Protocol and Procedure: Analysis of Gas Chromatography Data

PREPARED BY	ORGANIZATION	DATE
Dr. Terrance Loecke	Loecke Lab, University of Kansas	04/20/2021

APPROVALS	ORGANIZATION	DATE

Change Record

REVISION	DATE	DESCRIPTION OF CHANGE
A_DRAFT	7/26/2021	Initial draft release.
Rev. B	11/5/2021	Changed the check standard deviation tolerance from 2% to 5%.

Table of Contents

1. Data transfer from GC computer to cloud storage

1.1 Run R script to create unique GC results file names

1. Results files include metadata and GC parameters
    i. Metadata
        1. Unique sample filenames
        2. Acquired Date – Date and time of sample analysis
        3. Analysis Date - Date and time of last sample integration
        4. Vial position within sample trays
        5. Tray number
        6. Type of integration – auto or manual
        7. File Name – unique file of GC responses for each sample
    ii. GC parameters
        1. Compound Name – e.g., SF_6, CH_4, CO_2, or N_2O
        2. RT - retention time of signal of compounds on each detector
        3. Area – area count of the compound
        4. Height – height from baseline to peak for each compound
2. This script copies the results file and assigns a unique file name based on the directory
3. Open R script in R
    i. R script is saved on the GC computer with a shortcut on the desktop
4. Copy script into R console and type return
    i. If successful a list of TRUE will be returned for each results file
    ii. And the individual files will be saved into a folder with today's date in the directory named.
5. If changes are made to the script an updated file is saved with the date as part of the file name (e.g., GC2uniqueResults_v20210720.R).

2. Transfer results files from GC computer to cloud storage

1. Transfer results files from GC computer to cloud storage using a flash drive.
    a. The set of uniquely named results files are transferred from the GC computer to cloud storage (OneDrive) using a flash drive. The GC is not networked to avoid software updates that may disrupt analysis runs.

3. Prepare Lab Summary

A Lab Summary is prepared annually or following major changes to instruments or analytical methods. Reported in this summary are updates for specific method detection limits (MDL), method precision, and measurement uncertainty for each analytical method. These method summary statistics are calculated from the results of eight replicate samples following USEPA- 821-R-16-006. (https://www.epa.gov/sites/default/files/2016-12/documents/mdl-procedure_rev2_12-13-2016.pdf)

1. In brief, eight replicate samples are prepared that are 3-5 times the concentration or mass of the expected.
2. These samples are analyzed accordingly to the method SOP
3. Results are prepared as below in the Prepare sample report section.
4. The standard deviation, S, of the eight samples is calculated.
5. The mean of these eight replicates, u, is calculated as the arithmetic mean.
6. The MDL equals S * t-student at 99% (2.998 for 8 df for one-tail distribution)
7. Precision is calculated as S / u * 100
8. Measurement Uncertainty is S
9. The dates for each summary period are reported
10. This summary is added to the previous summary and submitted to the NEON data ingest system on-line.

4. Prepare sample report

Sample reports contain data submitted through the NEON ingest system. The reports are prepared using scripts written in R and saved on KU OneDrive.

1. Load gas standard tables.
    a. Table contains unique IDs for gas cylinders and dilutions. Recipes for making the dilutions are also contained within.
2. Manually inspect each results file for integration consistence of the standard curve. If inconsistent, adjust GC method to accommodate or reject the run. Retention time shifts following changing columns is the most common issue, but this happens only every 12 to 24 months.
3. Load results files one at a time in a for loop.
4. Join standard curve parameters and sample results files into a single data frame.
5. Develop standard curve for each compound (e.g., CO2, N2O, CH4, SF6).
    a. Develop curves for a high and low concentration range
    b. For ECD curves, develop specific curves for each channel.
6. Calculate slope (β), intercept (α), and fit (R2) for each curve
    a. Note, curves are calculated as area counts (Y axis) as a function of known standard concentration (X axis).
    b. R2 of less than 0.99 are rejected
7. Apply standard curves to unknowns (samples)
    a. Unknown conc = (unknown area - α)/ β

8. Calculate internal run precision statistic as the percent standard deviation of replicated checks prepared at approximately the mid-range of concentration for each analyte.
    a. Check % SD = standard deviation of check concentrations / mean of check concentrations * 100
    b. Greater than 5% error results in a rejection of the run
    c. NEON adds a flag for results between 2-5% upon ingest to the data portal.
9. Save copies of raw and processed data as csv and xlsx files respectively.
10. Update sample inventory
11. Join sample inventory metadata to sample report per sample.
12. Check for duplicate samples in report
13. Check for compliance of the sample name against the standard using regular expressions.
14. Check for appropriate concentration ranges
15. Join minimum detection limit table (LabSummary) to sample report.

