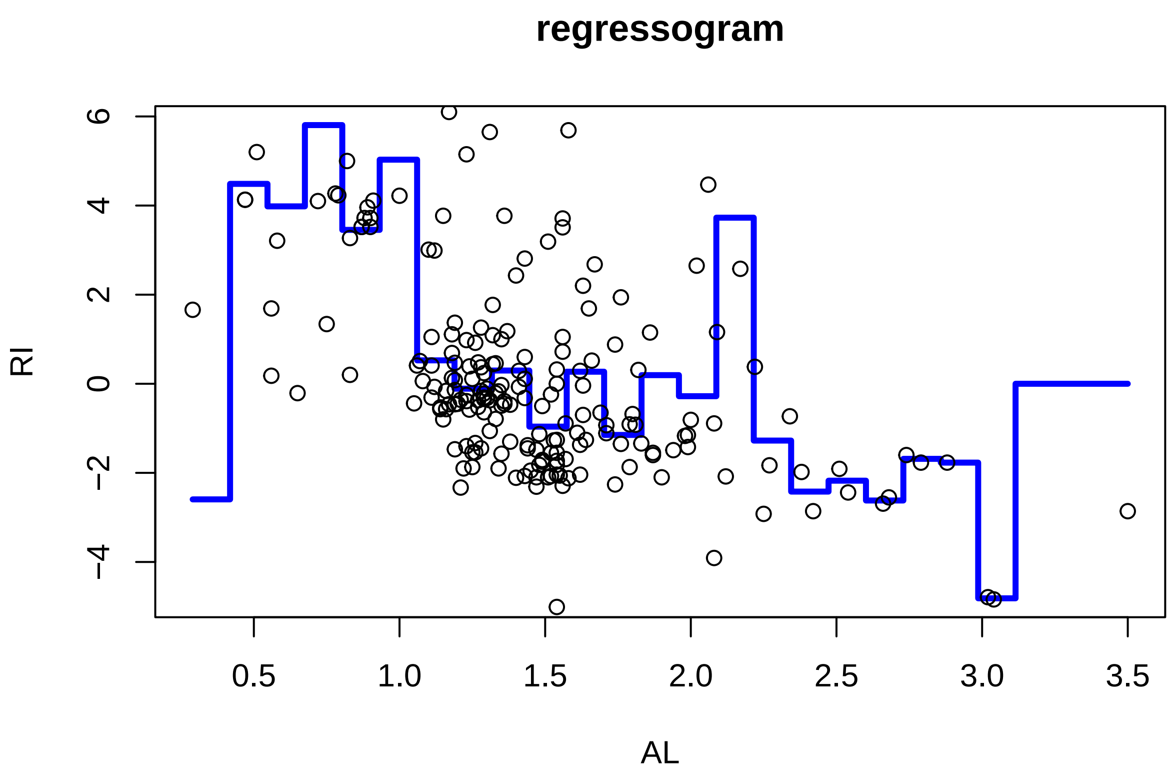
A4Q5

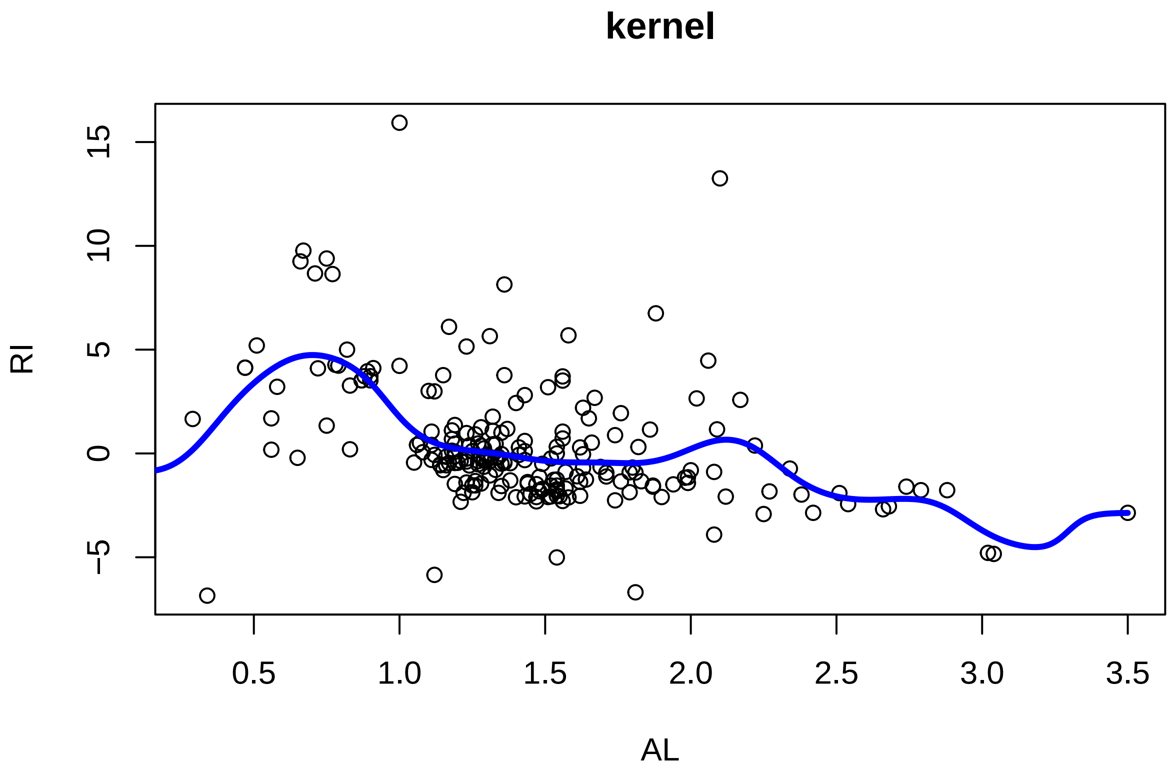
Undergraduate Student

```
glass <- read.table("glass.dat",header=T)
```

```
regressogram = function(x,y,lower,upper,nbins){
  n = length(x)
  B = seq(lower,upper,length=nbins+1)
  which = findInterval(x,B)
  N = tabulate(which)
  f.hat = rep(0,nbins)
  for(j in 1:nbins){
    if(N[j]>0) {
      f.hat[j] = mean(y[which == j])
    }
  }
  minimum = min(c(y,f.hat))
  maximum = max(c(y,f.hat))
  plot(B,c(f.hat,f.hat[nbins]),lwd=3,type="s",col="blue", main="regressogram",
       xlab="AL", ylab="RI")
  points(x,y)
  return(list(bins=B,f.hat=f.hat))
}

kernel = function(x,y,points,width){
  n = length(x)
  m = length(points)
  f.hat = rep(0,m)
  for(i in 1:m){
    weight = dnorm(points[i],x,width)
    f.hat[i] = sum(y*weight)/sum(weight)
  }
  return(f.hat)
}


value_fitted = function(x,y,h){
  n = length(x)
  f.hat = rep(0,n)
  S = rep(0,n)
  for(i in 1:n){
    weight = dnorm(x[i],x,h)
