Week2ICPMSLab

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──

## ✓ ggplot2 3.3.2 ✓ purrr 0.3.4  
## ✓ tibble 3.0.4 ✓ dplyr 1.0.2  
## ✓ tidyr 1.1.2 ✓ stringr 1.4.0  
## ✓ readr 1.4.0 ✓ forcats 0.5.0

## Warning: package 'ggplot2' was built under R version 3.6.2

## Warning: package 'tibble' was built under R version 3.6.2

## Warning: package 'tidyr' was built under R version 3.6.2

## Warning: package 'readr' was built under R version 3.6.2

## Warning: package 'purrr' was built under R version 3.6.2

## Warning: package 'dplyr' was built under R version 3.6.2

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(reshape2)

## Warning: package 'reshape2' was built under R version 3.6.2

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

ICPMS<-read.csv("~/ICPMSArielle/Data/ICPMS\_tidy\_example.csv")

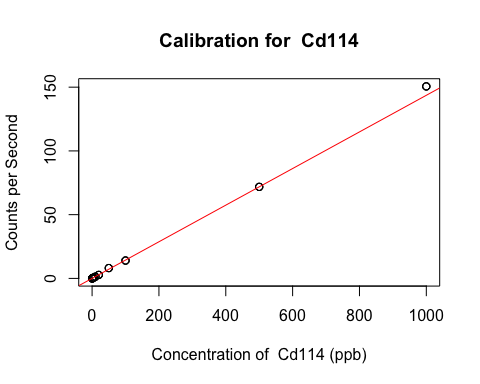
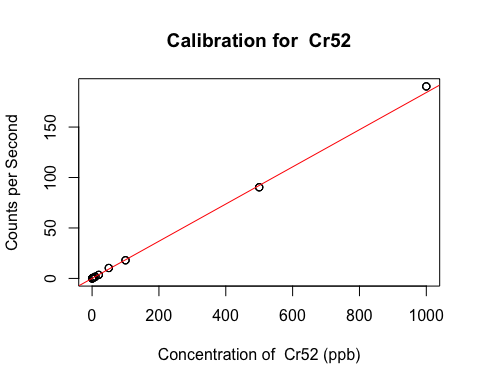
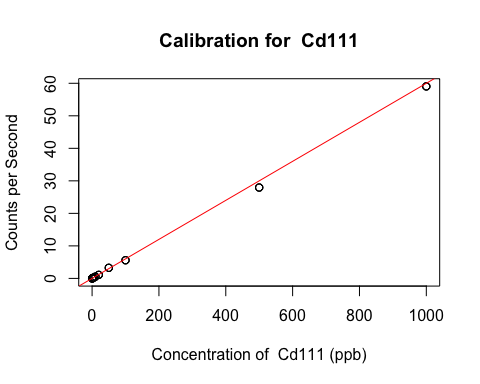
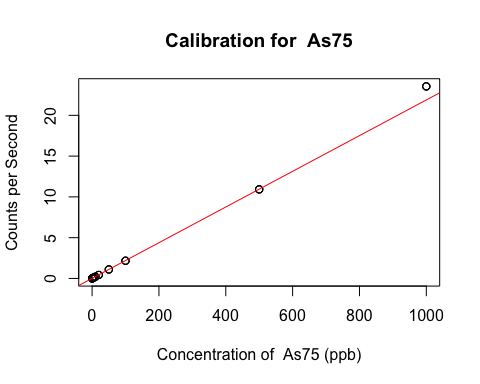
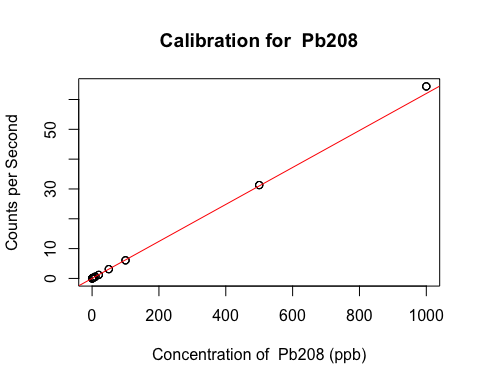
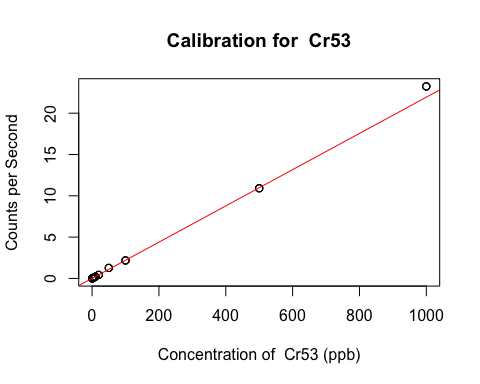
sample\_site<- unique(filter(ICPMS, site!= "MB", site!="")$site)  
#excluding method blank and quality control from the list of sites  
metals\_analyzed<- unique(ICPMS$metal)  
  