5. Specific script for GHG data analysis

Below is the script used to process the GHG samples. The specific directory references are set up to work within an RStudio project.

```
# GHG ----------------------

rm(list=ls())

library(lubridate)

library(readxl)

library(schoolmath)

library(openxlsx)

library(dplyr)


# import standard concentrations ----------------------------------------

stdTable <- data.frame(readxl::read_excel(path="../../../Projects/NEON gas/neonSequenceTableScripts/standardTables/gcStandards20190130.xlsx",sheet = "gcStandards"))

# stdTable <- data.frame(readxl::read_excel(path="../../GC/gcStandards20190130.xlsx",sheet = "gcStandards"))


dataDir <- "../../../Projects/NEON gas/NEON GHG/Needs to be checked by PI//"


lf <- list.files(path=dataDir, full.names = T,
```

```
            pattern = "TDL.txt|TDL.csv|TDL.CSV", include.dirs = T)
lfs <- list.files(path=dataDir, full.names = F,
             pattern = ".csv|.CSV", include.dirs = F)

##### import raw GC files ----------
tmps1 <- NULL
#V where is closing bracket??
for(i in lf){
  tmp <-read.csv(i, fileEncoding="UTF-16", header=T, sep="\t")
  tmp$ord <- 1:nrow(tmp)
  tmp$gcColumn <- ifelse(is.odd(tmp$ord),1,2)
  tmp$CH4_ppmv <- stdTable[match(tmp$Sample.Name,stdTable$gcStdID),"CH4_ppmv"]
  tmp$CO2_ppmv <- stdTable[match(tmp$Sample.Name,stdTable$gcStdID),"CO2_ppmv"]
  tmp$N2O_ppmv <- stdTable[match(tmp$Sample.Name,stdTable$gcStdID),"N2O_ppmv"]
  #rename 3 "area" columns
  names(tmp)[7] <- "area.CH4"
  names(tmp)[12] <- "area.CO2"
  names(tmp)[17] <- "area.N2O"


  std <- tmp[grep(x=tmp$Sample.Name, pattern = "ghg"),]
  tmp1 <- tmp[tmp$gcColumn==1,]
  tmp2 <- tmp[tmp$gcColumn==2,]
  std1 <- std[std$gcColumn==1,]
  std2 <- std[std$gcColumn==2,]
  std1$pRank <- percent_rank(std1$CH4_ppmv)
  std2 <- std[std$gcColumn==2,]
  std2$pRank <- percent_rank(std2$CH4_ppmv)
  #CH4 curve
```

```
nSTD <- length(std$ord)
if(length(std$area.CH4)==0) {
  tmp1$CH4_ppmv <- NA
} else {
  linearL <- lm(area.CH4 ~ CH4_ppmv, data=std[1:round(nSTD*.75,0),])
}

yintL <- linearL$coefficients[1]
mL <- linearL$coefficients[2]
tmp1$CH4R2Low <- summary(linearL)$r.squared
tmp2$CH4R2Low <- summary(linearL)$r.squared
tmp1$CH4_ppmvLow <- NA
tmp1$CH4_ppmvLow <- (tmp1$area.CH4-yintL)/mL
tmp2$CH4_ppmvLow <- NA
tmp2$CH4_ppmvLow <- (tmp2$area.CH4-yintL)/mL

if(length(std$area.CH4)==0) {
  tmp1$CH4_ppmv <- NA
} else {
  linearH <- lm(area.CH4 ~ CH4_ppmv, data=std[(round(nSTD*.25,0)+1):nSTD,])
}
tmp1$CH4R2High <- summary(linearH)$r.squared
tmp2$CH4R2High <- summary(linearH)$r.squared
yintH <- linearH$coefficients[1]
mH <- linearH$coefficients[2]
tmp1$CH4_ppmvHigh <- NA
tmp1$CH4_ppmvHigh <- (tmp1$area.CH4-yintH)/mH
tmp2$CH4_ppmvHigh <- NA
tmp2$CH4_ppmvHigh<- (tmp2$area.CH4-yintH)/mH
```

```
#CO2 curves


if(length(std$area.CO2)==0) {
  tmp1$CO2_ppmv <- NA
} else {
  linearL <- lm(area.CO2 ~ CO2_ppmv, data=std[1:round(nSTD*.75,0),])
}
tmp1$CO2R2Low <- summary(linearL)$r.squared
tmp2$CO2R2Low <- summary(linearL)$r.squared
yintL <- linearL$coefficients[1]
mL <- linearL$coefficients[2]
tmp1$CO2_ppmvLow <- NA
tmp1$CO2_ppmvLow <- (tmp1$area.CO2-yintL)/mL
tmp2$CO2_ppmvLow <- NA
tmp2$CO2_ppmvLow <- (tmp2$area.CO2-yintL)/mL


if(length(std$area.CO2)==0) {
  tmp1$CO2_ppmv <- NA
} else {
  linearH <- lm(area.CO2 ~ CO2_ppmv, data=std[(round(nSTD*.25,0)+1):nSTD,])
}
tmp1$CO2R2High <- summary(linearH)$r.squared
tmp2$CO2R2High <- summary(linearH)$r.squared
yintH <- linearH$coefficients[1]
mH <- linearH$coefficients[2]
tmp1$CO2_ppmvHigh <- NA
tmp1$CO2_ppmvHigh <- (tmp1$area.CO2-yintH)/mH
tmp2$CO2_ppmvHigh <- NA
```

```
tmp2$CO2_ppmvHigh<- (tmp2$area.CO2-yintH)/mH


#N2O curves
nSTD1 <- length(std1$ord)
if(length(std1$area.N2O)==0) {
  tmp1$N2O_ppmv <- NA
} else {
  linear1Low <- lm(area.N2O ~ N2O_ppmv, data=std1[1:round(nSTD1*.75,0),])
}
tmp1$N2OR2Low <- summary(linear1Low)$r.squared
yint1Low <- linear1Low$coefficients[1]
m1Low <- linear1Low$coefficients[2]
tmp1$N2O_ppmvLow <- NA
