ncol(df202204\_VG\_P2\_q),]  
head(df202204\_VG\_P2\_r[df202204\_VG\_P2\_r$V1\_date\_time\_num > 1648966197,], n=10L)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 19 0 1 04/02/2022 2022-04-02 23:10:27 1648966227 30  
## 20 0 1 04/02/2022 2022-04-02 23:10:57 1648966257 30  
## ...4961 1 1 04/02/2022 2022-04-02 23:11:27 1648966287 30  
## 25 0 1 04/02/2022 2022-04-02 23:13:28 1648966408 30  
## 26 0 1 04/02/2022 2022-04-02 23:13:58 1648966438 30  
## ...4962 2 1 04/02/2022 2022-04-02 23:14:28 1648966468 30  
## ...4963 3 1 04/02/2022 2022-04-02 23:15:28 1648966529 30  
## ...4964 4 1 04/02/2022 2022-04-02 23:16:29 1648966589 30  
## 33 0 1 04/02/2022 2022-04-02 23:17:29 1648966649 30  
## ...4965 5 1 04/02/2022 2022-04-02 23:17:59 1648966680 30  
## lagvg\_west V2\_seeing\_mean  
## 19 30 0.3360  
## 20 30 0.3630  
## ...4961 30 0.4005  
## 25 30 0.3690  
## 26 30 0.9190  
## ...4962 30 0.4925  
## ...4963 30 0.4020  
## ...4964 30 0.4195  
## 33 30 0.3820  
## ...4965 30 0.4795

tail(df202204\_VG\_P2\_r)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 7749 0 1 04/30/2022 2022-04-30 04:38:44 1651318724 5  
## 7750 0 1 04/30/2022 2022-04-30 04:39:14 1651318754 5  
## 7751 0 1 04/30/2022 2022-04-30 04:39:44 1651318784 5  
## 7752 0 1 04/30/2022 2022-04-30 04:40:28 1651318829 5  
## 7753 0 1 04/30/2022 2022-04-30 04:41:04 1651318865 5  
## 7754 0 1 04/30/2022 2022-04-30 04:41:41 1651318901 5  
## lagvg\_west V2\_seeing\_mean  
## 7749 80 1.176  
## 7750 80 3.741  
## 7751 80 6.001  
## 7752 80 4.896  
## 7753 80 4.892  
## 7754 80 3.577

#Save a copy of dataframe  
save(df202204\_VG\_P2\_r, file = "df202204\_VG\_P2\_r.RData")

## Data Exploration

Add flag *Seeing\_BG* to identify high average seeing measured by P2.

#Add an indicator for Seeing  
df202204\_VG\_P2\_s=mutate(df202204\_VG\_P2\_r, Seeing\_BG=ifelse(V2\_seeing\_mean >=1.2,1,0) )  
str(df202204\_VG\_P2\_s)

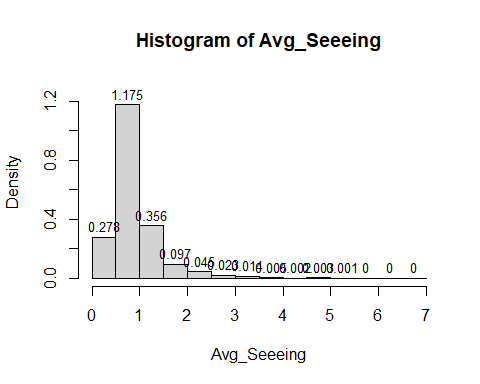
## 'data.frame': 5724 obs. of 9 variables:  
## $ group : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V1\_date\_time\_num: num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ V2\_seeing\_mean : num 0.398 0.446 0.429 0.418 0.339 ...  
## $ Seeing\_BG : num 0 0 0 0 0 0 0 0 0 0 ...

head(df202204\_VG\_P2\_s)

## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 0 1 04/02/2022 2022-04-02 23:01:14 1648965674 30  
## 2 0 1 04/02/2022 2022-04-02 23:01:44 1648965704 30  
## 3 0 1 04/02/2022 2022-04-02 23:02:14 1648965734 30  
## 4 0 1 04/02/2022 2022-04-02 23:02:44 1648965765 30  
## 5 0 1 04/02/2022 2022-04-02 23:03:14 1648965795 30  
## 6 0 1 04/02/2022 2022-04-02 23:03:44 1648965825 30  
## lagvg\_west V2\_seeing\_mean Seeing\_BG  
## 1 30 0.398 0  
## 2 30 0.446 0  
## 3 30 0.429 0  
## 4 30 0.418 0  
## 5 30 0.339 0  
## 6 30 0.589 0

Check the distribution of *lagvg\_east*, *lagvg\_west*, *V2\_seeing\_mean* and *Seeing\_BG*.

