High-Mg Carbonate XRD data analysis with Xd,Xq; 0-5% calcite

John M. Jaeger

July 10, 2017

## Code Description

This code is designed to import and analyze X-ray diffraction data collected at the University of Florida on a Rigaku Ultima IV diffractometer. The data are from carbonate mineral standards created to calibrate the diffractometer to provide quantitative measures of low- and high-Mg calcite.

The general calibration approach follows: Sepulcre, S., Durand, N., and Bard, E., 2009, Global and Planetary Change: Global And Planetary Change, v. 66, p. 1–9, doi: 10.1016/j.gloplacha.2008.07.008.

Establishing detection and quantificiation limits follows: Evard, H., Kruve, A., and Leito, I., 2016, Tutorial on estimating the limit of detection using LC-MS analysis, part I: Theoretical review: Analytica Chimica Acta, v. 942, p. 23–39, doi: 10.1016/j.aca.2016.08.043.

Code generated by: John M. Jaeger Associate Professor 241 Williamson Hall P.O. Box 112120 Dept. of Geological Sciences University of Florida Gainesville FL 32611-2120, USA (352) 846-1381 ORCID ID# orcid.org/0000-0003-0248-489X <http://people.clas.ufl.edu/jmjaeger/>

## Load packages

# if necessary uncomment and install packages.  
# install.packages("dplyr")  
# install.packages("knitr")  
library(dplyr)  
require(knitr)  
library(ezknitr)  
library(chemCal)  
library(reshape2)  
library(RColorBrewer)

## Install functions

rnorm2 <- function(n,mean,sd) { mean+sd\*scale(rnorm(n)) }

## Import the data

### Load peak area values for calibration standards

Low-Mg calcite peak at 29.4° 2-theta; Aragonite peak at 26.2° 2-theta (copper anode)

# load the End-member Dataset and give the dataset a shorter name  
  
XRD.data <- read.table("../raw\_data/UF\_HMC\_data\_20170515.csv",header=T,sep=",")   
XRDdata <- tbl\_df(XRD.data[1:18,])  
XRDdata <- as.data.frame(XRDdata)  
n = 3 #number of replicates of each calibration sample

### Establish ratios for aragonite blanks

We use peak areas for aragonite blank samples in the regression model (See Edvard et al. 2016a). Net peak areas are determined by subtracting an average background, which is the mean of eight measurements on either side of peak. First three blank samples are used here, but user can choose from any 24 total blank measurements.

#Load aragonite blank data  
aragonite.data <- read.table("../raw\_data/UF\_pure\_AC\_aragonite\_XRD.csv",header=T,sep=",")   
aragonitedata <- tbl\_df(aragonite.data)  
  
# HMC peak region spans from 29.20-30.34° two-theta (Cu) based on 0.3% HMC standard  
HMC\_region <- aragonitedata[371:485,]  
Cbkgdleft <- aragonitedata[363:370,]  
Cbkgdright <- aragonitedata[486:493,]  
Cbkgd.three <- rbind(Cbkgdleft[,2:4],Cbkgdright[,2:4])  
Cmelt.bkgd <- melt(Cbkgd.three)  
Cmeanbkgd <- mean(Cmelt.bkgd[,2])  
HMC.net <- HMC\_region[,2:4]-Cmeanbkgd  
HMC.netPA <- colSums(HMC.net)  
  
# Aragonite peak region spans from 25.58-26.60° two-theta (Cu)   
Arag\_region <- aragonitedata[9:111,]  
Abkgdleft <- aragonitedata[1:8,]  
Abkgdright <- aragonitedata[112:119,]  
Abkgd.three <- rbind(Abkgdleft[,2:4],Abkgdright[,2:4])  
Amelt.bkgd <- melt(Abkgd.three)  
Ameanbkgd <- mean(Amelt.bkgd[,2])  
Arag.net <- Arag\_region[,2:4]-Ameanbkgd  
Arag.netPA <- colSums(Arag.net)  
blank <- c(0,0,0)  
HMC.blank <- rbind(blank,HMC.netPA,Arag.netPA)  
HMCblank <- t(HMC.blank)  
colnames(HMCblank) <- c("HMCconc", "HMC\_area","Aragonite\_area")  
  