tmp1$N2O_ppmvLow <- (tmp1$area.N2O-yint1Low)/m1Low

nSTD2 <- length(std2$ord)
if(length(std2$area.N2O)==0) {
  tmp2$N2O_ppmv <- NA
} else {
  linear2Low <- lm(area.N2O ~ N2O_ppmv, data=std2[1:round(nSTD2*.75,0),])
}
tmp2$N2OR2Low <- summary(linear2Low)$r.squared
yint2Low <- linear2Low$coefficients[1]
m2Low <- linear2Low$coefficients[2]
tmp2$N2O_ppmvLow <- NA
tmp2$N2O_ppmvLow <- (tmp2$area.N2O-yint2Low)/m2Low

## high conc
```

```
if(length(std1$area.N2O)==0) {
  tmp1$N2O_ppmv <- NA
} else {
  linear1High <- lm(area.N2O ~ N2O_ppmv, data=std1[(round(nSTD1*.25,0)+1):nSTD1,])
}
tmp1$N2OR2High <- summary(linear1High)$r.squared
yint1High <- linear1High$coefficients[1]
m1High <- linear1High$coefficients[2]
tmp1$N2O_ppmvHigh <- NA
tmp1$N2O_ppmvHigh <- (tmp1$area.N2O-yint1High)/m1High



if(length(std2$area.N2O)==0) {
  tmp2$N2O_ppmv <- NA
} else {
  linear2High <- lm(area.N2O ~ N2O_ppmv, data=std2[(round(nSTD2*.25,0)+1):nSTD2,])
}
tmp2$N2OR2High <- summary(linear2High)$r.squared
yint2High <- linear2High$coefficients[1]
m2High <- linear2High$coefficients[2]
tmp2$N2O_ppmvHigh <- NA
tmp2$N2O_ppmvHigh <- (tmp2$area.N2O-yint2High)/m2High

####
tmpall <- NULL
tmpall <- dplyr::bind_rows(tmp1, tmp2)

tmpall <- tmpall[order(tmpall$Acquired.Date),]
thrsCH4 <- median(std$CH4_ppmv)
```

```
  tmpall$CH4_ppmvFinal <- ifelse(tmpall$CH4_ppmvLow <= thrsCH4,
                                 tmpall$CH4_ppmvLow, tmpall$CH4_ppmvHigh)
  thrsCO2 <- median(std$CO2_ppmv)
  tmpall$CO2_ppmvFinal <- ifelse(tmpall$CO2_ppmvLow <= thrsCO2,
                                 tmpall$CO2_ppmvLow, tmpall$CO2_ppmvHigh)
  thrsN2O <- median(std$N2O_ppmv)
  tmpall$N2O_ppmvFinal <- ifelse(tmpall$N2O_ppmvLow <= thrsN2O,
                                 tmpall$N2O_ppmvLow, tmpall$N2O_ppmvHigh)

  tmpall$CH4_R2Final <- ifelse(tmpall$CH4_ppmvLow <= thrsCH4,
                               tmpall$CH4R2Low, tmpall$CH4R2High)

  tmpall$CO2_R2Final <- ifelse(tmpall$CO2_ppmvLow <= thrsCO2,
                               tmpall$CO2R2Low, tmpall$CO2R2High)

  tmpall$N2O_R2Final <- ifelse(tmpall$N2O_ppmvLow <= thrsN2O,
                               tmpall$N2OR2Low, tmpall$N2OR2High)
# add standard dev of checks
  ###
  chk <- tmpall[grep(tmpall$Sample.Name, pattern = "heck"),]
  chk <- chk[3:nrow(chk),] # remove first two as bypass
  chkStatCH4 <- sd(chk$CH4_ppmvFinal)/mean(chk$CH4_ppmvFinal)*100
  chkStatCO2 <- sd(chk$CO2_ppmvFinal)/mean(chk$CO2_ppmvFinal)*100
  chkStatN2O <- sd(chk$N2O_ppmvFinal)/mean(chk$N2O_ppmvFinal)*100
# tmpall$chkStatCH4 <- chkStatCH4
# tmpall$chkStatCO2 <- chkStatCO2
# tmpall$chkStatN2O <- chkStatN2O

tmpall$CH4CheckStandardPercentDev <- chkStatCH4
```

```
tmpall$CO2CheckStandardPercentDev <- chkStatCO2

tmpall$N2OCheckStandardPercentDev <- chkStatN2O

sn <- paste0("NEON_GHG_",Sys.Date())

sn <- gsub(x=sn, pattern = ":", replacement = ".")

fcomponents <- unlist(strsplit(i, split = "\\/"))

fraw <- fcomponents[grepl(pattern = "csv|CSV|txt", x=fcomponents)]

fraw2 <- unlist(strsplit(fraw, split = "\\."))[1]

exfname <- paste0("../../../Projects/NEON gas/NEON GHG/Needs to be checked by PI/",fraw2,".xlsx")

#openxlsx::write.xlsx(tmp, file = exfname, sheetName = "raw", row.names = F)

openxlsx::write.xlsx(tmpall, file = exfname, sheetName=sn, append = T, row.names = F)

}


# import previous report parameters  ---------------------------

source("scr/updateGHGReportList.R")

fileName <- paste0("data/neonGHGReported_", Sys.Date(),".csv")

reports <- readr::read_csv(file = fileName, col_names = T)


##get ingest data

source("scr/updateMasterList.R")

inv <- paste0("data/neonMasterInventory_",Sys.Date(),".csv")


inv2 <- read.csv(file=inv,header = T,sep=",", stringsAsFactors = F)


# combine for reporting ----------------------------------------

dataDir <- "../../../Projects/NEON gas/NEON GHG/Needs to be checked by PI/"

lf <- list.files(path=dataDir, full.names = T, pattern = "TDL.xlsx", include.dirs = T)


tmps2 <- NULL