#preveiw the lists to check for potential issues  
sample\_site

## [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)  
}



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

remove (equation, cal, slope, slope\_std, intercept, intercept\_std, w, model, unique\_metal)

#function to analyze the samples and add a comment with the inputs and outputs of the functions  
#inputs: unique\_site (as a character, ex. "A")  
#outputs: concentration vector  
  
sample\_analysis<-function(unique\_site){  
 #initiated "for loops" to select a specific sample and a specific metal  
 concentration\_data <- NULL  
 for (unique\_metal in metals\_analyzed){  
 sample <- filter(ICPMS, metal == unique\_metal, site == unique\_site)  
 data <- NULL  
   
 for (ID in sample$sample\_key) {  
 sample\_data <- filter(sample, sample\_key == ID)  
 cal <- filter(ICPMS\_cal, metal == unique\_metal)  
 #convert the cps readings into concentrations  
 #sample analysis  
 m <- cal$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   
 #propagate the error in the concetration  
 #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  
 #if the "site is not eh method blank, store the concentration data  
   
 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 the site is the method blank, average the concentrations and then store the 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)  
  
   
}

#create a function that runs a different function on each of the soil sample sites  
  
#inputs: a function  
#outputs: a data frame with the function outputs from each site  
  
run\_sites<-function(Function){  
 value <- NULL  
 for(sites in sample\_site){  
 site\_value <- Function(sites)  
 value <-rbind(site\_value, value)  
 }  
 return(value)  
}

#analyze the method blank and all the samples  
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 dilutns  
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

#correct for the method blank and perform error propagation as needed  
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 propogation: subtraticoon 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>

#define the dilution factors and measuremnt errors  
#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 dilutions 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,  
 #101 is the factor diluted by at OHSU to make the solutions dilute enought to run the ICPMS on  
 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, #removing uneccesary columns  
 -type,  
 -mass\_of\_soil,  
 -total\_volume,  
 -cps,  
 -rsd,  
 -conc\_dil\_blanked,  
 -conc\_dil\_blanked\_error,  
 -conc\_dil,  
 -conc\_dil\_error)

#cleaning up the environment  
rm(list=ls()[!(ls()%in% c("ICPMS", "sample\_data"))])

#now i am going to take the averages of all the metals and create a new data frame with that, not sure if I will be able to make a fucntion, i may just do it by hand  
  
###Just testing this method, not really what I need  
##sample\_data2<-aggregate(sample\_data[, 6:9], list(sample\_data$metal), mean)  
  
#just grouping by metals for avgs, no qc site or mb  
#similar to gillians code  
AVG\_concMETAL<- sample\_data%>%  
 filter(site!= "QC")%>%  
 group\_by(metal)%>%  
 summarise(mean\_conc = mean(conc\_blanked), sd\_conc = sd(conc\_blanked), n = n()) %>%  
 mutate(CI = qnorm(0.975)\*sd\_conc/sqrt(n),  
 lower\_ci = mean\_conc - CI,  
 upper\_ci = mean\_conc + CI)

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

#avg of metla per site  
AVG\_concSITE<- sample\_data%>%  
  
 group\_by(metal,site)%>%  
 summarise(mean\_conc = mean(conc\_blanked), sd\_conc = sd(conc\_blanked), n = n()) %>%  
 mutate(CI = qnorm(0.975)\*sd\_conc/sqrt(n),  
 lower\_ci = mean\_conc - CI,  
 upper\_ci = mean\_conc + CI)

