ICPMS Data Analysis

```
library(tidyverse)
library(readr)
```

```
#importing the tidied ICPMS data
ICPMS <- read.csv("~/chem313/313_icpms/data/tidy_ICPMS.csv")
```

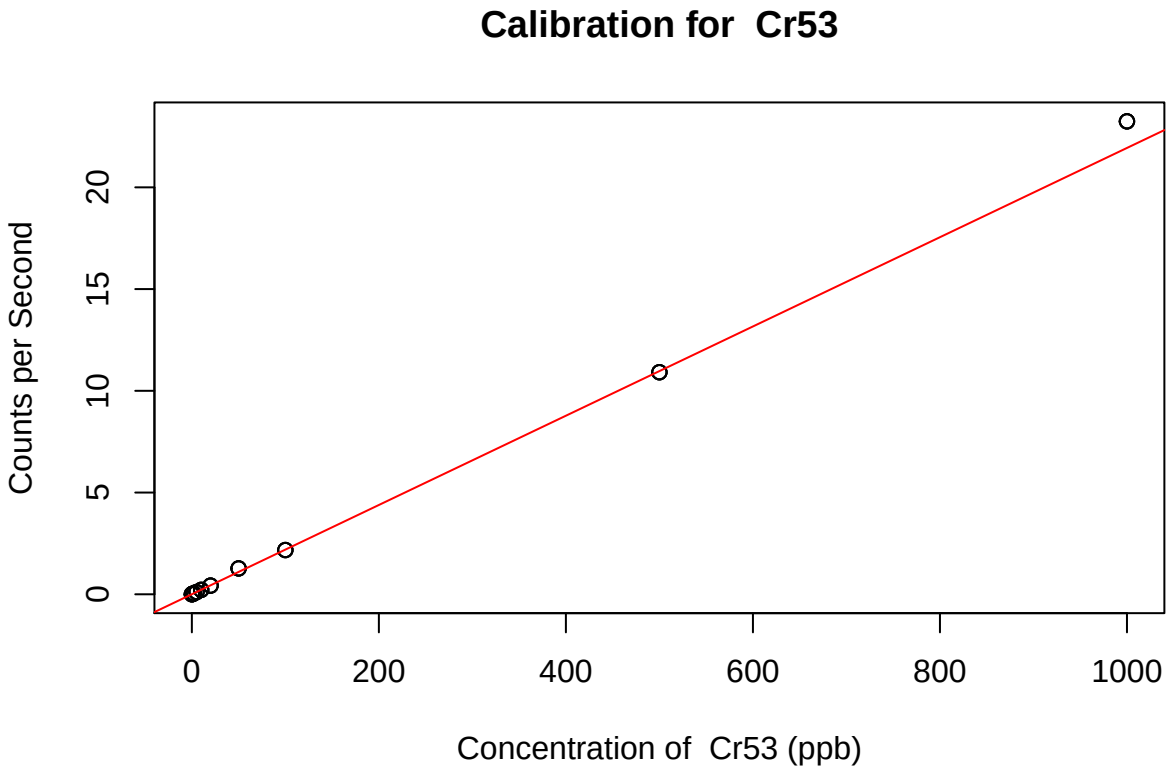
```
#defining lists to be used in for loops
sample_sites <- unique(filter(ICPMS, site != "MB", site != "")$site)
#excluding method blank and quality control from list of sites
metals_analyzed <- unique(ICPMS$metal)

#preview lists to check for issues
sample_sites
```

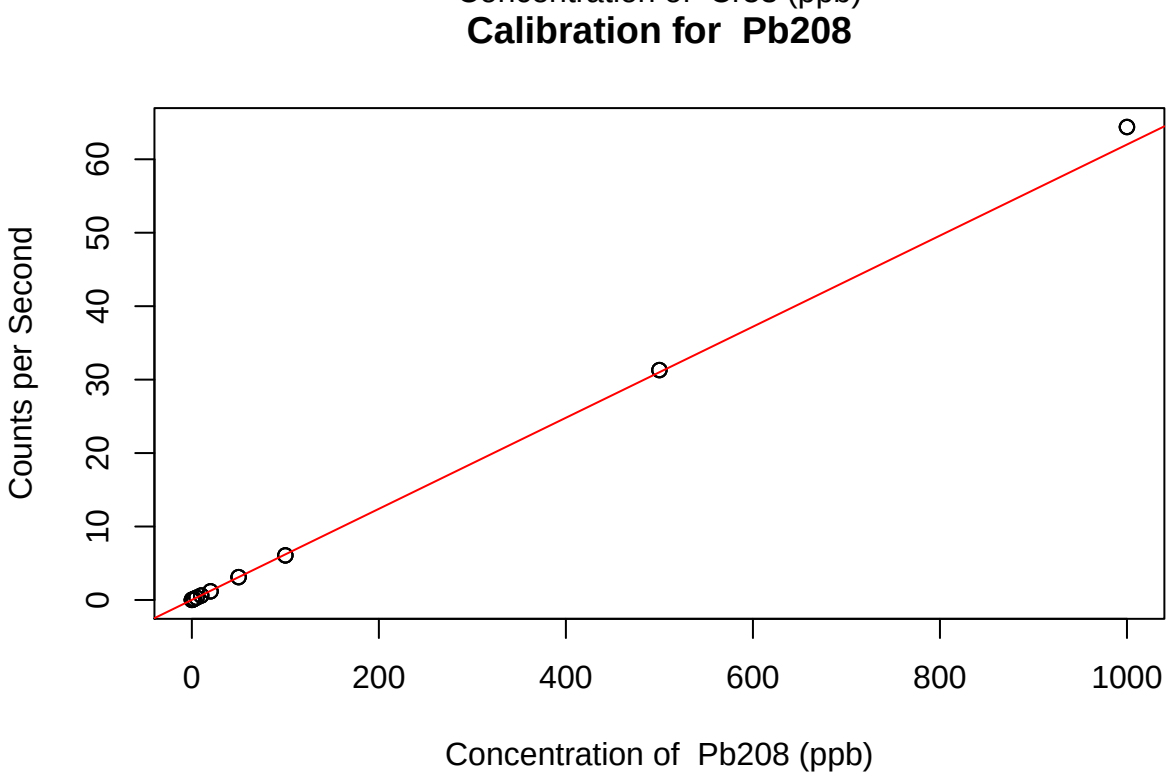
```
## [1] F  QC D  E  C  B  A
## Levels:  A B C D E F MB QC
```

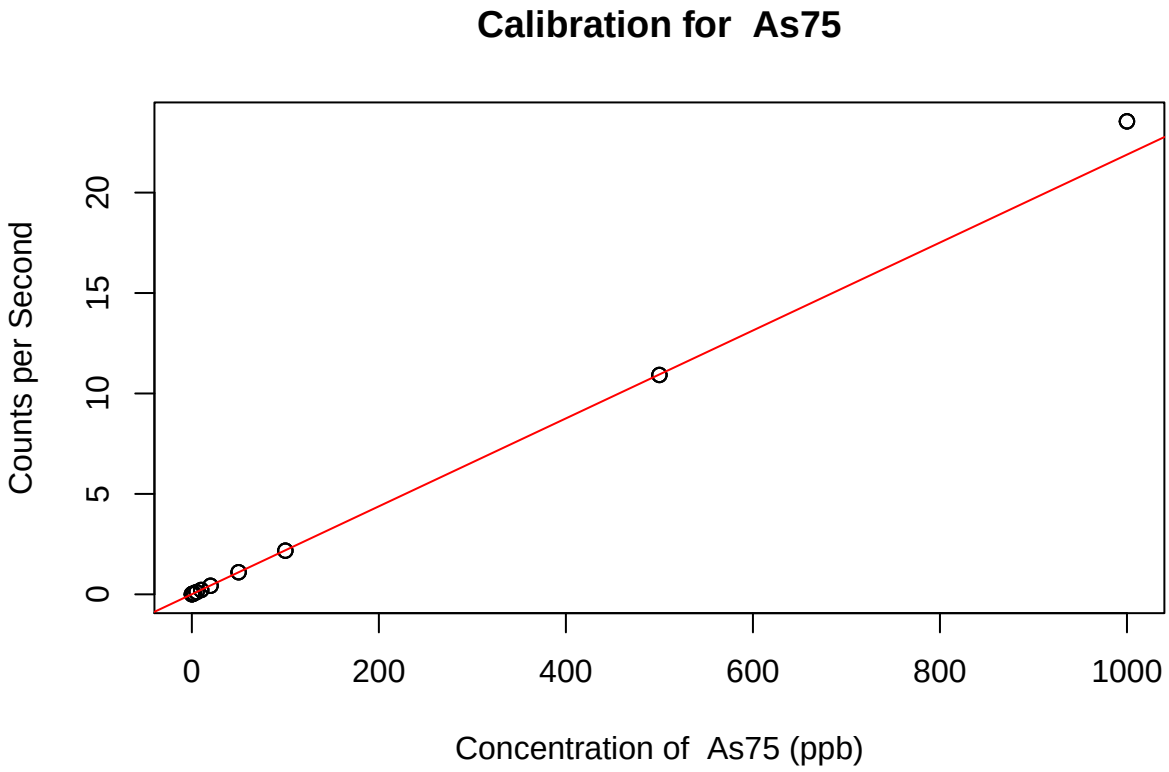
```
metals_analyzed
```

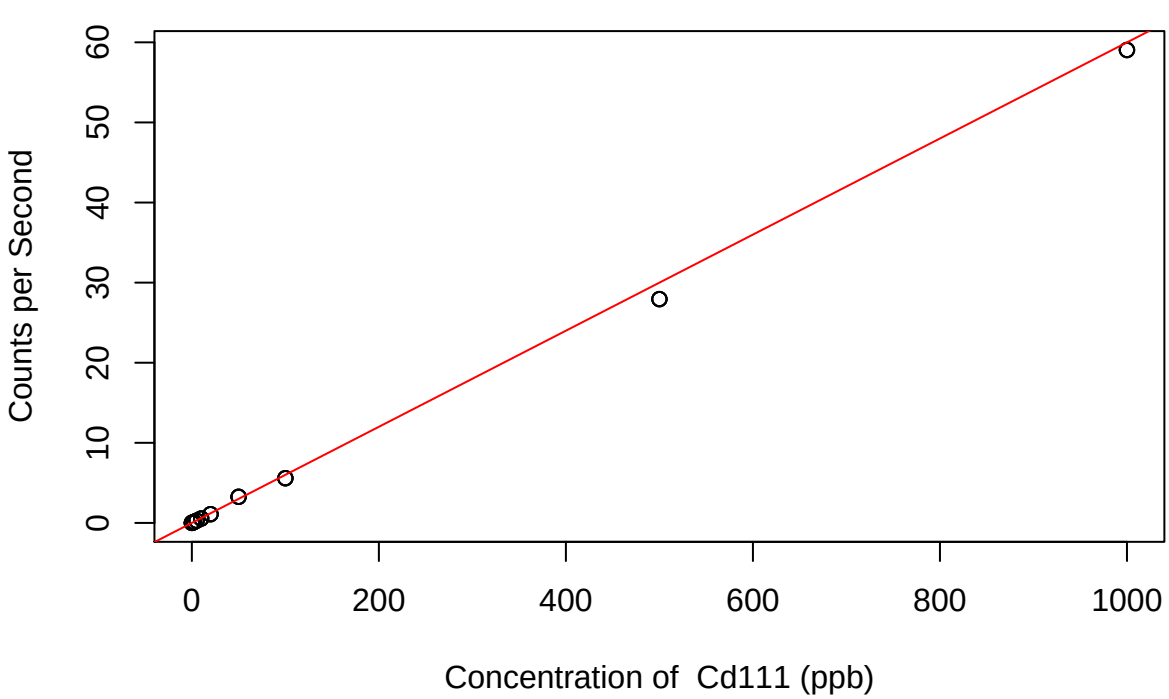
```
## [1] Cr53  Pb208 As75  Cd111 Cr52  Cd114
## Levels: As75 Cd111 Cd114 Cr52 Cr53 Pb208
```

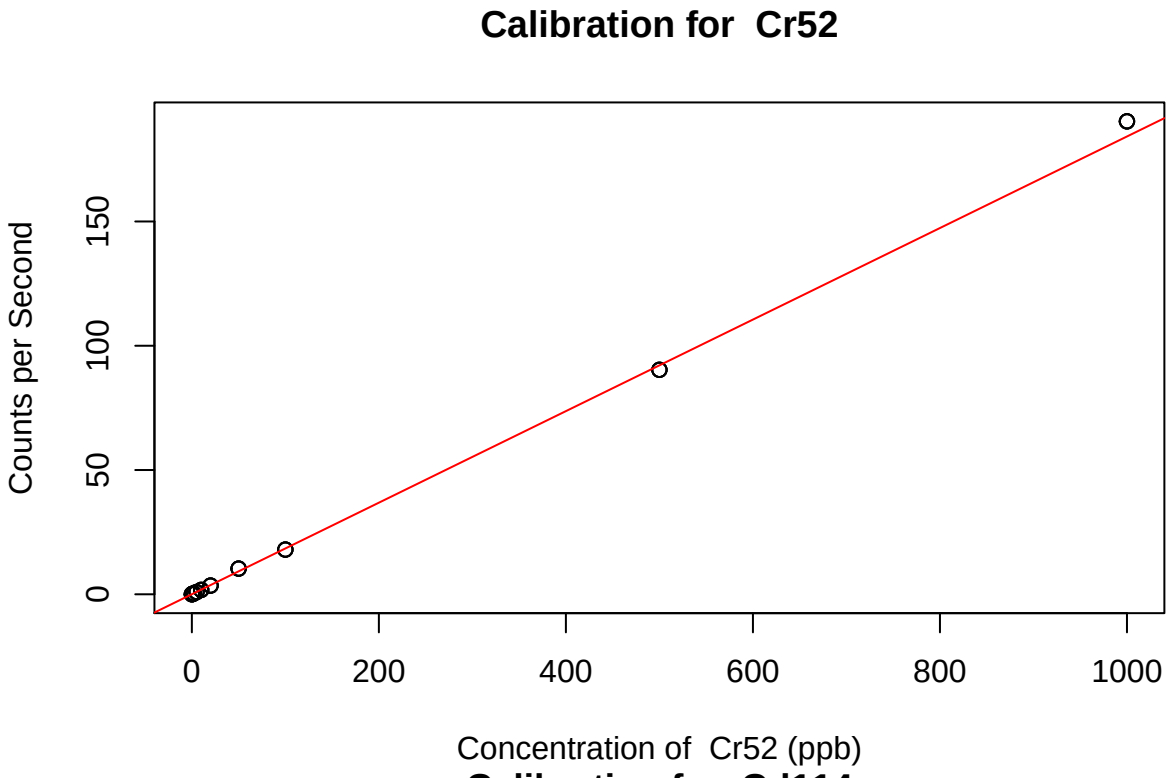
```
ICPMS_cal <- NULL
for (unique_metal in metals_analyzed) {
  #filtering for a single metal then selecting variables of interest
  cal <- ICPMS %>%
    filter(type == "Cal1" | type == "Cal2" | type == "Cal3") %>%
    filter(metal == unique_metal) %>%
    select(concentration, cps, rsd)
  #weighted linear regression
  w <- 1/(cal$cps*cal$rsd)^2
  model <- lm(cal$cps ~ cal$concentration, weights = w)
  #pulling out relevant info from model
  slope <- model$coefficients[2]
  intercept <- model$coefficients[1]
  slope_std <- summary(model)$coefficients[2,2]
  intercept_std <- summary(model)$coefficients[1,2]
  #plotting cal curve
  plot(cal$cps ~ cal$concentration,