for(i in lf){
```

```
  sn <- paste0("NEON_GHG_",Sys.Date())
  tmp <- readxl::read_xlsx(path = i, col_names = TRUE, sheet = sn)


  tmp <- tmp[,c("Vial","Acquired.Date","Sample.Name",
                "File.Name","CH4_ppmvFinal",
                "CO2_ppmvFinal","N2O_ppmvFinal",
                "CH4CheckStandardPercentDev",
                "CO2CheckStandardPercentDev",
                "N2OCheckStandardPercentDev")]


  tmp$CH4_ppmvFinal[tmp$CH4_ppmvFinal<0] <- 0
  tmp$CO2_ppmvFinal[tmp$CO2_ppmvFinal<0] <- 0
  tmp$N2O_ppmvFinal[tmp$N2O_ppmvFinal<0] <- 0
# correction of common labeling mistakes ----------------------
  tmp$Sample.Name <- gsub(pattern = "\\.AIR", x = tmp$Sample.Name, replacement = "\\.air")
  tmp$Sample.Name <- gsub(pattern = "\\.WAT", x = tmp$Sample.Name, replacement = "\\.wat")
  tmp$Sample.Name <- gsub(pattern = "\\.SS\\.", x = tmp$Sample.Name, replacement = "\\.ss\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.C0\\.", x = tmp$Sample.Name, replacement = "\\.c0\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.CO\\.", x = tmp$Sample.Name, replacement = "\\.c0\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.C1\\.", x = tmp$Sample.Name, replacement = "\\.c1\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.C2\\.", x = tmp$Sample.Name, replacement = "\\.c2\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.OT\\.", x = tmp$Sample.Name, replacement = "\\.ot\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.IN\\.", x = tmp$Sample.Name, replacement = "\\.in\\.")
  tmp$Sample.Name <- gsub(pattern = "\\.RE\\.", x = tmp$Sample.Name, replacement = "\\.re\\.")
  tmp$Sample.Name <- gsub(pattern = "CIRO", x = tmp$Sample.Name, replacement = "LIRO")


  tmp$Sample.Name <- as.character(tmp$Sample.Name)
  # tmp$sampleCode1 <-
as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleCode"])
```

```
 # tmp$sampleCode2 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleID),"sampleCode"])
 # tmp$sampleID <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleID"])


 tmp$sampleID1 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleID"])
 tmp$sampleID2 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleID),"sampleID"])
 tmp$sampleID3 <- as.character(ifelse(!is.na(tmp$sampleID1),tmp$sampleID1,
                 ifelse(!is.na(tmp$sampleID2),tmp$sampleID2,
                 tmp$Sample.Name)))

 tmps2 <- rbind(tmps2, tmp)
}

cnames <- names(tmps2)
tmps <- as.data.frame(tmps2)
tmps$`Analysis Date` <- as.Date(lubridate::dmy_hms(as.character(tmps$Acquired.Date)))
table(tmps$sampleID3%in%inv2$sampleID)
tmps$`Received Date` <- as.Date(lubridate::ymd(inv2[match(tmps$sampleID3,
inv2$sampleID),"shipmentReceivedDate"]))
tmps <- tmps[grep(x=tmps$sampleID3, pattern =
"ghg|blank|check|bypass|stop|BLANK|Blank|Stop|STOP|Check|KNZ_",invert = T),]
tmps$`Site ID` <- substr(tmps$sampleID3,1,4)
table(tmps$sampleID3%in%inv2$sampleID)

tmps$`Sample Condition` <- inv2[match(tmps$sampleID3,inv2$sampleID),"sampleCondition"]
tmps[,c("Shipment ID","Received by","Sample Condition")] <-
inv2[match(tmps$sampleID3,inv2$sampleID),
                                        c("shipmentID",
                                          "receivedBy",
                                          "sampleCondition")]
```

```
tmps$`Remarks` <- "NA"

tmps$`Sample Condition`[is.na(tmps$`Sample Condition`)] <- "OK"

dups <- tmps[duplicated(tmps$sampleID3),]

tmps$`Analysis Date`[tmps$sampleID3%in%dups$sampleID3]

contest <- tmps[tmps$sampleID3%in%dups$sampleID3,]

contest[contest$Sample.Name==unique(contest$Sample.Name)[1],]

# dropfilenames <- as.character(contest$File.Name[grep(x=contest$File.Name, pattern = "2020-09-
23")])

droptemp <- NULL

keep <- NULL

for (i in unique(contest$Sample.Name)){

  pmet <- contest[contest$Sample.Name==i,]

  keeptemp <- pmet[which.max(pmet$`Analysis Date`),]

  keep <- rbind(keep, keeptemp)

}

keep$Remarks <- "reran on Analysis Date"


tmps <- tmps[!tmps$File.Name%in%contest$File.Name,]