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

#QC site avgs  
AVG\_concQC<- sample\_data%>%  
 filter(site== "QC")%>%  
 group\_by(metal)%>%  
 summarise(mean\_conc = mean(conc\_blanked), sd\_conc = sd(conc\_blanked), n = n()) %>%  
 mutate(CI = qnorm(0.975)\*sd\_conc/sqrt(n),  
 lower\_ci = mean\_conc - CI,  
 upper\_ci = mean\_conc + CI)

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

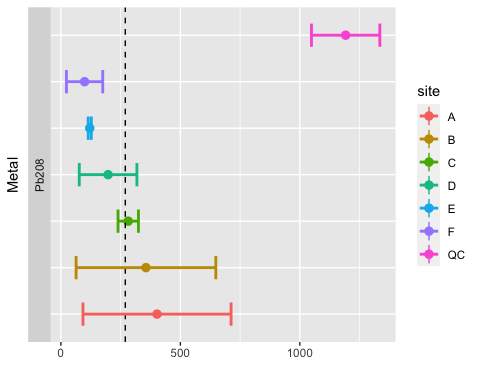
AVG\_concSITE <- AVG\_concSITE %>%  
 mutate(CI = CI/1000)  
write.csv(AVG\_concSITE, "~/ICPMSArielle/Data/AVG\_concSITE.csv")

AVG\_concSITE <- AVG\_concSITE %>%  
 mutate(mean\_conc = mean\_conc/1000,  
 sd\_conc = sd\_conc/1000,  
 lower\_ci = lower\_ci/1000,  
 upper\_ci = upper\_ci/1000,   
 CI = CI/1000  
 )  
  
AVG\_concMETAL <- AVG\_concMETAL %>%  
 mutate(mean\_conc = mean\_conc/1000,  
 sd\_conc = sd\_conc/1000,  
 lower\_ci = lower\_ci/1000,  
 upper\_ci = upper\_ci/1000,   
 CI = CI/1000  
 )  
AVG\_concSITE\_PB <- AVG\_concSITE %>%  
 filter(metal == "Pb208")  
  
AVG\_concSITE\_AS <- AVG\_concSITE %>%  
 filter(metal == "As75")  
  
AVG\_concSITE\_CD111 <- AVG\_concSITE %>%  
 filter(metal == "Cd111")  
  
AVG\_concSITE\_CD114 <- AVG\_concSITE %>%  
 filter(metal == "Cd114")  
  
AVG\_concSITE\_Cr52 <- AVG\_concSITE %>%  
 filter(metal == "Cr52")  
  
AVG\_concSITE\_Cr53 <- AVG\_concSITE %>%  
 filter(metal == "Cr53")

##plots  
  
p = ggplot(data=AVG\_concSITE\_PB,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =269.5352927, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

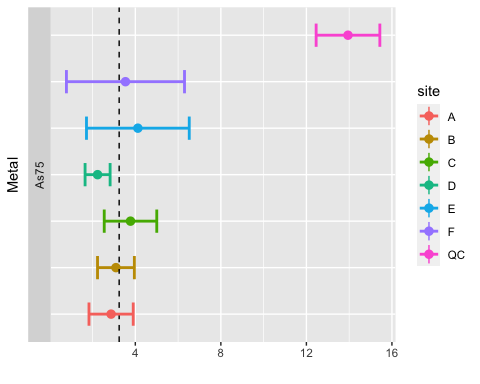
p



p = ggplot(data=AVG\_concSITE\_AS,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =3.2445059, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

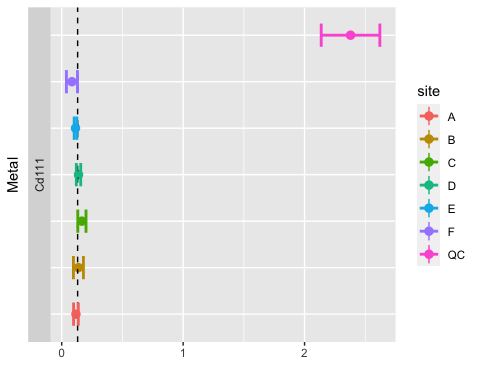
p



p = ggplot(data=AVG\_concSITE\_CD111,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =0.1310686, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