    weight = weight/sum(weight)
    f.hat[i] = sum(y*weight)
    S[i] = weight[i]
  }
  return(list(fitted=f.hat,S=S))
}

CV = function(x,y,width){
  n = length(x)
  m = length(width)
  cv = rep(0,m)
  nu = rep(0,m)
  gcv = rep(0,m)
  for(i in 1:m){
    tmp = value_fitted(x,y,width[i])
    cv[i] = mean(((y - tmp$fitted)/(1-tmp$S))^2)
    nu[i] = sum(tmp$S)
    gcv[i] = mean((y - tmp$fitted)^2)/(1-nu[i]/n)^2
  }
  return(list(cv=cv,gcv=gcv,nu=nu))
}

H <-seq(0.1,max(glass$Al),0.01)
cv_result = CV(glass$Al,glass$RI,H)


points = seq(min(glass$Al), max(glass$Al), length=100)
binwidth=H[which.min(cv_result$cv)]
nbins = round((max(glass$Al)-min(glass$Al))/binwidth)

#regressogram
stat <- regressogram(glass$Al, glass$RI, min(glass$Al), max(glass$Al),
                     nbins)
```



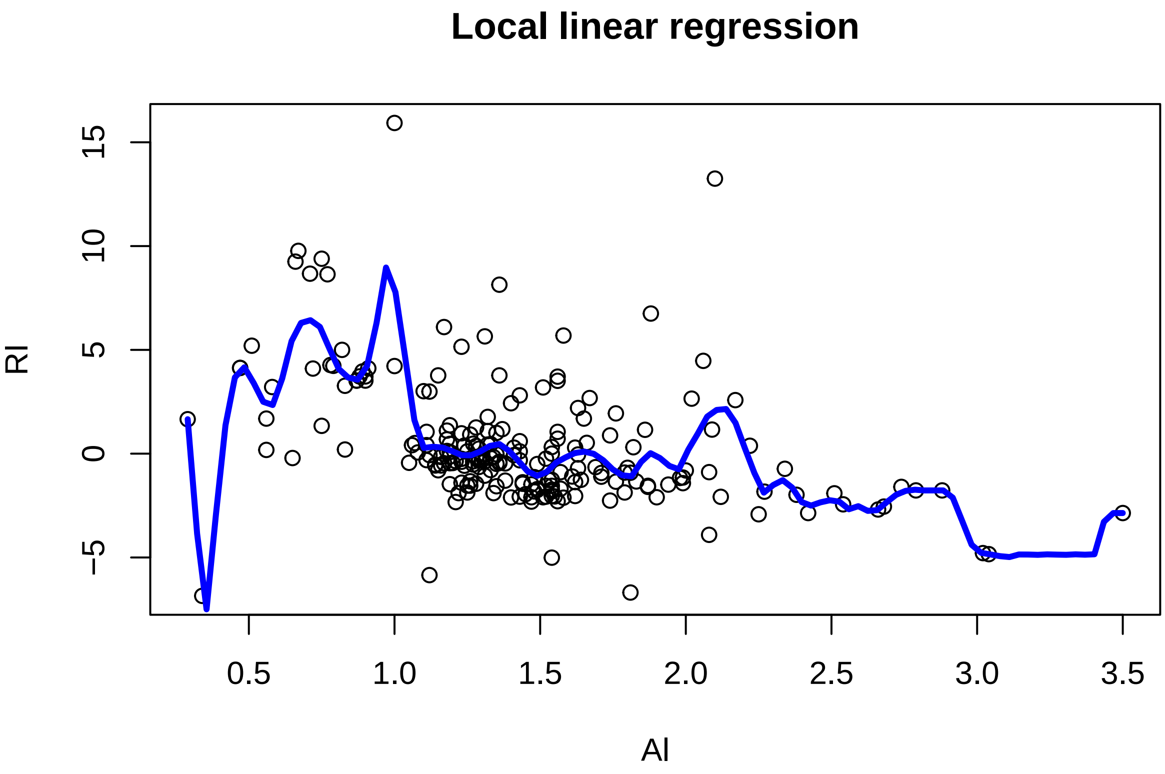
```
# kernel
estimate <- kernel(glass$Al, glass$RI, H, binwidth)
plot(glass$Al,glass$RI,xlab="AL",ylab="RI", main="kernel")
lines(H, estimate, col="blue", lwd=3)