tmps <- rbind(tmps, keep)


# construct penultimate df ----------


df1 <- tmps[,c("Analysis Date","Vial","Received Date","Sample.Name",

          "File.Name","CH4_ppmvFinal","CO2_ppmvFinal",

          "N2O_ppmvFinal","sampleID3","Site ID","Shipment ID",

          "Received by","Sample Condition","CH4CheckStandardPercentDev",

          "CO2CheckStandardPercentDev","N2OCheckStandardPercentDev")]
```

```
#convert names to ingest template names

# ingest template

ingest <- read_xls(path = "Attachment 2b_Dissolved Gas Data Ingest_External
Lab_updated_20210505.xls", sheet = "perSample")

names(df1) <- c("Analysis Date","Vial","Received Date","Lab Sample Name",

        "File Name","CH4 ppm","CO2 ppm","N2O ppm","Sample ID",

        "Site ID","Shipment ID","Received by","Sample Condition",

        "CH4CheckStandardPercentDev",

        "CO2CheckStandardPercentDev","N2OCheckStandardPercentDev")


reports$`Lab Processor` <- "B.Richards"

df2 <- reports[1,c("Lab Name", "Lab Processor","Volume Analyzed (mL)")]

df3 <- tmps["Remarks"]

# need summary file

sumFile <- read.csv(file = "../../NEON share/audit reports/April
2021/REA_TCR_Summary_KU20210505.csv")

sumFile$Lab.Specific.Start.Date <- lubridate::ymd(sumFile$Lab.Specific.Start.Date)

sumFileCH4 <- sumFile[sumFile$Analyte=="methane" & sumFile$Lab.Specific.Start.Date=="2021-04-
29",]

sumFileCO2 <- sumFile[sumFile$Analyte=="carbon dioxide" & sumFile$Lab.Specific.Start.Date=="2021-
04-29",]

sumFileN2O <- sumFile[sumFile$Analyte=="Nitrous oxide" & sumFile$Lab.Specific.Start.Date=="2021-
04-29",]

df4 <- data.frame('Run detection limit - CH4ppm'=sumFileCH4$Method.Detection.Limit, "Run detection
limit - CO2ppm"=sumFileCO2$Method.Detection.Limit, "Run detection limit -
N2Oppm"=sumFileN2O$Method.Detection.Limit, "Precision (CV%) - CH4ppm"=sumFileCH4$Precision,
"Precision (CV%) - CO2ppm"=sumFileCO2$Precision, "Precision (CV%) -
N2Oppm"=sumFileN2O$Precision, "Certified Standard Accuracy (% +/-)"=2)


names(df4) <- c('Run detection limit - CH4ppm',

        'Run detection limit - CO2ppm',

        'Run detection limit - N2Oppm',
```

```
        'Precision (CV%) - CH4ppm','Precision (CV%) - CO2ppm',
        'Precision (CV%) - N2Oppm',
        'Certified Standard Accuracy (% +/-)')

# df4 <- reports[1,c("Run detection limit - CH4ppm", "Run detection limit - CO2ppm", "Run detection limit - N2Oppm","Precision (CV%) - CH4ppm", "Precision (CV%) - CO2ppm","Precision (CV%) - N2Oppm" ,"Certified Standard Accuracy (% +/-)")]

dfF <- cbind(df1, df2, df3, df4)

# colnames(dfF) <- names(reports)
dfF$`Sample Code` <- NA
# dfF$ "CH4CheckStandardPercentDev" <- NA
# dfF$ "CO2CheckStandardPercentDev" <- NA
# dfF$ "N2OCheckStandardPercentDev" <- NA
dfF$Remarks <- as.character(dfF$Remarks)
dfF <- dfF[order(dfF$`Received Date`,decreasing = F),]
dfcomplete <- dfF
dfF <- dfF[,names(ingest)]

dfF$Remarks[dfF$`CH4 ppm`==0] <- "BelowDetectionLimit"
dfF$Remarks[dfF$`CO2 ppm`==0] <- "BelowDetectionLimit"
dfF$Remarks[dfF$`N2O ppm`==0] <- "BelowDetectionLimit"
# dfF <- dfF[ifelse(duplicated(dfF$`Sample ID`),]
table(reports$`Sample ID`%in%dfF$`Sample ID`)
# dfF <- dfF[!dfF$`Sample ID`%in%reports$`Sample ID`,]
table(duplicated(dfF$`Sample ID`))

range(dfF$`CH4 ppm`, na.rm = T)
```

```
sum(is.na(dfF$`CH4 ppm`))>0
dfF$`CH4 ppm`[is.na(dfF$`CH4 ppm`)] <- "" # use if true
range(dfF$`CO2 ppm`, na.rm = T)
sum(is.na(dfF$`CO2 ppm`))>0
dfF$`CO2 ppm`[is.na(dfF$`CO2 ppm`)] <- "" # use if true
range(dfF$`N2O ppm`, na.rm = T)
sum(is.na(dfF$`N2O ppm`))>0
#dfF$`N2O ppm`[is.na(dfF$`CH4 ppm`)] <- "" # use if true
unique(dfF$`Received Date`)
table(dfF$`Received Date`)
unique(dfF$`Site ID`)
unique(dfF$`Sample Condition`)
unique(dfF$shipmentID)
# stop before writing to drive ----------
# check data in console and change date of file name
#
#
PostDir <- "../../../Projects/NEON gas/NEON GHG/Posted to NEON box/2021May_/"
write.csv(dfF, paste0(PostDir, "sdg_20210725.csv"), row.names = F)
```