XRDdata.all <- rbind(HMCblank,XRD.data)#combine blank data with calibration data  
XRDdata.HMC <- XRDdata.all[1:21,] # use only data from 0-5% HMC  
calib.levels <- as.double(nrow(XRDdata.HMC))/n #number of calibration concentration levels

### convert peak areas to peak area ratios

AreaRatio <- XRDdata.HMC$HMC\_area/(XRDdata.HMC$HMC\_area+XRDdata.HMC$Aragonite\_area)  
HMCconc <- XRDdata.HMC$HMCconc  
HMC.lodata <- data.frame(HMCconc,AreaRatio)  
HMClo <- tbl\_df(HMC.lodata)

## Create ordinary least squares (OLS) regression model

We first develop an OLS linear calibration model for low-Mg calcite. We create a residual plot to examine for homoscedasticity and whether a linear model is an appropraite fit (see Edavrd et al., 2016 for rationale of these inital tests).

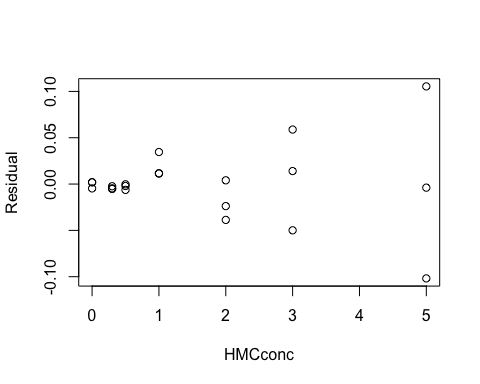
HMC.lm = lm(AreaRatio~ HMCconc, data=HMClo)  
vcov(HMC.lm)

## (Intercept) HMCconc  
## (Intercept) 1.587684e-04 -4.762246e-05  
## HMCconc -4.762246e-05 2.825061e-05

summary(HMC.lm)

##   
## Call:  
## lm(formula = AreaRatio ~ HMCconc, data = HMClo)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.101831 -0.005334 -0.002074 0.011295 0.105405   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.002322 0.012600 0.184 0.856   
## HMCconc 0.036027 0.005315 6.778 1.79e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0406 on 19 degrees of freedom  
## Multiple R-squared: 0.7074, Adjusted R-squared: 0.692   
## F-statistic: 45.94 on 1 and 19 DF, p-value: 1.79e-06

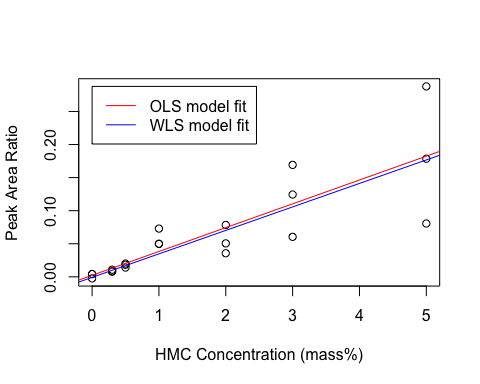
Bcalc = HMC.lm$coefficient[[2]]  
Acalc = HMC.lm$coefficient[[1]]  
Residual = AreaRatio - Bcalc \* HMCconc - Acalc  
# Data is plotted  
plot(HMCconc, Residual)

 Residuals show no trend and cluster around zero, so linear model is appropriate. Variance in residuals do have distinctive "funnel" shape, so data are considered heteroscedastic. OLS model is not approprate, so WLS is preferred. Results using an OLS approach are presented at end for comparision with WLS results.

