Low-Mg Carbonate XRD data analysis with Xd,Xq; 0-5% calcite,Smodej et al.(2015) data

John M. Jaeger

June 29, 2017

## Code Description

This code is designed to import X-ray diffraction data collected by: Smodej, J., Reuning, L., Wollenberg, U., Zinke, J., Pfeiffer, M., and Kukla, P.A., 2015, Two-dimensional X-ray diffraction as a tool for the rapid, nondestructive detection of low calcite quantities in aragonitic corals: Geochemistry Geophysics Geosystems,, p. n/a–n/a, doi: 10.1002/2015GC006009.

The general calibration approach follows: Smodulcre, 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 uncertainty and detection+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.

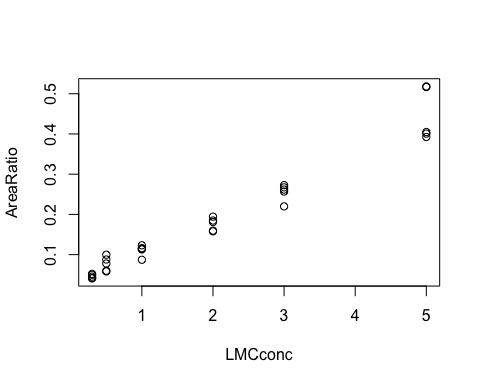
Code generated by: John M. Jaeger Associate Professor 241 Williamson Hall P.O. Box 112120 Dept. of Geological Sciences University of Florida Gainesville FL 32611-2120, USA (352) 846-1381 ORCID ID# orcid.org/0000-0003-0248-489X <http://people.clas.ufl.edu/jmjaeger/>

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

Import the data, which are reported as LMC concentration, and peak area ratios for five replicate measurements.

# load the Dataset   
XRD.data <- read.table("../raw\_data/Smodej\_2015\_LMC.csv",header=T,sep=",")   
  
n <- as.double(nrow(XRD.data)-1) #number of replicates of each calibration sample  
calib.levels <- as.double(nrow(XRD.data))#number of calibration concentration levels  
test.data <- XRD.data[,1:n+1]  
LMC.conc <- XRD.data[,1]  
  
Area.ratio.data <- stack(test.data)  
LMCconc.data <- (rep(LMC.conc,n))  
  
Smod.data <- as.data.frame(cbind(LMCconc.data,Area.ratio.data[,1]))  
colnames(Smod.data) <- c("LMCconc","AreaRatio")  
plot(Smod.data)



LMCconc <- Smod.data$LMCconc  
AreaRatio <- Smod.data$AreaRatio

We develop an OLS linear calibration model for lo-Mg calcite. We create residual plot to examine for homoscedasticity and whether a linear model is an appropraite fit.

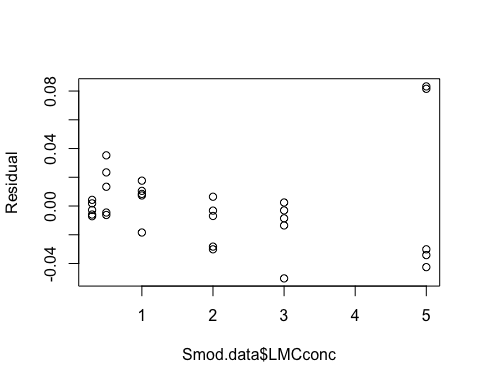
Smod.lm = lm(AreaRatio~ LMCconc, data=Smod.data)  
vcov(Smod.lm)

## (Intercept) LMCconc  
## (Intercept) 7.287156e-05 -2.185776e-05  
## LMCconc -2.185776e-05 1.111412e-05

summary(Smod.lm)

##   
## Call:  
## lm(formula = AreaRatio ~ LMCconc, data = Smod.data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.050370 -0.012279 -0.003114 0.008146 0.083232   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.023308 0.008536 2.73 0.0108 \*   
## LMCconc 0.082331 0.003334 24.70 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.02994 on 28 degrees of freedom  
## Multiple R-squared: 0.9561, Adjusted R-squared: 0.9545   
## F-statistic: 609.9 on 1 and 28 DF, p-value: < 2.2e-16

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

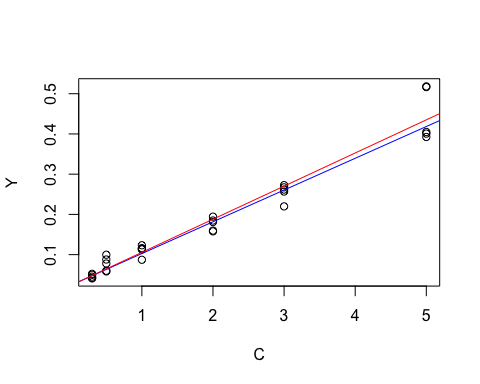
 Residuals show no trend and cluster around zero, so linear model is appropriate. Variance in residuals does have distinctive "funnel" shape with increasing lo-Mg calcite, so data are considered hetroscedastic. WLS model is approprate.