#Produce a density histogram  
Avg\_Seeeing <- df202204\_VG\_P2\_s$V2\_seeing\_mean  
  
h <-hist(Avg\_Seeeing, freq=FALSE, breaks=c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,6.5,7),ylim=c(0,1.3))  
text(h$mids,h$density,labels=round(h$density,digits=3), adj=c(0.5, -0.5), cex=.8)



# Bar chart with High Low Seeing  
library(ggplot2)  
  
g <- ggplot(df202204\_VG\_P2\_s, aes(x = factor(Seeing\_High\_Low))) +  
 geom\_bar()   
  
summary(df202204\_VG\_P2\_s[,c("V1\_date","V1\_date\_time","lagvg\_east","lagvg\_west","V2\_seeing\_mean","Seeing\_BG")])

## V1\_date V1\_date\_time lagvg\_east   
## Length:5724 Min. :2022-04-02 23:01:14 Min. : 0.00   
## Class :character 1st Qu.:2022-04-19 22:05:45 1st Qu.: 0.00   
## Mode :character Median :2022-04-22 21:36:25 Median : 15.00   
## Mean :2022-04-20 19:18:34 Mean : 22.91   
## 3rd Qu.:2022-04-25 01:31:38 3rd Qu.: 30.00   
## Max. :2022-04-30 04:41:41 Max. :100.00   
## lagvg\_west V2\_seeing\_mean Seeing\_BG   
## Min. : 0.00 Min. :0.1600 Min. :0.0000   
## 1st Qu.: 10.00 1st Qu.:0.5850 1st Qu.:0.0000   
## Median : 20.00 Median :0.7440 Median :0.0000   
## Mean : 32.38 Mean :0.9032 Mean :0.1733   
## 3rd Qu.: 50.00 3rd Qu.:1.0340 3rd Qu.:0.0000   
## Max. :100.00 Max. :6.7040 Max. :1.0000

Generate vent gate bins

df202204\_VG\_P2\_t <- df202204\_VG\_P2\_s %>% mutate(Vent\_Gate\_East\_bin=cut(lagvg\_east, breaks = c(-1,0,10,20,30,40,50,60,70,80,90,100)), Vent\_Gate\_West\_bin=cut(lagvg\_west, breaks = c(-1,0,10,20,30,40,50,60,70,80,90,100)))  
str(df202204\_VG\_P2\_t)

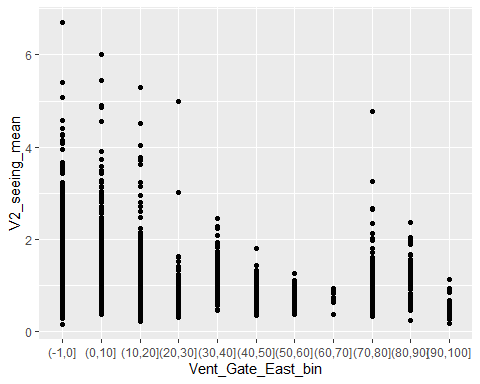
## 'data.frame': 5724 obs. of 11 variables:  
## $ group : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ P2\_ind : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ V1\_date : chr "04/02/2022" "04/02/2022" "04/02/2022" "04/02/2022" ...  
## $ V1\_date\_time : POSIXct, format: "2022-04-02 23:01:14" "2022-04-02 23:01:44" ...  
## $ V1\_date\_time\_num : num 1.65e+09 1.65e+09 1.65e+09 1.65e+09 1.65e+09 ...  
## $ lagvg\_east : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ lagvg\_west : num 30 30 30 30 30 30 30 30 30 30 ...  
## $ V2\_seeing\_mean : num 0.398 0.446 0.429 0.418 0.339 ...  
## $ Seeing\_BG : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Vent\_Gate\_East\_bin: Factor w/ 11 levels "(-1,0]","(0,10]",..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ Vent\_Gate\_West\_bin: Factor w/ 11 levels "(-1,0]","(0,10]",..: 4 4 4 4 4 4 4 4 4 4 ...

head(df202204\_VG\_P2\_t, n=20L)