## Create weighted least squares (WLS) regression model

We determine WLS calibration regression according to methods and R code of: Evard, H., Kruve, A., and Leito, I., 2016, Tutorial on estimating the limit of detection using LC-MS analysis, part I: Theoretical review: Analytica Chimica Acta, v. 942, p. 23–39, doi: 10.1016/j.aca.2016.08.043.

signal.data <- as.matrix(AreaRatio)  
signal.data1 <- matrix(nrow = calib.levels, ncol = n)  
conc.data1 <- matrix(nrow = calib.levels, ncol = 1)  
for(u in 1:calib.levels){  
 signal.data1[u,1] <- signal.data[3\*u-2]  
}  
for(u in 1:calib.levels){  
 signal.data1[u,2] <- signal.data[3\*u-1]  
}  
for(u in 1:calib.levels){  
 signal.data1[u,3] <- signal.data[3\*u]  
}  
for(u in 1:calib.levels){  
 conc.data1[u] <- HMCconc[3\*u-2]  
  
}  
  
Cin <- unlist(conc.data1)  
Y <- c(signal.data1[, 1:n])  
C <- rep(Cin, n)  
  
# Calculating weights for calibration samples  
 SdRow <- NULL  
 sdvalm <- NULL  
 mean.PAR <- NULL  
 for(k in 1:nrow(signal.data1)){  
 SdRow[k] <- sd(signal.data1[k, ])^-2  
 sdvalm[k] <- sd(signal.data1[k, ])  
 mean.PAR[k] <- mean(signal.data1[k, ])  
 }  
 Nsamp <- nrow(signal.data1) \* ncol(signal.data1)  
 sumCal <- sum(SdRow)  
 sumvar <- (sumCal) / (Nsamp)   
 w <- NULL  
 #weights are normalized to make unitless; see Edvard et al. 2016 for justification  
 for(v in 1:nrow(signal.data1)){  
 w[v] <- sd(signal.data1[v, ])^-2 / sumvar  
 }  
 W <- rep(w, n)#weights for WLS model  
   
  
 WLS <- lm(Y ~ C, weights = W)  
 bw\_calc = lm(Y ~ C, weights = W)$coeff[2]  
 aw\_calc = lm(Y ~ C, weights = W)$coeff[1]  
   
 sigmodel <- bw\_calc\*Cin+aw\_calc  
 plot(C,Y, ann=FALSE)  
 abline(aw\_calc, bw\_calc,col = "blue")# WLS fit  
 abline(Acalc,Bcalc, col="red")#OLS fit  
 title(xlab="HMC Concentration (mass%)")  
 title(ylab="Peak Area Ratio")  
 legend (0, max(Y), c("OLS model fit","WLS model fit"),col=c("red", "blue"),lty=c(1,1))

 The slope and intercept of the two regression lines are very slightly different, which will affect the resulting limit calculations that depend on these values. See text for further discussion.

## Establish limits of detection and quantification

We determine the limits of decision, detection, and quantification (Yc, Xd, Xq, respectively) from calibration regression according to: Burdge, J.R., MacTaggart, D.L., and Farwell, S.O., 1999, Realistic detection limits from confidence bands: J Chem Educ., v.76(3), p. 434-439  
and Zorn, M.E., Gibbons, R.D., and Sonzogni, W.C., 1997, Weighted least-squares approach to calculating limits of detection and quantification by modeling variability as a function of concentration: Analytical chemistry, v. 69, p. 3069–3075.

The decision limit (Lc or Yc as used here) is the peak area ratio above which the signal is deemed at a 95% probabilty to rise above noise. The intersection of this y-value with the regression line is the Xc, or critical level in the concentration domain; however, it is suggested by Edvard et al. (2016) that the crtiical level be expressed only in the signal domain. The limit of detection (Xd here but often refered to as LoD) is the detection limit in concentration units (Yd here in signal units, sometimes denoted as Ld).

# Equations for Yc and Xd used here are from Burdge, J.R., MacTaggart, D.L., and Farwell, S.O., 1999, Realistic detection limits from confidence bands: J Chem Educ., v.76(3), p. 434-439   
  