```



```
library(locfit)
```

```
## locfit 1.5-9.5    2022-03-01
```

```
h<-seq(0.1,max(glass$Al),0.01)
alphamatrix <- matrix(0,ncol=2,nrow=length(h))
alphamatrix[,2] <- h
gcvRIlocfit <- gcvplot(glass$RI~glass$Al,alpha=alphamatrix,deg=1)
opt.h <- max(gcvRIlocfit$alpha[gcvRIlocfit$values==min(gcvRIlocfit$values),2])
RIlocfit.opt<-locfit(glass$RI~glass$Al,alpha=c(0,opt.h),deg=1)
plot(glass$Al,glass$RI,xlab="Al",ylab="RI",main="Local linear regression")
lines(RIlocfit.opt,lwd=3,col="blue")
```



```
smooth.matrix<-function(x,df){
  n<-length(x)
  A<-matrix(0,n,n)
  for (i in 1:n){
    y<-rep(0,n)
    y[i]=1
    yi=predict(smooth.spline(x,y,df=df),x)$y
    A[,i]=yi
  }
  return((A+t(A))/2)
}
summary(RIlocfit.opt)
```

```
## Estimation type: Local Regression
```

```
##
## Call:
## locfit(formula = glass$RI ~ glass$Al, alpha = c(0, opt.h), deg = 1)
##
## Number of data points:  214
## Independent variables:  glass$Al
## Evaluation structure: Rectangular Tree
## Number of evaluation points:  65
## Degree of fit:  1
## Fitted Degrees of Freedom:  25.472
```