6. Specific script to analyze SF_6 data

```
# SF6 ----------------------
rm(list = ls())
library(lubridate)
library(readxl)
library(schoolmath)
library(openxlsx)
library(dplyr)
```

```
# import standard concentrations ----------------------------------------

stdTable <- data.frame(readxl::read_excel(path="../../../Projects/NEON
gas/neonSequenceTableScripts/standardTables/gcStandards20190130.xlsx",sheet = "gcStandards"))

dataDir <- "../../../Projects/NEON gas/NEON sf6/Needs to be checked by PI/"

lf <- list.files(path=dataDir, full.names = T, pattern = "TDL.CSV", include.dirs = T)

lfs <- list.files(path=dataDir, full.names = F, pattern = ".csv|CSV", include.dirs = F)

#####

tmps1 <- NULL

for(i in lf){

 tmp <-read.csv(i, fileEncoding="UTF-16", header=T, sep="\t", stringsAsFactors = F, comment.char = "#")

 tmp <- tmp[,1:11]

 tmp$ord <- 1:nrow(tmp)

 tmp$gcColumn <- ifelse(is.odd(tmp$ord),1,2)

 tmp$SF6_ppmv <- stdTable[match(tmp$Sample.Name,stdTable$gcStdID),"SF6_ppmv"]

 #rename "area" columns

 names(tmp)[7] <- "area.SF6"

 names(tmp)[11] <- "File.Name"

 std <- tmp[grep(x=tmp$Sample.Name, pattern = "sf6"),]

 tmp1 <- tmp[tmp$gcColumn==1,]

 tmp2 <- tmp[tmp$gcColumn==2,]

 std1 <- std[std$gcColumn==1,]

 std1$pRank <- percent_rank(std1$SF6_ppmv)

 std2 <- std[std$gcColumn==2,]
```

```
std2$pRank <- percent_rank(std2$SF6_ppmv)


#SF6 curves


if(length(std1$area.SF6)==0) {
  tmp1$SF6_ppmv <- NA
} else {
  linear1Low <- lm(area.SF6 ~ SF6_ppmv, data=std1[std1$pRank<=.7,])
}
tmp1$SF6R2Low <- summary(linear1Low)$r.squared
yint1Low <- linear1Low$coefficients[1]
m1Low <- linear1Low$coefficients[2]
tmp1$SF6_ppmvLow <- NA
tmp1$SF6_ppmvLow <- (tmp1$area.SF6-yint1Low)/m1Low



if(length(std2$area.SF6)==0) {
  tmp2$SF6_ppmv <- NA
} else {
  linear2Low <- lm(area.SF6 ~ SF6_ppmv, data=std2[std2$pRank<=.7,])
}
tmp2$SF6R2Low <- summary(linear2Low)$r.squared
yint2Low <- linear2Low$coefficients[1]
m2Low <- linear2Low$coefficients[2]
tmp2$SF6_ppmvLow <- NA
tmp2$SF6_ppmvLow <- (tmp2$area.SF6-yint2Low)/m2Low


## high conc
```

```
if(length(std1$area.SF6)==0) {
  tmp1$SF6_ppmv <- NA
} else {
  linear1High <- lm(area.SF6 ~ SF6_ppmv, data=std1[std1$pRank>=.3,])
}
tmp1$SF6R2High <- summary(linear1High)$r.squared
yint1High <- linear1High$coefficients[1]
m1High <- linear1High$coefficients[2]
tmp1$SF6_ppmvHigh <- NA
tmp1$SF6_ppmvHigh <- (tmp1$area.SF6-yint1High)/m1High



if(length(std2$area.SF6)==0) {
  tmp2$SF6_ppmv <- NA
} else {
  linear2High <- lm(area.SF6 ~ SF6_ppmv, data=std2[std2$pRank>=.3,])
}
tmp2$SF6R2High <- summary(linear2High)$r.squared
yint2High <- linear2High$coefficients[1]
m2High <- linear2High$coefficients[2]
tmp2$SF6_ppmvHigh <- NA
tmp2$SF6_ppmvHigh <- (tmp2$area.SF6-yint2High)/m2High

####
tmpall <- NULL
tmpall <- dplyr::bind_rows(tmp1, tmp2)

tmpall <- tmpall[order(tmpall$Acquired.Date),]
thrsSF6 <- median(std$SF6_ppmv, na.rm = T)
```

```
 tmpall$SF6_ppmvFinal <- ifelse(tmpall$SF6_ppmvLow <= thrsSF6,
                                tmpall$SF6_ppmvLow, tmpall$SF6_ppmvHigh)

 tmpall$SF6_R2Final <- ifelse(tmpall$SF6_ppmvLow <= thrsSF6,
                              tmpall$SF6R2Low, tmpall$SF6R2High)

 chk <- tmpall[grep(tmpall$Sample.Name, pattern = "heck"),]
 chk <- chk[3:nrow(chk),] # remove first two as bypass
 chkStat <- sd(chk$SF6_ppmvFinal)/mean(chk$SF6_ppmvFinal)*100

 tmpall$gasCheckStandardPercentDev <- chkStat

 sn <- paste0("NEON_SF6_",Sys.Date())