       xlab = paste("Concentration of ", unique_metal, "(ppb)"),
       ylab = "Counts per Second") +
    abline(model, col = "red") +
    title(paste("Calibration for ", unique_metal))
  #storing info from calibration curve
  equation <- tibble(metal = unique_metal, slope, slope_std, intercept, intercept_std)
  ICPMS_cal <- rbind(ICPMS_cal, equation)
}
```

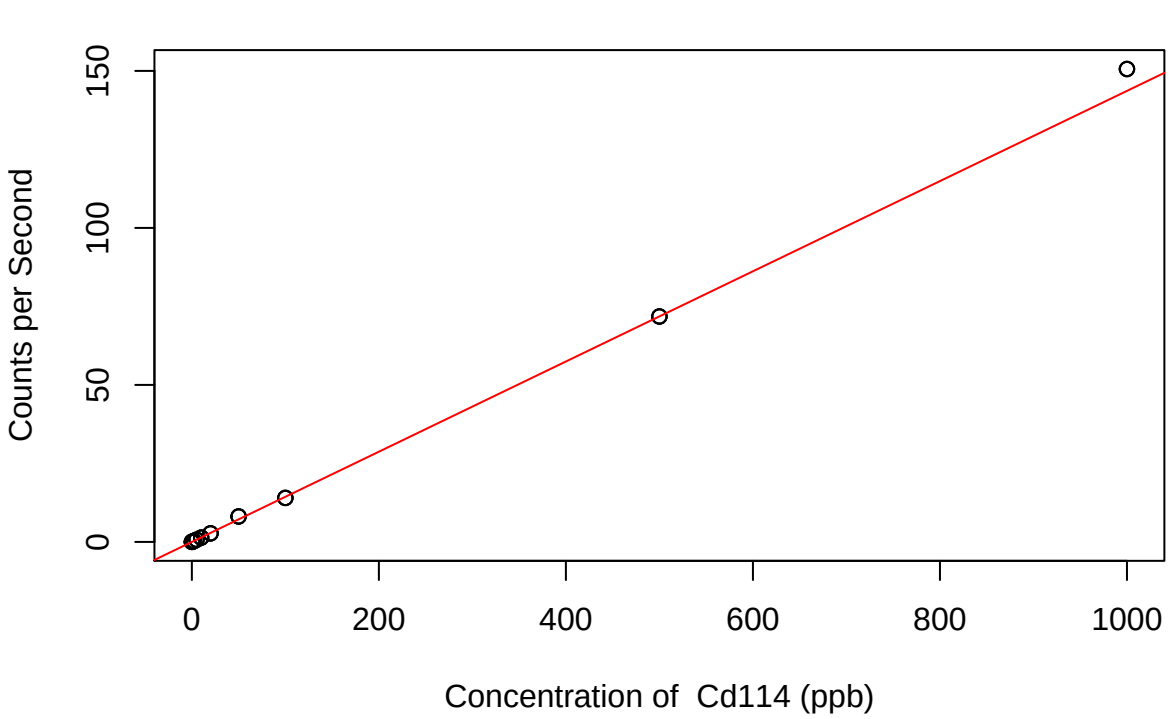

Calibration for Cr53
Counts per Second
0
5
10
15
20
0
200
400
600
800
1000
Concentration of Cr53 (ppb)


Calibration for Pb208
Counts per Second
0
10
20
30
40
50
60
0
200
400
600
800
1000
Concentration of Pb208 (ppb)


Calibration for As75
Counts per Second
0
5
10
15
20
0
200
400
600
800
1000
Concentration of As75 (ppb)


Calibration for Cd111
Counts per Second
0
10
20
30
40
50
60
0
200
400
600
800
1000
Concentration of Cd111 (ppb)






```
ICPMS_cal
```

```
## # A tibble: 6 x 5
##   metal  slope slope_std intercept intercept_std
##   <chr>  <dbl>     <dbl>     <dbl>         <dbl>
## 1 Cr53  0.0219 0.000161  0.00199       0.00115
```

```
## 2 Pb208 0.0620 0.000216  0.00736       0.000269
## 3 As75  0.0219 0.0000284 0.00522       0.000160
## 4 Cd111 0.0600 0.000858  0.0000272     0.0000887
## 5 Cr52  0.184  0.00204   0.0150        0.00713
## 6 Cd114 0.144  0.00141   0.0000761     0.0000750
```

```r
#inputs: unique_site (as a character)
#outputs: concentration vector

sample_analysis <- function(unique_site){
  concentration_data <- NULL
  for (unique_metal in metals_analyzed){