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.

Y <- Smod.data[,2]  
C <- Smod.data[,1]  
  
# Calculating weights for calibration samples  
 SdRow <- NULL  
 for(k in 1:nrow(test.data)){  
 SdRow[k] <- sd(test.data[k, ])^-2  
 }  
 Nsamp <- nrow(test.data) \* ncol(test.data)  
 sumCal <- sum(SdRow)  
 sumvar <- (sumCal) / (Nsamp)   
 w <- NULL  
 for(v in 1:nrow(test.data)){  
 w[v] <- sd(test.data[v, ])^-2 / sumvar  
 }  
 W <- rep(w, n)  
   
  
 WLS <- lm(Y ~ C, weights = W)  
 bw\_calc = lm(Y ~ C, weights = W)$coeff[2]  
 aw\_calc = lm(Y ~ C, weights = W)$coeff[1]  
 summary(WLS)

##   
## Call:  
## lm(formula = Y ~ C, weights = W)  
##   
## Weighted Residuals:  
## Min 1Q Median 3Q Max   
## -0.044836 -0.008535 0.002931 0.018991 0.047039   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.023767 0.002598 9.15 6.59e-10 \*\*\*  
## C 0.078922 0.002710 29.12 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.02419 on 28 degrees of freedom  
## Multiple R-squared: 0.968, Adjusted R-squared: 0.9669   
## F-statistic: 847.9 on 1 and 28 DF, p-value: < 2.2e-16

sigmodel <- bw\_calc\*C+aw\_calc  
 plot(C,Y)  
 abline(aw\_calc, bw\_calc,col = "blue")#WLS  
 abline(Acalc,Bcalc, col="red")#OLS



We determine Lc,Ld,and Xq from calibration regression 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.

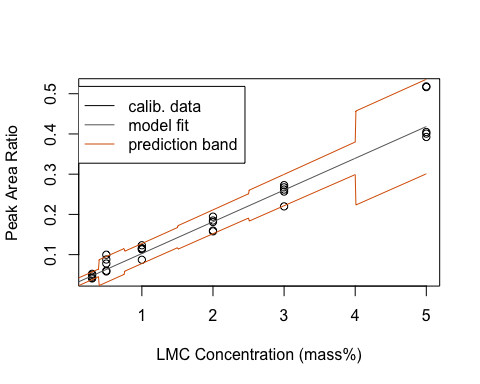
# Equations for Yc and Xd used here are from Burdge, J.R., MacTaggart, D.L., and Farwell, S.O., 1999, Realistic detection limits from confidence bands: J Chem Educ., v.76(3), p. 434-439   
  
#create model of st deviation as a function of concentration, used to model weights for intercept and unknown values  
#stval.lm <- lm(sdvalm~poly(LMCconc,2,raw=TRUE))  
p <- 0 #parameters in model of st dev versus concentration, which is zero for this because blanks were measured and used in model, 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)  
tval <- t.quantiles[2]  
m <- 1# number of replicate measurements of unknown, usually 1 for establishing limits  
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)  
Swx\_y <- sqrt((Syyw-bw\_calc\*Sxyw)/(Nsamp-2))  
Swx\_yN <- Swx\_y/sqrt(sum(W)/Nsamp)  
gw <- ((tval\*Swx\_y)^2)/(bw\_calc^2\*Sxxw)  
  
Q <- sqrt((1/(m\*W[1]))+(1/sum(W))+xbarw^2/Sxxw)  
A <- ((tval\*Swx\_y\*Q)/bw\_calc)-(xbarw\*gw)  
B <- xbarw\*gw\*(xbarw\*gw-(2\*tval\*Swx\_y\*Q/bw\_calc))  
Cv <- 1/sum(W)+xbarw^2/Sxxw+gw/(m\*W[1])  
Yc.wls <- aw\_calc+tval\*Swx\_y\*(sqrt(1/(m\*W[1])+(1/sum(W))+(xbarw^2/Sxxw)))  
Xd.wls <- (A+(sqrt(B+(gw\*Sxxw\*(((1-gw)/(m\*W[2]))+Cv)))))/(1-gw) # 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 estiamte of xd (i.w., W[2])  
Yq.wls <- 10\*(sqrt(1/SdRow[2]))+aw\_calc #From Zorn et al. (1997), eqn. 23  
Xq.wls <- (Yq.wls-aw\_calc)/bw\_calc #From Zorn et al. (1997), eqn. 24  
  
cat("WLS limit of detection (%LMC): ", round(Xd.wls,digits = 1)," ")