#If no blank replicates are measured, it is necessary to create a model of signal st deviation as a function of concentration, which is then used to model weights for intercept and unknown values  
stval.lm <- lm(sdvalm~poly(Cin,2,raw=TRUE)) #use if necessary to model St dev in calibration samples to assign weights; See Zorn et al. (1997)  
p <- 0 #parameters in model of st dev versus concentration, which is zero for this because we use measured values applied to ranges closest in % HMC. Use 2 for linear model or 3 for 2nd order polynomial; See Zorn et al. (1997)  
t.quantiles <- qt(c(.05, .95), Nsamp-p-2)# degrees of freedom in regression model  
tval <- t.quantiles[2]# degrees of freedom in regression model  
m <- 1# number of replicate measurements of unknown, usually 1 for establishing limits  
#See Burge et al. (1999) for explantion of variables below:  
xbarw <- sum(C\*W)/sum(W)  
Sxxw <- sum(W\*C^2)-((sum(C\*W))^2/sum(W))  
Syyw <- sum(W\*Y^2)-((sum(Y\*W))^2/sum(W))  
Sxyw <- sum(W\*Y\*C)-(sum(W\*C)\*sum(W\*Y))/sum(W)  
Swr <- sqrt((Syyw-bw\_calc\*Sxyw)/(Nsamp-2))  
SwrN <- Swr/sqrt(sum(W)/Nsamp)  
gw <- ((tval\*Swr)^2)/(bw\_calc^2\*Sxxw)  
  
Q <- sqrt((1/(m\*W[1]))+(1/sum(W))+xbarw^2/Sxxw)  
A <- ((tval\*Swr\*Q)/bw\_calc)-(xbarw\*gw)  
B <- xbarw\*gw\*(xbarw\*gw-(2\*tval\*Swr\*Q/bw\_calc))  
Cv <- 1/sum(W)+xbarw^2/Sxxw+gw/(m\*W[1])  
Yc.wls <- aw\_calc+tval\*Swr\*(sqrt(1/(m\*W[1])+(1/sum(W))+(xbarw^2/Sxxw)))# decision limit in signal units  
Xd.wls <- (A+(sqrt(B+(gw\*Sxxw\*(((1-gw)/(m\*W[2]))+Cv)))))/(1-gw) # limit of detection in concentration units. We do not iterate to find precise weight at Xd as recommended by Burdge et al. and simply assume the weight at the calibration point closest to estimate of Xd (i.e., W[2] here)  
  
# We can also esitmate limits of quantification  
Yq.wls <- 10\*(sqrt(1/SdRow[2]))+aw\_calc #limit of quantification in signal units; From Zorn et al. (1997), eqn. 23  
Xq.wls <- (Yq.wls-aw\_calc)/bw\_calc #limit of quantification in concentration units; From Zorn et al. (1997), eqn. 24  
  
  
cat("WLS limit of detection (%HMC): ", round(Xd.wls,digits = 1)," ")

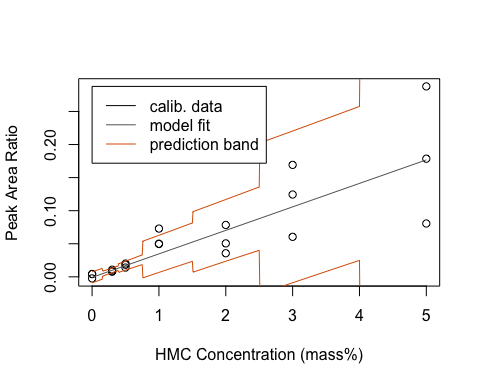
## WLS limit of detection (%HMC): 0.3

cat("WLS limit of quantification (%HMC): ", round(Xq.wls,digits = 1)," ")

## WLS limit of quantification (%HMC): 0.4

## Establish prediction bands for WLS calibration regression

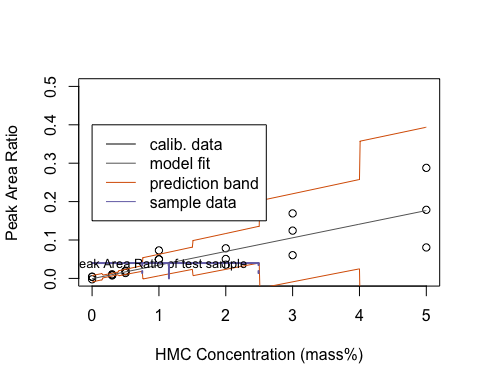
wls.predval <- matrix(nrow = calib.levels, ncol = m)  
  