    sample <- filter(ICPMS, metal == unique_metal, site == unique_site)
    data <- NULL
    #iterating through each sample in unique_metal
    for (ID in sample$sample_key) {
      sample_data <- filter(sample, sample_key == ID)
      cal <- filter(ICPMS_cal, metal == unique_metal)
      #sample analysis
      m <- cal$slope
      b <- cal$intercept
      y <- sample_data$cps
      b_e <- cal$intercept_std
      m_e <- cal$slope_std
      x <- (y-b)/m #The units are dependent on the calibration standards (Kg/mL)
      RSD <- ((sample_data$rsd/100)*sample_data$cps)
      CPS <- sample_data$cps
      #error propagation
      e_yb <- sqrt((RSD)^2 + (b_e)^2) #error in y-b from calibration
      yb <- CPS - b
      e_x <- x*sqrt((e_yb/yb)^2 +(m_e/m)^2) #error in x from calibration
      #storing data for sites that are not method blanks
      data <- rbind(data, data.frame(sample_key = ID, x, e_x))
      if (unique_site != "MB"){
        concentration_data <- data_frame(sample_key = sample_data$sample_key,
                                         analyst = sample_data$analyst,
                                         metal = unique_metal,
                                         site = unique_site,
                                         conc_dil = x,
                                         conc_dil_error = e_x) %>%
          rbind(concentration_data)
      }
    }
    if (unique_site == "MB"){
      x <- mean(data$x)
      e_x <- sd(data$x)
      concentration_data <- data_frame(metal = unique_metal,