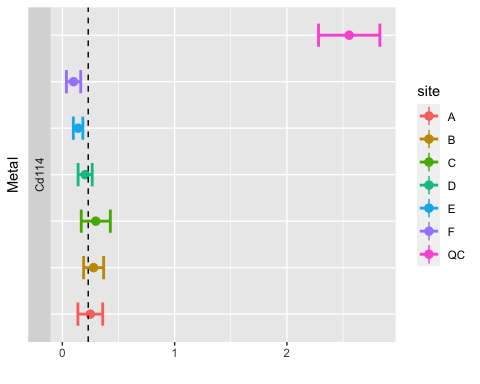
p



p = ggplot(data=AVG\_concSITE\_CD114,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =0.2303492, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

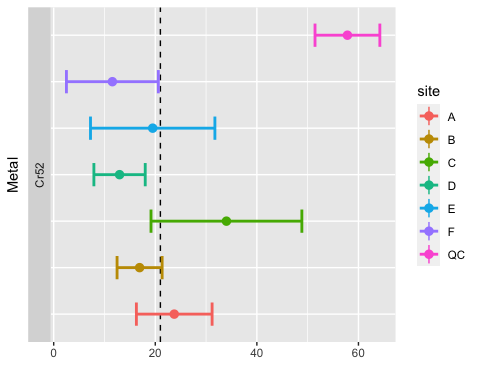
p



p = ggplot(data=AVG\_concSITE\_Cr52,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =21.0045338, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

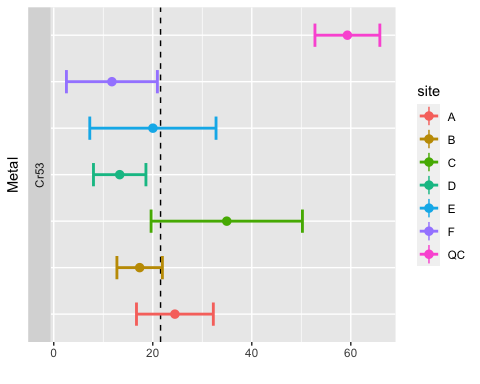
p



p = ggplot(data=AVG\_concSITE\_Cr53,  
 aes(x = site,y = mean\_conc, ymin = lower\_ci, ymax = upper\_ci ))+  
 geom\_pointrange(aes(col=site))+  
 geom\_hline(aes(fill=site),yintercept =21.5752868, linetype=2)+  
 xlab('Metal')+ ylab("")+  
 geom\_errorbar(aes(ymin=lower\_ci, ymax=upper\_ci,col=site),width=0.5,cex=1)+   
 facet\_wrap(~metal,strip.position="left",nrow=9,scales = "free\_y") +  
 theme(plot.title=element\_text(size=16,face="bold"),  
 axis.text.y=element\_blank(),  
 axis.ticks.y=element\_blank(),  
)+ coord\_flip()

## Warning: geom\_hline(): Ignoring `mapping` because `yintercept` was provided.

p



#Lets create somehting they have in commone to merge the two tables together  
AVG\_concQC$site <- "QC"  
AVG\_concMETAL$site <- "NA"  
#merging them  
AVG\_conc=full\_join(AVG\_concMETAL,AVG\_concQC,by=c("metal", "mean\_conc", "sd\_conc", "n", "CI", "lower\_ci", "upper\_ci", "site"))  
write.csv(AVG\_conc, "~/ICPMSArielle/Data/AVG\_conc.csv")  
write.csv(AVG\_concSITE, "~/ICPMSArielle/Data/AVG\_concSITE.csv")

##making some nicer loooking dataframes for export  
AVG\_concSITE\_TRANS <- AVG\_concSITE %>%  
dplyr::select(site, mean\_conc, metal,)  
  
AVG\_concSITE\_TRANS\_SD <- AVG\_concSITE %>%  
dplyr::select(site, sd\_conc, metal,)  
AVG\_concSITE\_TRANS <- melt(AVG\_concSITE\_TRANS, id.vars = c("site", "metal"))  
  