HMC.model <- seq(0, 5, by=0.01)# set up a range of HMC values spanning limits at 0.01% HMC intervals  
PAR.model <- bw\_calc\*HMC.model+aw\_calc #model peak area ratio (PAR) as a function of HMC.model  
weight.modin <- cbind(Cin,w) # link calibration concentration measurement weights to concentrations  
  
#instead of modeling weights as a polynomial function of %HMC, we choose to set weights based on measured values. Weights for HMC.model values are assigned based on nearest measured concentration replicates.  
j <- length(PAR.model)  
wls.predvaltest <- matrix(nrow = j, ncol = 1)  
for(jv in 1:j){  
 lookvalue <- HMC.model[jv]  
 weightx <- which(abs(Cin-lookvalue)==min(abs(Cin-lookvalue)))  
 wls.predvaltest[jv,1] <- weight.modin[weightx[1],2]  
}  
Inverse.HMCdata <- cbind(HMC.model,PAR.model,wls.predvaltest)  
wls.predval2 <- matrix(nrow = j, ncol = m)  
  
#We develop 95% prediction band equation from Zorn et al., equation 13  
for(pv in 1:j){  
 wls.predval2[pv,1] <-tval\*Swr\*(sqrt(1/(m\*wls.predvaltest[pv,1])+(1/sum(W))+((Inverse.HMCdata[pv,1]-xbarw)^2/Sxxw)))  
}  
predband.uM <- PAR.model+wls.predval2  
predband.lM <- PAR.model-wls.predval2  
  
Inverse.HMCdata <- cbind(HMC.model,PAR.model,wls.predvaltest,wls.predval2,predband.uM,predband.lM)  
colnames(Inverse.HMCdata) <- c("HMC.model", "PAR.model","weight.model","uncert.model","upper.uncert","lower.uncert")  
cols<-brewer.pal(n=8,name="Dark2")  
  
plot(C,Y, ann=FALSE)  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,2], col = cols[8])  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,5], col = cols[2], lty=1)  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,6], col = cols[2], lty=1)  
#segments(0, Yc.wls, HMC.xupperC, unknown.PAR,col =cols[3],lty=1,lwd=2)  
title(xlab="HMC Concentration (mass%)")  
title(ylab="Peak Area Ratio")  
legend(0, max(Y), c("calib. data","model fit","prediction band"),col=c("black", cols[8],cols[2]), lty=c(1,1,1))



## Create an inverse model for establishing %HMC from a peak area ratio value

We create a model of fit and uncertainity to perform inverse modeling. We use the intercept of the P.A.R. with the regression line to determine the most likely %HMC value. We use the intercept of the P.A.R. with the prediction bands from the calibration curve to establish the 95% predition limits on the % HMC.

unknown.PAR <- 0.04 # value chosen to compare with model fits of Sepulcure et al. (2009) and Smodej et al. (20XX)  
  
if(unknown.PAR > max(PAR.model)) {  
 stop("Your unknown sample value is outside calibration range")  
 }  
HMC.x <- which(abs(PAR.model-unknown.PAR)==min(abs(PAR.model-unknown.PAR)))  
HMC.inverse <- HMC.model[HMC.x ]# inverse fit of PAR that provides best fit to %HMC  
  
unk.PARline <- rep(unknown.PAR,j)  
  
#First, we find intersection of PAR value with upper prediction band  
x1=unk.PARline  
x2=Inverse.HMCdata[,5]  
# Find points where x1 is above x2.  
above<-x1>x2  
# Points always intersect when above=TRUE, then FALSE or reverse  
intersect.points<-which(diff(above)!=0)  
# Find the slopes for each line segment.  
x1.slopes<-x1[intersect.points+1]-x1[intersect.points]  
x2.slopes<-x2[intersect.points+1]-x2[intersect.points]  
# Find the intersection for each segment.  
x.points<-intersect.points + ((x2[intersect.points] - x1[intersect.points]) / (x1.slopes-x2.slopes))  
HMC.xlower <- HMC.model[x.points]  
HMC.xlowerC <- HMC.xlower[which(abs(HMC.xlower-HMC.inverse)==min(abs(HMC.xlower-HMC.inverse)))]  
  