                                       site = unique_site,
                                       conc_dil = x,
                                       conc_dil_error = e_x) %>%
        rbind(concentration_data)
    }
  }
  return(concentration_data)
```

```
}
```

```
#inputs: a function
#outputs: a data frame with the function outputs from each site

run_sites <- function(Function){
  value <- NULL
  for (site in sample_sites){
    site_value <- Function(site)
    value <- rbind(site_value, value)
  }
  return(value)
}
```

```
MB <- sample_analysis("MB") #(ug/kg)
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
uncor_sample <- run_sites(sample_analysis) #values do not account for dilutions (ug/kg)

MB
```

```
## # A tibble: 6 x 4
##   metal site  conc_dil conc_dil_error
##   <chr> <chr>    <dbl>          <dbl>
## 1 Cd114 MB     0.00657        0.00674
## 2 Cr52  MB     0.535          0.0570
## 3 Cd111 MB     0.00511        0.00560
## 4 As75  MB     0.0475         0.0550
## 5 Pb208 MB     0.455          0.739
## 6 Cr53  MB     0.543          0.0637
```

```
uncor_sample
```

```
## # A tibble: 204 x 6
##    sample_key analyst metal site  conc_dil conc_dil_error
##         <int> <fct>   <chr> <chr>    <dbl>          <dbl>
##  1         56 AVM     Cd114 A       0.140         0.00279
##  2         54 LAK     Cd114 A       0.0638        0.00126
##  3         48 AH      Cd114 A       0.0788        0.00216
##  4         46 LML     Cd114 A       0.0864        0.00253
##  5         56 AVM     Cr52  