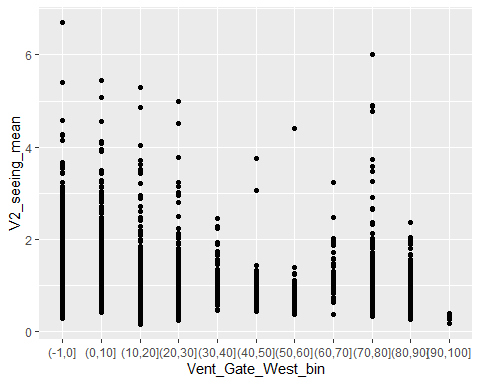
## group P2\_ind V1\_date V1\_date\_time V1\_date\_time\_num lagvg\_east  
## 1 0 1 04/02/2022 2022-04-02 23:01:14 1648965674 30  
## 2 0 1 04/02/2022 2022-04-02 23:01:44 1648965704 30  
## 3 0 1 04/02/2022 2022-04-02 23:02:14 1648965734 30  
## 4 0 1 04/02/2022 2022-04-02 23:02:44 1648965765 30  
## 5 0 1 04/02/2022 2022-04-02 23:03:14 1648965795 30  
## 6 0 1 04/02/2022 2022-04-02 23:03:44 1648965825 30  
## 7 0 1 04/02/2022 2022-04-02 23:04:15 1648965855 30  
## 8 0 1 04/02/2022 2022-04-02 23:04:45 1648965885 30  
## 9 0 1 04/02/2022 2022-04-02 23:05:15 1648965915 30  
## 10 0 1 04/02/2022 2022-04-02 23:05:45 1648965946 30  
## 11 0 1 04/02/2022 2022-04-02 23:06:15 1648965976 30  
## 12 0 1 04/02/2022 2022-04-02 23:06:45 1648966006 30  
## 13 0 1 04/02/2022 2022-04-02 23:07:16 1648966036 30  
## 14 0 1 04/02/2022 2022-04-02 23:07:46 1648966066 30  
## 15 0 1 04/02/2022 2022-04-02 23:08:26 1648966106 30  
## 16 0 1 04/02/2022 2022-04-02 23:08:56 1648966137 30  
## 17 0 1 04/02/2022 2022-04-02 23:09:26 1648966167 30  
## 18 0 1 04/02/2022 2022-04-02 23:09:56 1648966197 30  
## 19 0 1 04/02/2022 2022-04-02 23:10:27 1648966227 30  
## 20 0 1 04/02/2022 2022-04-02 23:10:57 1648966257 30  
## lagvg\_west V2\_seeing\_mean Seeing\_BG Vent\_Gate\_East\_bin Vent\_Gate\_West\_bin  
## 1 30 0.398 0 (20,30] (20,30]  
## 2 30 0.446 0 (20,30] (20,30]  
## 3 30 0.429 0 (20,30] (20,30]  
## 4 30 0.418 0 (20,30] (20,30]  
## 5 30 0.339 0 (20,30] (20,30]  
## 6 30 0.589 0 (20,30] (20,30]  
## 7 30 0.617 0 (20,30] (20,30]  
## 8 30 1.001 0 (20,30] (20,30]  
## 9 30 0.446 0 (20,30] (20,30]  
## 10 30 0.511 0 (20,30] (20,30]  
## 11 30 0.418 0 (20,30] (20,30]  
## 12 30 0.384 0 (20,30] (20,30]  
## 13 30 0.359 0 (20,30] (20,30]  
## 14 30 0.349 0 (20,30] (20,30]  
## 15 30 1.178 0 (20,30] (20,30]  
## 16 30 0.586 0 (20,30] (20,30]  
## 17 30 0.479 0 (20,30] (20,30]  
## 18 30 0.386 0 (20,30] (20,30]  
## 19 30 0.336 0 (20,30] (20,30]  
## 20 30 0.363 0 (20,30] (20,30]

Seeing spread by vent gate bin

#Mean Seeing neasurements per Vent gate plot  
ggplot(df202204\_VG\_P2\_t, aes(x=Vent\_Gate\_East\_bin, y=V2\_seeing\_mean)) + geom\_point()



ggplot(df202204\_VG\_P2\_t, aes(x=Vent\_Gate\_West\_bin, y=V2\_seeing\_mean)) + geom\_point()

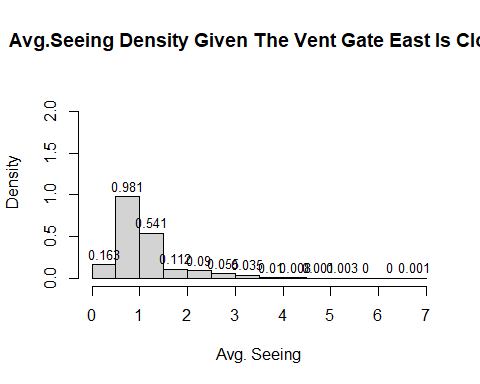


Seeing distributions when the Vent Gate East is closed.