#Next, we find intersection of PAR value with lower prediction band  
x1=unk.PARline  
x2=Inverse.HMCdata[,6]  
# Find points where x1 is below x2.  
below<-x1<x2  
# Points always intersect when below=TRUE, then FALSE or reverse  
intersect.points<-which(diff(below)!=0)  
# Find the slopes for each line segment.  
x1.slopes<-x1[intersect.points+1]-x1[intersect.points]  
x2.slopes<-x2[intersect.points+1]-x2[intersect.points]  
# Find the intersection for each segment.  
x.points<-intersect.points + ((x2[intersect.points] - x1[intersect.points]) / (x1.slopes-x2.slopes))  
HMC.xupper <- HMC.model[x.points]  
HMC.xupperC <- HMC.xupper[which(abs(HMC.xupper-HMC.inverse)==min(abs(HMC.xupper-HMC.inverse)))]  
  
if (length(HMC.xlowerC) == 0){  
 stop("Your unknown sample value is outside prediction band")  
 }   
if (length(HMC.xupperC) == 0){  
 stop("Your unknown sample value is outside prediction band")  
 }   
  
  
  
#display.brewer.pal(8,"Dark2")  
#pdf("unknownHMCsample\_UF.pdf", encoding = "MacRoman", width=7, height=5, useDingbats=FALSE)  
plot(C,Y, ann=FALSE,ylim=c(0,0.5),xlim=c(0,5))  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,2], col = cols[8])  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,5], col = cols[2], lty=1)  
lines(Inverse.HMCdata[,1],Inverse.HMCdata[,6], col = cols[2], lty=1)  
segments(0, unknown.PAR, HMC.xupperC, unknown.PAR,col =cols[3],lty=1,lwd=2)  
text(1, unknown.PAR-unknown.PAR/20,"Peak Area Ratio of test sample",cex=0.8)  
segments(HMC.inverse, unknown.PAR, HMC.inverse, 0,col =cols[3],lty=1,lwd=2)  
segments(HMC.xlowerC, unknown.PAR,HMC.xlowerC, 0,col =cols[3],lty=3,lwd=2)  
segments(HMC.xupperC, unknown.PAR,HMC.xupperC, 0,col =cols[3],lty=3,lwd=2)  
title(xlab="HMC Concentration (mass%)")  
title(ylab="Peak Area Ratio")  
legend(0,0.4, c("calib. data","model fit","prediction band","sample data"),col=c("black", cols[8],cols[2],cols[3]), lty=c(1,1,1,1))



#dev.off()

Provide the user with best fit and uncertainty for unknown sample from WLS regression model

cat("WLS best fit, mass %HMC: ", round(HMC.inverse,digits = 1), " ")

## WLS best fit, mass %HMC: 1.2

cat("WLS lower %HMC (95% CI): ", round(HMC.xlowerC,digits = 1)," ")

## WLS lower %HMC (95% CI): 0.8

cat("WLS upper %HMC (95% CI): ", round(HMC.xupperC,digits = 1)," ")

## WLS upper %HMC (95% CI): 2.5

## Alternative ordinary least squares calibration approach

It is possible to determine Yc,Xd,and Xq from OLS calibration regression if data are considered homoscedatic according to: Evard, H., Kruve, A., and Leito, I., 2016, Tutorial on estimating the limit of detection using LC-MS analysis, part I: Theoretical review: Analytica Chimica Acta, v. 942, p. 23–39, doi: 10.1016/j.aca.2016.08.043.

Nsamp <- length(HMCconc)  
  
#using method of Lavagnini & Magno (2006)  
newdata = data.frame(HMCconc=0)  
Yc.all <- predict(HMC.lm, newdata, interval="predict", level = 0.95, df=Nsamp-n)  
Yc <- Yc.all[3]  
Ld\_data <- data.frame(AreaRatio=Yc)  
ivtest=inverse.predict(HMC.lm,Yc,alpha=0.05)  
Xc <- ivtest$Prediction  
inverse.HMC.lm <- lm(HMCconc~AreaRatio, data=HMClo)  
Xd.all <- predict(inverse.HMC.lm, Ld\_data, interval="predict", level = 0.95, df=Nsamp-n)  
Xd <- Xd.all[3]  
  