A       6.74          0.139
##  6         54 LAK     Cr52  A       6.92          0.0969
##  7         48 AH      Cr52  A      12.1           0.151
##  8         46 LML     Cr52  A       9.56          0.133
##  9         56 AVM     Cd111 A       0.0507        0.00537
## 10         54 LAK     Cd111 A       0.0399        0.00273
## # ... with 194 more rows
```

```
sample_data_mb <- NULL

for (unique_metal in metals_analyzed){
  MB_metal <- filter(MB, metal == unique_metal)
  sample_metal <- filter(uncor_sample, metal == unique_metal)
```

```
  conc_dil_blanked <- sample_metal$conc_dil - MB_metal$conc_dil

  #error propagation: subtraction of MB
  conc_dil_blanked_error <- sqrt((sample_metal$conc_dil_error)^2 +
                                 (MB_metal$conc_dil_error)^2)

  sample_data_mb <- sample_metal %>%
    mutate(conc_dil_blanked, conc_dil_blanked_error) %>%
    rbind(sample_data_mb)
}

sample_data_mb
```

```
## # A tibble: 204 x 8
##    sample_key analyst metal site  conc_dil conc_dil_error conc_dil_blanked
##         <int> <fct>   <chr> <chr>    <dbl>          <dbl>            <dbl>
##  1         56 AVM     Cd114 A       0.140        0.00279            0.133
##  2         54 LAK     Cd114 A       0.0638       0.00126            0.0573
##  3         48 AH      Cd114 A       0.0788       0.00216            0.0722
##  4         46 LML     Cd114 A       0.0864       0.00253            0.0798
##  5         58 SS      Cd114 B       0.0848       0.00117            0.0782
##  6         