## WLS limit of detection (%LMC): 0.5

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

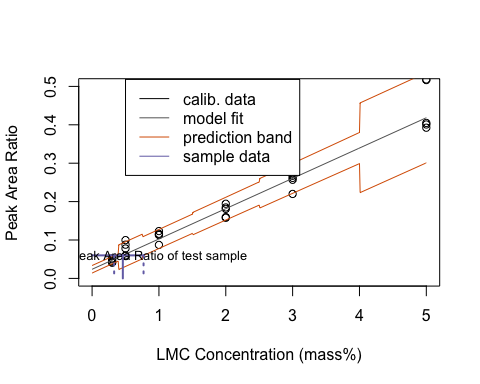
## WLS limit of quantification (%LMC): 2.3

wls.predval <- matrix(nrow = calib.levels, ncol = m)  
  
LMC.model <- seq(0, 5, by=0.01)  
PAR.model <- bw\_calc\*LMC.model+aw\_calc  
weight.modin <- cbind(LMCconc,w)  
  
  
j <- length(PAR.model)  
wls.predvaltest <- matrix(nrow = j, ncol = 1)  
for(jv in 1:j){  
 lookvalue <- LMC.model[jv]  
 weightx <- which(abs(LMCconc-lookvalue)==min(abs(LMCconc-lookvalue)))  
 wls.predvaltest[jv,1] <- weight.modin[weightx[1],2]  
}  
Inverse.LMCdata <- cbind(LMC.model,PAR.model,wls.predvaltest)  
wls.predval2 <- matrix(nrow = j, ncol = m)  
#prediction band equation from Zorn et al., equation 13  
for(pv in 1:j){  
 wls.predval2[pv,1] <-tval\*Swx\_y\*(sqrt(1/(m\*wls.predvaltest[pv,1])+(1/sum(W))+((Inverse.LMCdata[pv,1]-xbarw)^2/Sxxw)))  
}  
predband.uM <- PAR.model+wls.predval2  
predband.lM <- PAR.model-wls.predval2  
  
Inverse.LMCdata <- cbind(LMC.model,PAR.model,wls.predvaltest,wls.predval2,predband.uM,predband.lM)  
colnames(Inverse.LMCdata) <- c("LMC.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.LMCdata[,1],Inverse.LMCdata[,2], col = cols[8])  
lines(Inverse.LMCdata[,1],Inverse.LMCdata[,5], col = cols[2], lty=1)  
lines(Inverse.LMCdata[,1],Inverse.LMCdata[,6], col = cols[2], lty=1)  
#segments(0, Yc.wls, LMC.xupperC, unknown.PAR,col =cols[3],lty=1,lwd=2)  
title(xlab="LMC 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 a dataframe model of fit and uncertainity to inverse modeling

unknown.PAR <- 0.06  
  
if(unknown.PAR > max(PAR.model)) {  
 stop("Your unknown sample value is outside calibration range")  
 }  
LMC.x <- which(abs(PAR.model-unknown.PAR)==min(abs(PAR.model-unknown.PAR)))  
LMC.inverse <- LMC.model[LMC.x ]  
  
unk.PARline <- rep(unknown.PAR,j)  
#find intersection with upper prediction band  
x1=unk.PARline  
x2=Inverse.LMCdata[,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))  
LMC.xlower <- LMC.model[x.points]  
LMC.xlowerC <- LMC.xlower[which(abs(LMC.xlower-LMC.inverse)==min(abs(LMC.xlower-LMC.inverse)))]  
  
#find intersection with lower prediction band  
x1=unk.PARline  
x2=Inverse.LMCdata[,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))  
LMC.xupper <- LMC.model[x.points]  
LMC.xupperC <- LMC.xupper[which(abs(LMC.xupper-LMC.inverse)==min(abs(LMC.xupper-LMC.inverse)))]  
  
if (length(LMC.xlowerC) == 0){  
 stop("Your unknown sample value is outside prediction band")  
 }   
if (length(LMC.xupperC) == 0){  
 stop("Your unknown sample value is outside prediction band")  
 }   
  
  
  