#using method of Miller & Miller(1988) and Edvard et al. (2016)   
# Output parameters:  
Bcalc <- HMC.lm$coefficient[[2]] # estimated slope  
Syx <- ( sum(resid(HMC.lm)^2 ) / (Nsamp - 2) )^0.5 # Standard deviation of residuals  
Acalc <- HMC.lm$coefficient[[1]] # estimated intercept  
Aerror <- summary(HMC.lm)$coef[1,2] # estimated standard deviation of intercept  
CCaC <- (1.645 \* Syx)/Bcalc #also known as Yc  
CCaS <- (1.645 \* Syx)#also known as Lc or Xc  
Xd.ols <- 3.3\*Syx/Bcalc #CCb, also limit of determination in concentration, also known as LoD   
Xq.ols <- 10\*Syx/Bcalc #Limit of quantification in concentration   
Xd.int <- 3.3\* Aerror/Bcalc #limit of determination based only on varince at intercept  
cat("OLS Decision value (peak area ratio): ", round(CCaS,digits = 3), " ")

## OLS Decision value (peak area ratio): 0.067

cat("OLS Limit of detection (%HMC): ", round(Xd.ols,digits = 1)," ")

## OLS Limit of detection (%HMC): 3.7

cat("OLS Limit of quantification (%HMC): ", round(Xq.ols,digits = 1)," ")

## OLS Limit of quantification (%HMC): 11.3

The ordinary least squares decision limit (threshold) (CCa) based on peak area ratio is 0.024. The ordinary least squares limit of determination in concentration units is 1.1% HMC The ordinary least squares limit of quantification in concentration units is 3.3% HMC

We establish peak area ratio values from calibration associated with crtical 3% by mass low-Mg calcite cutoff, which is within of limit of quantification:

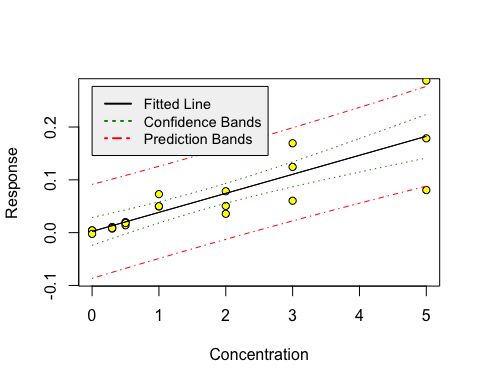
newdata = data.frame(HMCconc=3)  
predict(HMC.lm, newdata, interval="predict")

## fit lwr upr  
## 1 0.1104018 0.02220642 0.1985971

predict(HMC.lm, newdata, interval="confidence")

## fit lwr upr  
## 1 0.1104018 0.08678775 0.1340158

calplot(HMC.lm)



Calculate accuracy and precision for reverse model (95% CI) using a peak area ratio of 0.133 (predicited from linear model fit for a mass% of 3 for low-Mg calcite; see output above)

ivtest=inverse.predict(HMC.lm,0.133)  
cat("Inverse fit (%HMC): ", round(ivtest$Prediction, digits = 1), " ")

## Inverse fit (%HMC): 3.6

cat("Inverse confidence limits (%HMC): ", round(ivtest$`Confidence Limits`, digits = 1), " ")

## Inverse confidence limits (%HMC): 2.6 4.7

#ivtest$Prediction  
#ivtest$`Confidence Limits`

Accuracy and precision for linear calibration model at crtical cutoff is 3.0 +/- 0.7% based on calibration data of April 2017.

Use linear model to predict low-Mg calcite concentration from peak area ratio of unknown sample (e.g., 0.100):

ivtest=inverse.predict(HMC.lm,0.100)  
cat("Inverse fit (%HMC): ", round(ivtest$Prediction, digits = 1), " ")

## Inverse fit (%HMC): 2.7

cat("Inverse confidence limits (%HMC): ", round(ivtest$`Confidence Limits`, digits = 1), " ")

## Inverse confidence limits (%HMC): 1.7 3.7

Sample concentration is above limit of detection. Caution should be used aoplying absolute value because it is below limit of quantification.