52 MF      Cd114 B       0.149        0.00166            0.143
##  7         45 KAD     Cd114 B       0.116        0.00498            0.109
##  8         41 LF      Cd114 B       0.146        0.00370            0.139
##  9         40 AB      Cd114 B       0.0798       0.000987           0.0732
## 10          6 MRMJ    Cd114 C       0.0818       0.00187            0.0752
## # ... with 194 more rows, and 1 more variable: conc_dil_blanked_error <dbl>
```

```
#error propagation
vol_e <- 1
mass_e <- 0.001
dil_1010_e <- sqrt(1^2 + 10^2)
dil_e <- sqrt((dil_1010_e/1010)^2 + (1/10)^2) #error in 101 dilution factor

#correct for dilution and propagate error
sample_data <- merge(ICPMS, sample_data_mb) %>%
  unique() %>%
  mutate(conc_blanked = conc_dil_blanked*(total_volume/1000)/(mass_of_soil/1000)*101,
         conc_blanked_error = conc_blanked *
           sqrt((conc_dil_blanked_error/conc_dil_blanked)^2 +
           (dil_e/101)^2 +
           (mass_e/mass_of_soil)^2 +
           (vol_e/total_volume)^2),
         conc_unblanked = conc_dil*(total_volume/1000)/(mass_of_soil/1000)*101,
         conc_unblanked_error = conc_unblanked*
           sqrt((conc_dil_error/conc_dil)^2 +
                  (dil_e/101)^2 +
                  (mass_e/mass_of_soil)^2 +
                  (vol_e/total_volume)^2)) %>%
  select(-concentration,
         -type,
         -mass_of_soil,
         -total_volume,
         -cps,
```

```
         -rsd,
         -conc_dil_blanked,
         -conc_dil_blanked_error,
         -conc_dil,