#display.brewer.pal(8,"Dark2")  
  
#pdf("unknownLMCsample\_smodej.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.LMCdata[,1],Inverse.LMCdata[,2], col = cols[8])  
lines(Inverse.LMCdata[,1],Inverse.LMCdata[,5], col = cols[2], lty=1)  
lines(Inverse.LMCdata[,1],Inverse.LMCdata[,6], col = cols[2], lty=1)  
segments(0, unknown.PAR, LMC.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(LMC.inverse, unknown.PAR, LMC.inverse, 0,col =cols[3],lty=1,lwd=2)  
segments(LMC.xlowerC, unknown.PAR,LMC.xlowerC, 0,col =cols[3],lty=3,lwd=2)  
segments(LMC.xupperC, unknown.PAR,LMC.xupperC, 0,col =cols[3],lty=3,lwd=2)  
title(xlab="LMC Concentration (mass%)")  
title(ylab="Peak Area Ratio")  
legend(0.5, max(Y), 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()

We determine Lc,Ld,and Xq from calibration regression where data are 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(Y)  
#using method of Lavagnini & Magno (2006)  
newdata = data.frame(LMCconc=0)  
Lc.all <- predict(Smod.lm, newdata, interval="predict", level = 0.95, df=Nsamp-n)  
Lc <- Lc.all[3]  
Ld\_data <- data.frame(AreaRatio=Lc)  
ivtest=inverse.predict(Smod.lm,Lc,alpha=0.05)  
Xc <- ivtest$Prediction  
inverse.Smod.lm <- lm(LMCconc~AreaRatio, data=Smod.data)  
Xd.all <- predict(inverse.Smod.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 <- Smod.lm$coefficient[[2]] # estimated slope  
Syx <- ( sum(resid(Smod.lm)^2 ) / (Nsamp - 2) )^0.5 # Standard deviation of residuals  
Acalc <- Smod.lm$coefficient[[1]] # estimated intercept  
Aerror <- summary(Smod.lm)$coef[1,2] # estimated standard deviation of intercept  
CCaC <- (1.645 \* Syx)/Bcalc  
CCaS <- (1.645 \* Syx)  
Xd.resid <- 3.3\*Syx/Bcalc #CCb, also limit of determination in concentration   
Xq.resid <- 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): 0.049

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

## OLS Limit of detection (%LMC): 1.2

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

## OLS Limit of quantification (%LMC): 3.6

The ordinary least squares decision limit (threshold) (CCa) based on peak area ratio is 0.049. The ordinary least squares limit of determination in concentration units is 1.2% LMC The ordinary least squares limit of quantification in concentration units is 3.6% LMC

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

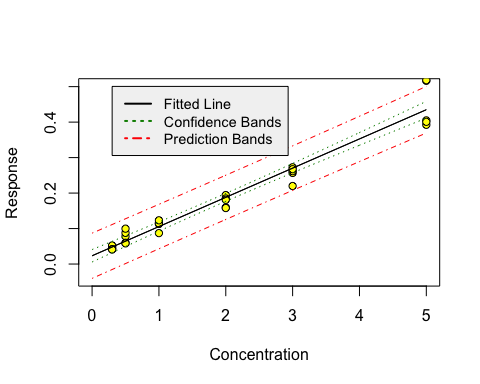
newdata = data.frame(LMCconc=3)  
predict(Smod.lm, newdata, interval="predict")

## fit lwr upr  
## 1 0.2703008 0.2075549 0.3330466

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

## fit lwr upr  
## 1 0.2703008 0.2570648 0.2835367

calplot(Smod.lm)



Calculate accuracy and precision for reverse model (95% CI) using a peak area ratio of 0.27 (predicited from linear model fit for a mass% of 3 for low-Mg calcite; see output above)

ivtest=inverse.predict(Smod.lm,0.27)  
cat("Inverse fit (%LMC): ", round(ivtest$Prediction, digits = 1), " ")

## Inverse fit (%LMC): 3

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

## Inverse confidence limits (%LMC): 2.5 3.5

#ivtest$Prediction  
#ivtest$`Confidence Limits`

accuracy and precision for linear calibration model at crtical cutoff is 3.0 +/- 0.5%

Use linear model to predict low-Mg calcite concentration from peak area ratio of unknown sample (e.g., 0.300):

ivtest=inverse.predict(Smod.lm,0.3)  
cat("Inverse fit (%LMC): ", round(ivtest$Prediction, digits = 1), " ")

## Inverse fit (%LMC): 3.4

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

## Inverse confidence limits (%LMC): 2.9 3.9

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