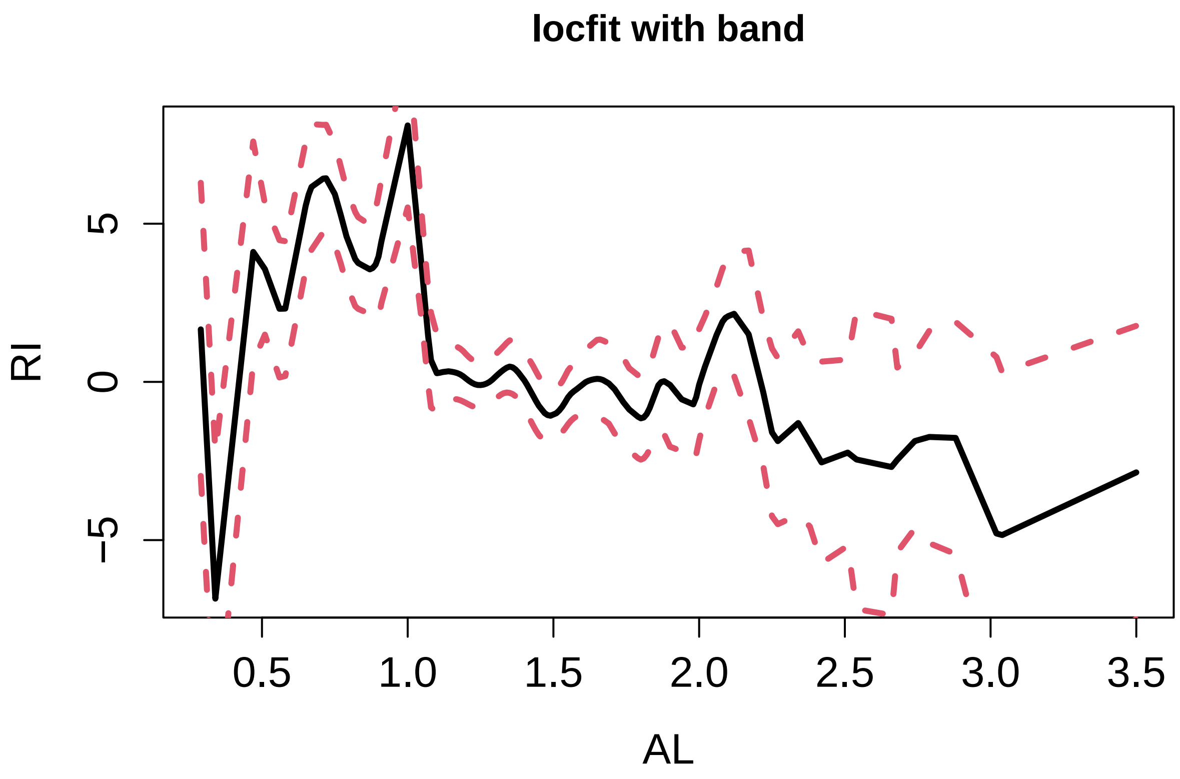
```
#variance
S<-smooth.matrix(glass$Al,25.472)
num <- sum((glass$RI-predict(RIlocfit.opt,glass$Al))^2)
denum<-length(glass$Al)-2*sum(diag(S))+sum(diag(t(S)%*%S))
sigma2hat1<-num/denum
sigma2hat1
```

```
## [1] 5.542919
```

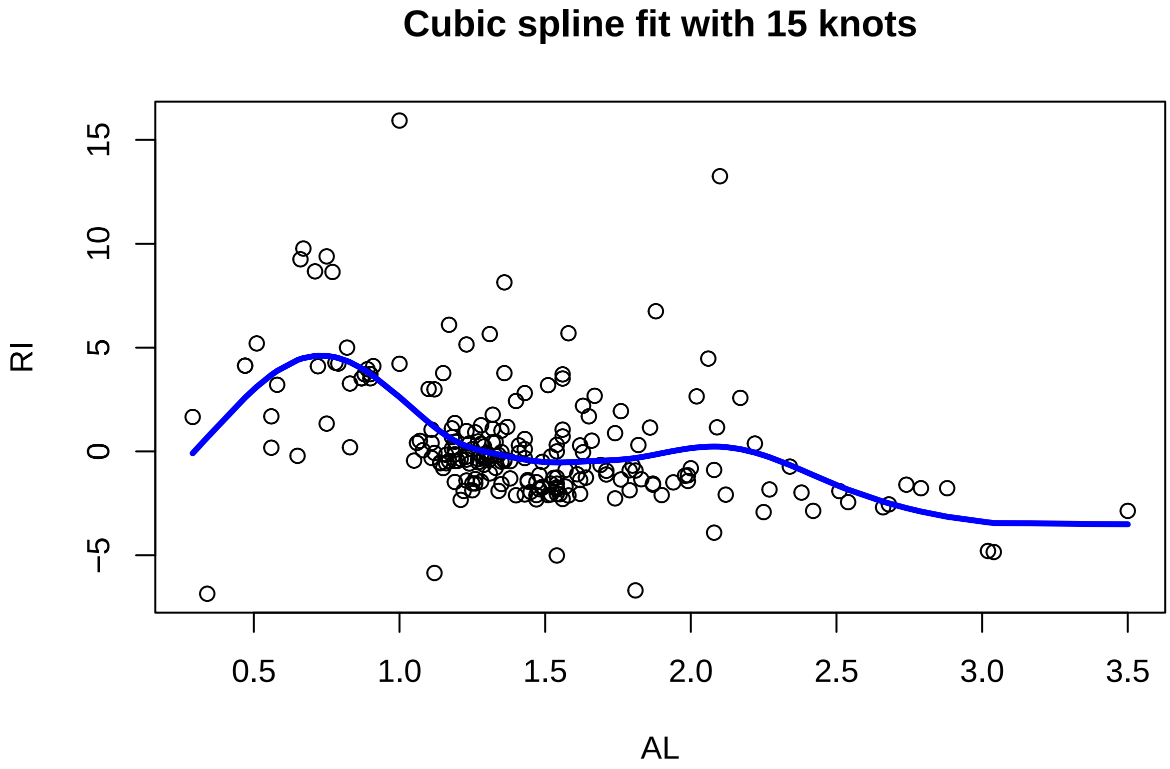
```
nu1 <- as.numeric(RIlocfit.opt$dp[6])
nu2 <- as.numeric(RIlocfit.opt$dp[7])
lfsigma2hat1 <- sum(residuals(RIlocfit.opt)^2)/(length(glass$Al)-2*nu1+nu2)
lfsigma2hat1
```

```
## [1] 5.585231
```

```
diaghat <- predict(RIlocfit.opt, where="data", what="inf1")
norm.s <- predict(RIlocfit.opt, where="data", what="vari")
critval <- kappa0(RIlocfit.opt, cov=0.975)$crit.val
locfitRIpred <- predict(RIlocfit.opt, where="data")
o<-order(glass$Al)
plot(glass$Al[o],locfitRIpred[o],lwd=3,xlab="AL",ylab="RI",
     cex=3,cex.axis=1.3,cex.lab=1.3,type="l", main="locfit with band")