 sn <- gsub(x=sn, pattern = ":", replacement = ".")
 fcomponents <- unlist(strsplit(i, split = "\\/"))
 fraw <- fcomponents[grepl(pattern = "csv|CSV", x=fcomponents)]
 fraw2 <- unlist(strsplit(fraw, split = "\\."))[1]
 exfname <- paste0("../../../Projects/NEON gas/NEON SF6/Needs to be checked by PI/",fraw2,".xlsx")
 openxlsx::write.xlsx(tmp, file = exfname, sheetName = "raw", row.names = F)
 openxlsx::write.xlsx(tmpall, file = exfname, sheetName=sn, append = T, row.names = F)

 ##########################
 # need to append excel file with "final" tab that includes all metadata required for report
 ## and excludes checks, stds, other non samples

}

source("scr/updateSF6ReportList.R")
fileName <- paste0("data/neonSF6Reported_", Sys.Date(),".csv")
```

```
reports <- readr::read_csv(file = fileName, col_names = T)


##get ingest data
source("scr/updateMasterList.R")
inv <- paste0("data/neonMasterInventory_",Sys.Date(),".csv")


inv2 <- read.csv(file=inv,header = T,sep=",", stringsAsFactors = F)


repo <- reports


# combine for reporting ----------------------------------------


dataDir <- "../../../Projects/NEON gas/NEON SF6/Needs to be checked by PI/"
lf <- list.files(path=dataDir, full.names = T, pattern = ".xlsx", include.dirs = T)


tmps2 <- NULL
for(i in lf){
 sn <- paste0("NEON_SF6_",Sys.Date())
 sn <- gsub(x=sn, pattern = ":", replacement = ".")
 tmp <- readxl::read_xlsx(path = i, col_names = TRUE, sheet = sn)


 tmp <- tmp[,c("Vial","Acquired.Date","Sample.Name",
           "File.Name","SF6_ppmvFinal","gasCheckStandardPercentDev")]


 tmp$SF6_ppmvFinal[tmp$SF6_ppmvFinal<0] <- 0
 tmp$Sample.Name <- gsub(pattern = "\\.AIR", x = tmp$Sample.Name, replacement = "\\.air")
 tmp$Sample.Name <- gsub(pattern = "\\.WAT", x = tmp$Sample.Name, replacement = "\\.wat")
 tmp$Sample.Name <- gsub(pattern = "\\.SS\\.", x = tmp$Sample.Name, replacement = "\\.ss\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.C0\\.", x = tmp$Sample.Name, replacement = "\\.c0\\.")
```

```
 tmp$Sample.Name <- gsub(pattern = "\\.CO\\.", x = tmp$Sample.Name, replacement = "\\.c0\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.C1\\.", x = tmp$Sample.Name, replacement = "\\.c1\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.C2\\.", x = tmp$Sample.Name, replacement = "\\.c2\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.OT\\.", x = tmp$Sample.Name, replacement = "\\.ot\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.IN\\.", x = tmp$Sample.Name, replacement = "\\.in\\.")
 tmp$Sample.Name <- gsub(pattern = "\\.RE\\.", x = tmp$Sample.Name, replacement = "\\.re\\.")

 tmp$Sample.Name <- as.character(tmp$Sample.Name)
 # tmp$sampleCode1 <-
as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleCode"])
 # tmp$sampleCode2 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleID),"sampleCode"])
 # tmp$sampleID <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleID"])

 tmp$sampleID1 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleCode),"sampleID"])
 tmp$sampleID2 <- as.character(inv2[match(tmp$Sample.Name,inv2$sampleID),"sampleID"])
 tmp$sampleID3 <- as.character(ifelse(!is.na(tmp$sampleID1),tmp$sampleID1,
                              ifelse(!is.na(tmp$sampleID2),tmp$sampleID2,
                                  tmp$Sample.Name)))

 tmps2 <- rbind(tmps2, tmp)
}

cnames <- names(tmps2)
tmps <- as.data.frame(tmps2)

tmps$`Analysis Date` <- as.Date(lubridate::dmy_hms(as.character(tmps$Acquired.Date)))