AVG\_concSITE\_TRANS\_SD <- melt(AVG\_concSITE\_TRANS\_SD, id.vars = c("site", "metal"))  
AVG\_concSITE\_TRANS\_SD <- dcast(AVG\_concSITE\_TRANS\_SD, site ~ metal)  
  
AVG\_concSITE\_TRANS <- dcast(AVG\_concSITE\_TRANS, site ~ metal)  
write.csv(AVG\_concSITE\_TRANS, "~/ICPMSArielle/Data/AVG\_concSITE\_TRANS.csv")  
write.csv(AVG\_concSITE\_TRANS\_SD, "~/ICPMSArielle/Data/AVG\_concSITE\_TRANS\_SD.csv")

#I found this function on stack exchange  
#https://stats.stackexchange.com/questions/30394/how-to-perform-two-sample-t-tests-in-r-by-inputting-sample-statistics-rather-tha  
  
# m1, m2: the sample means  
# s1, s2: the sample standard deviations  
# n1, n2: the same sizes  
# m0: the null value for the difference in means to be tested for. Default is 0.   
# equal.variance: whether or not to assume equal variance. Default is FALSE.   
t.test2 <- function(m1,m2,s1,s2,n1,n2,m0=0,equal.variance=FALSE)  
{  
 if( equal.variance==FALSE )   
 {  
 se <- sqrt( (s1^2/n1) + (s2^2/n2) )  
 # welch-satterthwaite df  
 df <- ( (s1^2/n1 + s2^2/n2)^2 )/( (s1^2/n1)^2/(n1-1) + (s2^2/n2)^2/(n2-1) )  
 } else  
 {  
 # pooled standard deviation, scaled by the sample sizes  
 se <- sqrt( (1/n1 + 1/n2) \* ((n1-1)\*s1^2 + (n2-1)\*s2^2)/(n1+n2-2) )   
 df <- n1+n2-2  
 }   
 t <- (m1-m2-m0)/se   
 dat <- c(m1-m2, se, t, 2\*pt(-abs(t),df))   
 names(dat) <- c("Difference of means", "Std Error", "t", "p-value")  
 return(dat)   
}

#comaparing between metals  
t.test2(13945.41904, 3244.505905,2633.02005, 1335.144796,12, 22)

## Difference of means Std Error t p-value   
## 1.070091e+04 8.116407e+02 1.318430e+01 2.419866e-09

t.test2(2378.70526,131.068641, 426.7552319, 40.39792594, 12, 22 )

## Difference of means Std Error t p-value   
## 2.247637e+03 1.234943e+02 1.820032e+01 1.278064e-09

t.test2(2555.015649, 230.3492002,482.7139834,116.6396999, 12, 22 )

## Difference of means Std Error t p-value   
## 2.324666e+03 1.415490e+02 1.642305e+01 1.922390e-09

t.test2(57831.13336,21004.53376, 12140.64641, 482.7139834, 12, 22 )

## Difference of means Std Error t p-value   
## 3.682660e+04 3.506213e+03 1.050324e+01 4.449920e-07

t.test2(59315.0092,21575.28679, 11585.50655, 12516.89764,12, 22 )

## Difference of means Std Error t p-value   
## 3.773972e+04 4.278647e+03 8.820481e+00 4.822628e-09

t.test2(269535.2927, 1191817.581,222762.7727,252957.7849, 22,12 )

## Difference of means Std Error t p-value   
## -9.222823e+05 8.710859e+04 -1.058773e+01 9.867960e-10

t.test2(1191817.581, 269535.2927,252957.7849,222762.7727, 12,22 )

## Difference of means Std Error t p-value   
## 9.222823e+05 8.710859e+04 1.058773e+01 9.867960e-10

##Now lets compare the site c cr to each other  
  
#cr52  
t.test2(3.400937e+01, 21.0045338, 1.694230e+01, 12.14064641, 5,22 )

## Difference of means Std Error t p-value   
## 13.0048362 8.0067529 1.6242335 0.1655444

#Cr53  
t.test2(3.494743e+01,21.5752868, 1.743691e+01, 12.51689764, 5, 22)

## Difference of means Std Error t p-value   
## 13.3721432 8.2420054 1.6224380 0.1658855