#Keep records with lagvg\_east == 0  
df\_tmp1 =df202204\_VG\_P2\_t[df202204\_VG\_P2\_t$lagvg\_east == 0,]  
summary(df\_tmp1[,c("lagvg\_east")])

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0 0 0 0 0 0

#Produce a seeing density histogram given vent gate east == 0  
h1 <-hist(df\_tmp1$V2\_seeing\_mean, freq=FALSE, breaks=c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,6.5,7),main="Avg.Seeing Density Given The Vent Gate East Is Close", xlab="Avg. Seeing",ylim=c(0,2.3))  
text(h1$mids,h1$density,labels=round(h1$density,digits=3), adj=c(0.5, -0.5), cex=.8)

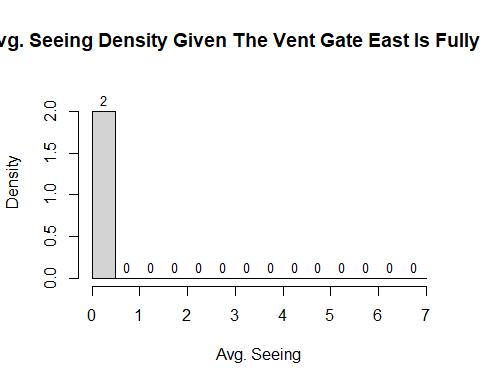


Seeing distributions when the Vent Gate East is fully opened

#Keep records with lagvg\_east == 100  
df\_tmp1 =df202204\_VG\_P2\_t[df202204\_VG\_P2\_t$lagvg\_east == 100,]  
summary(df\_tmp1[,c("lagvg\_east")])

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 100 100 100 100 100 100

#Produce a seeing density histogram given vent gate east == 100  
h1 <-hist(df\_tmp1$V2\_seeing\_mean, freq=FALSE, breaks=c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,6.5,7),main="Avg. Seeing Density Given The Vent Gate East Is Fully Open", xlab="Avg. Seeing",ylim=c(0,2.3))  
text(h1$mids,h1$density,labels=round(h1$density,digits=3), adj=c(0.5, -0.5), cex=.8)

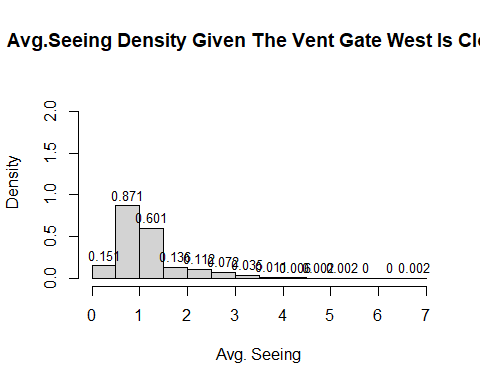


Seeing distributions when the Vent Gate West is closed.

#Keep records with lagvg\_east == 0  
df\_tmp1 =df202204\_VG\_P2\_t[df202204\_VG\_P2\_t$lagvg\_west == 0,]  
summary(df\_tmp1[,c("lagvg\_west")])

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0 0 0 0 0 0

#Produce a seeing density histogram given vent gate east == 0  
h1 <-hist(df\_tmp1$V2\_seeing\_mean, freq=FALSE, breaks=c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,6.5,7),main="Avg.Seeing Density Given The Vent Gate West Is Close", xlab="Avg. Seeing",ylim=c(0,2.3))  
text(h1$mids,h1$density,labels=round(h1$density,digits=3), adj=c(0.5, -0.5), cex=.8)

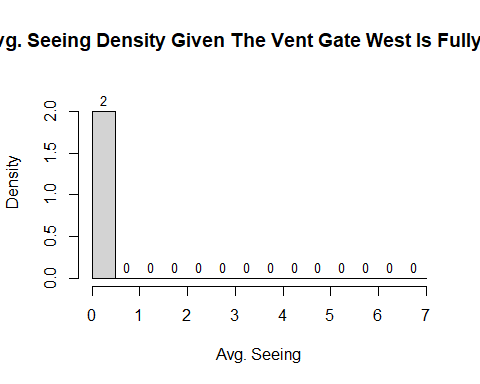


Seeing distributions when the Vent Gate West is fully opened

#Keep records with lagvg\_east == 100  
df\_tmp1 =df202204\_VG\_P2\_t[df202204\_VG\_P2\_t$lagvg\_west == 100,]  
summary(df\_tmp1[,c("lagvg\_west")])