```

```
tmps$`Received Date` <- as.Date(lubridate::ymd(inv2[match(tmps$sampleID3,
inv2$sampleID),"shipmentReceivedDate"]))

tmps <- tmps[grep(x=tmps$sampleID3, pattern = "sf6|blank|check|bypass|stop|STOP",invert = T),]

tmps$`Site ID` <- substr(tmps$sampleID3,1,4)

tmps[,c("Shipment ID","Received by","Sample Condition","Remarks")] <-
inv2[match(tmps$sampleID3,inv2$sampleID),

                                                    c("shipmentID",

                                                      "receivedBy",

                                                      "sampleCondition",

                                                      "remarks")]

dups <- tmps[duplicated(tmps$sampleID3),]

dups$sampleID3

tmps$`Analysis Date`[tmps$sampleID3%in%dups$sampleID3]

contest <- tmps[tmps$sampleID3%in%dups$sampleID3,]

dropfilenames <- as.character(contest$File.Name[grep(x=contest$File.Name, pattern = "2020-11-
18|2020-11-20")])

tmps <- tmps[!tmps$File.Name%in%dropfilenames,]

tmps <- tmps[!duplicated(tmps$sampleID3),]


# construct penultimate df

df1 <- tmps[,c("Analysis Date","Vial","Received Date","Sample.Name",

        "File.Name")]

df2 <- "SF6"

df3 <- tmps[,c("SF6_ppmvFinal","sampleID3","Site ID",

        "Shipment ID","Received by","Sample Condition")]

df4 <- repo[1,c("Lab Name", "Analyzed By")]

df4[,2] <- "B. Richards"

df5 <- tmps[,"Remarks"]

sumFile <- read.csv(file = "../../NEON share/audit reports/April
2021/REA_TCR_Summary_KU20210505.csv")
```

```
sumFile$Lab.Specific.Start.Date <- lubridate::ymd(sumFile$Lab.Specific.Start.Date)

sumFileSF6 <- sumFile[sumFile$Analyte=="SF6" & sumFile$Lab.Specific.Start.Date=="2021-04-29",]

df6 <- data.frame("Volume Analyzed"=1,"Run Detection Limit"=sumFileSF6$Method.Detection.Limit,
"Precision"=sumFileSF6$Precision, "Certified Standard Accuracy"=2)

names(df6) <- c("Volume Analyzed","Run Detection Limit", "Precision", "Certified Standard Accuracy")

df7 <- tmps[,"gasCheckStandardPercentDev"]

dfF <- cbind(df1, df2, df3, df4, df5, df6, df7)

colnames(dfF) <- c(names(repo),"gasCheckStandardPercentDev")

dfF <- dfF[order(dfF$`Received Date`,decreasing = F),]

dfF$Remarks <- as.character(dfF$Remarks)

dfF$Remarks[dfF$`Gas Concentration`<=dfF$`Run Detection Limit`] <- "BelowDetectionLimit" # check to
see if re-analyzed

dfF$`Sample Condition`[is.na(dfF$`Sample Condition`)] <- "OK"


dfF <- dfF[!duplicated(dfF$`Sample ID`),]

table(repo$`Sample ID`%in%dfF$`Sample ID`)

range(dfF$`Gas Concentration`)

unique(dfF$`Received Date`)

unique(dfF$`Site ID`)

unique(dfF$`Sample Condition`)

unique(dfF$`Shipment ID`)

# check sample ID against regex (?i)^    "[A-Z]{4}\.[0-9]{2}\.[0-9]{8}\.GAS"

table(grepl(x=dfF$`Sample ID`, pattern = "[A-Z]{4}\\.[0-9]{2}\\.[0-9]{8}\\.GAS" ))

dfF$gasSampleCode <- NA

dfF$analyticalReplicateGAS <- NA

# dfF$ <- NA


PostDir <- "../../../Projects/NEON gas/NEON SF6/SF6 data deposited to NEON box/2019_/"

sn <- paste0("rea_gas_",Sys.Date())
```

```
sn <- gsub(x=sn, pattern = "-", replacement = "")

write.csv(dfF, paste0(PostDir, sn,".csv"), row.names = F)
```

7. Version control of scripts

1. When changes are made to a script an updated file is saved with the date as part of the file name (e.g., GC2uniqueResults_v20210720.R).
2. The KU OneDrive subscription maintains previous versions files for at least 2 years and more versions for more recent changes (e.g., all changes for 30 days).