         -conc_dil_error)
```

```
rm(list = ls()[!ls() %in% c("ICPMS", "sample_data")])
```

```
sample_conc <- sample_data %>%
  filter(site != "QC") %>%
  group_by(metal, site) %>%
  summarize(conc_mean = mean(conc_blanked),
            conc_sd = sd(conc_blanked),
            count = n()) %>%
  mutate(error = qnorm(0.975)*conc_sd/sqrt(count),
         lower = conc_mean - error,
         upper = conc_mean + error,
         mass_frac = conc_mean/1000) %>%
  select(-count,
         -error)
```

```
## `summarise()` regrouping output by 'metal' (override with `.groups` argument)
```

```
sample_conc_icpms <- sample_conc %>%
  filter(metal == "Cr52")

write.csv(sample_conc, "~/chem313/313_icpms/data/sample_conc_allmetals.csv")
write.csv(sample_conc_icpms, "~/chem313/313_icpms/data/sample_icpms.csv")
```

```
qc_conc <- sample_data %>%
  filter(site == "QC") %>%
  group_by(metal) %>%
  summarize(conc_mean = mean(conc_unblanked),
            conc_sd = sd(conc_unblanked),
            count = n()) %>%
  mutate(error = qnorm(0.975)*conc_sd/sqrt(count),
         lower_ci = conc_mean - error,
         upper_ci = conc_mean + error,
         mass_frac = conc_mean/1000,
         per_recovery = case_when(
           metal == "Cd111" | metal == "Cd114" ~ mass_frac/2.94*100,
           metal == "Cr52" | metal == "Cr53" ~ mass_frac/121.9*100,
           metal == "Pb208" ~ mass_frac/150*100
         )) %>%
  select(-count,
         -error)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```