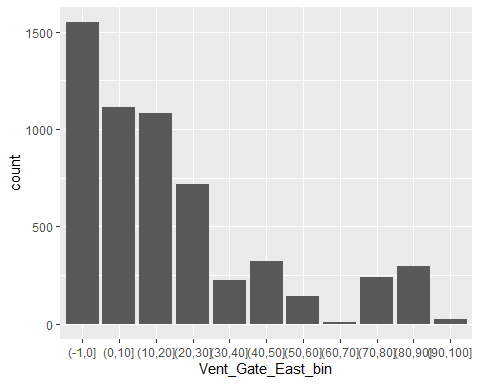
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 100 100 100 100 100 100

#Produce a seeing density histogram given vent gate east == 100  
h1 <-hist(df\_tmp1$V2\_seeing\_mean, freq=FALSE, breaks=c(0,0.5,1,1.5,2,2.5,3,3.5,4,4.5,5,5.5,6,6.5,7),main="Avg. Seeing Density Given The Vent Gate West Is Fully Open", xlab="Avg. Seeing",ylim=c(0,2.3))  
text(h1$mids,h1$density,labels=round(h1$density,digits=3), adj=c(0.5, -0.5), cex=.8)

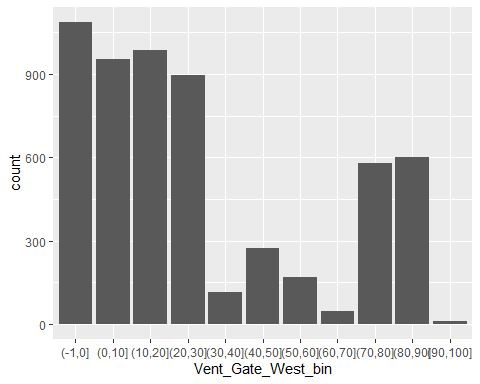


Vent Gate frequency distributions.

#Frequency plots  
ggplot(df202204\_VG\_P2\_t, aes(x=Vent\_Gate\_East\_bin)) + geom\_bar()

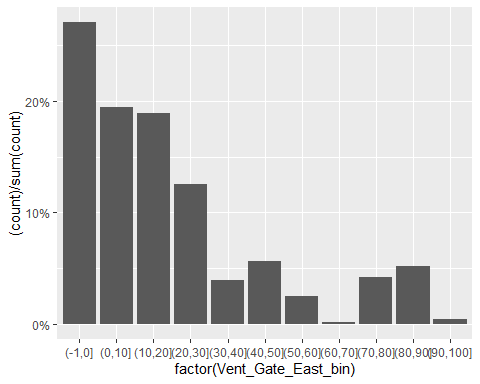


ggplot(df202204\_VG\_P2\_t, aes(x=Vent\_Gate\_West\_bin)) + geom\_bar()

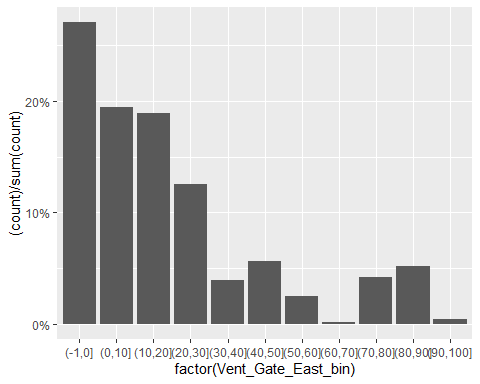


Vent Gate density distributions

#Density plots  
ggplot(df202204\_VG\_P2\_t, aes(x=factor(Vent\_Gate\_East\_bin))) + geom\_bar(aes(y= (..count..)/sum(..count..))) + scale\_y\_continuous(labels=scales::percent\_format(accuracy = 1L))



ggplot(df202204\_VG\_P2\_t, aes(x=factor(Vent\_Gate\_East\_bin))) + geom\_bar(aes(y= (..count..)/sum(..count..))) + scale\_y\_continuous(labels=scales::percent\_format(accuracy = 1L))



## ====================End===================