lines(glass$Al[o],locfitRIpred[o]+qnorm(0.975)*sqrt(lfsigma2hat1*norm.s[o]),
      col=2,lwd=3,lty=2)
lines(glass$Al[o],locfitRIpred[o]-qnorm(0.975)*sqrt(lfsigma2hat1*norm.s[o]),
      col=2,lwd=3,lty=2)
```



```
splinefit1<-smooth.spline(glass$Al,glass$RI,cv=TRUE, all.knots=FALSE,nknots=15)
plot(glass$Al,glass$RI,xlab="AL", ylab="RI",main="Cubic spline fit with 15 knots")
lines(splinefit1,lwd=3,col="blue")
```



```r
#variance
df <- splinefit1$df
S<-smooth.matrix(glass$Al,df)
num <- sum((glass$RI-predict(splinefit1,glass$Al)$y)^2)
denum<-length(glass$Al)-2*sum(diag(S))+sum(diag(t(S)%*%S))
sigma2hat1<-num/denum
sigma2hat1
```

```
## [1] 6.473874
```

Comment: From four graphs, we see that cubic spline has the most smooth curve while it follows the trend of the data, therefore it best represents the data. Local linear regression seems to try to cover most of the points, but it becomes not smooth. Kernel method has a smooth curve, but it has a high variation.